7.5

IBM WebSphere MQ -Guide de référence des applications de développement

Remarque

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales figurant à la section «Remarques», à la page 1495.

Cette édition s'applique à la version 7 édition 5 d' IBM® WebSphere MQ et à toutes les éditions et modifications ultérieures, sauf indication contraire dans les nouvelles éditions.

Table des matières

Références relatives au développement d'applications

Utilisez les informations de cette section pour vous aider à développer vos applications IBM WebSphere MQ :

Tâches associées

Développement d'applications

Référence des applications MQI

Utilisez les liens fournis dans cette section pour vous aider à développer vos applications MQI:

- «Exemples de code», à la page 8
- «Constantes», à la page 50
- «Types de données utilisés dans l'interface MQI», à la page 217
- «Appels de fonctions», à la page 608
- «Attributs des objets», à la page 787
- «Codes retour», à la page 863
- «Règles de validation des options MQI», à la page 864
- «Codages de machine», à la page 891
- «Options de rapport et indicateurs de message», à la page 894
- «Exit de conversion de données», à la page 898
- «Propriétés spécifiées en tant qu'éléments MQRFH2», à la page 921
- «Conversion de page de code», à la page 930

Concepts associés

«Références des exits utilisateur, des exits API et des services installables», à la page 1022
Utilisez les liens fournis dans cette section pour vous aider à développer vos exits utilisateur, exits API et applications de services installables:

«Les classes IBM WebSphere MQ pour les bibliothèques Java», à la page 1438
L'emplacement des bibliothèques IBM WebSphere MQ classes for Java varie en fonction de la plateforme. Indiquez cet emplacement lorsque vous démarrez une application.

Tâches associées

Développement d'applications

Référence associée

«Référence SOAP», à la page 964
Transport WebSphere MQ pour les informations de référence SOAP classées par ordre alphabétique.

«Informations de référence pour le pont IBM WebSphere MQ pour HTTP», à la page 1226
Rubriques de référence pour IBM WebSphere MQ bridge for HTTP, classées par ordre alphabétique

«Les classes et interfacesIBM WebSphere MQ .NET», à la page 1262
Les classes et interfaces IBM WebSphere MQ .NET sont répertoriées par ordre alphabétique. Les propriétés, les méthodes et les constructeurs sont décrits.

«IBM WebSphere MQ classes C++», à la page 1325
Les classes C++ IBM WebSphere MQ encapsulent l'interface MQI (IBM WebSphere MQ Message Queue Interface). Il existe un fichier d'en-tête C++ unique, **imqi.hpp**, qui couvre toutes ces classes.

Information associée

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

Exemples de code

Utilisez les informations de référence présentées dans cette section pour accomplir les tâches adaptées à vos besoins métier.

Exemples de langage C

Cet ensemble de rubriques est principalement issu des modèles d'application WebSphere MQ for z/OS . Elles sont applicables à toutes les plateformes, sauf indication contraire.

Connexion à un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQCONN pour connecter un programme à un gestionnaire de files d'attente dans un lot z/OS .

Cet extrait est extrait du modèle d'application Parcourir (programme CSQ4BCA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS).

```
#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;
      }
.
   }
```

Déconnexion d'un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQDISC pour déconnecter un programme d'un gestionnaire de files d'attente dans un lot z/OS .

Les variables utilisées dans cette extraction de code sont celles qui ont été définies dans «Connexion à un gestionnaire de files d'attente», à la page 8. Cet extrait est extrait du modèle d'application Parcourir (programme CSQ4BCA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

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

Création d'une file d'attente dynamique

Cet exemple montre comment utiliser l'appel MQOPEN pour créer une file d'attente dynamique.

Cet extrait provient de l'exemple d'application Mail Manager (programme CSQ4TCD1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

```
⋮
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;
}
⋮
```

Ouverture d'une file d'attente existante

Cet exemple montre comment utiliser l'appel MQOPEN pour ouvrir une file d'attente qui a déjà été définie.

Cet extrait est extrait du modèle d'application Parcourir (programme CSQ4BCA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS).

```
#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 */
```

Fermeture d'une file d'attente

Cet exemple montre comment utiliser l'appel MQCLOSE pour fermer une file d'attente.

Cet extrait est extrait du modèle d'application Parcourir (programme CSQ4BCA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS).

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

Insertion d'un message à l'aide de MQPUT

Cet exemple montre comment utiliser l'appel MQPUT pour placer un message dans une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ. Pour connaître les noms et les emplacements des exemples d'application, voir Exemples de programmes (plateformes sauf z/OS).

```
...
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 */
        }
}
```

Insertion d'un message à l'aide de MQPUT1

Cet exemple montre comment utiliser l'appel MQPUT1 pour ouvrir une file d'attente, insérer un message unique dans la file d'attente, puis fermer la file d'attente.

Cet extrait provient de l'exemple d'application Credit Check (programme CSQ4CCB5) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

```
⋮
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;
   }
⋮
```

obtention d'un message

Cet exemple montre comment utiliser l'appel MQGET pour supprimer un message d'une file d'attente.

Cet extrait est extrait du modèle d'application Parcourir (programme CSQ4BCA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS).

```
#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 */
```

Obtention d'un message à l'aide de l'option d'attente

Cet exemple montre comment utiliser l'option d'attente de l'appel MQGET.

Ce code accepte les messages tronqués. Cet extrait provient de l'exemple d'application Credit Check (programme CSQ4CCB5) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS).

```
.
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();
         }
⋮
```

Obtention d'un message à l'aide de la notification

La signalisation est disponible uniquement avec WebSphere MQ for z/OS.

Cet exemple montre comment utiliser l'appel MQGET pour définir un signal afin d'être averti lorsqu'un message approprié arrive dans une file d'attente. Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
⋮
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;
}
```

Interrogation des attributs d'un objet

Cet exemple montre comment utiliser l'appel MQINQ pour demander les attributs d'une file d'attente.

Cette extraction est extraite de l'exemple d'application Attributs de file d'attente (programme CSQ4CCC1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

```
#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 */
```

Définition des attributs d'une file d'attente

Cet exemple montre comment utiliser l'appel MQSET pour modifier les attributs d'une file d'attente.

Cette extraction est extraite de l'exemple d'application Attributs de file d'attente (programme CSQ4CCC1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

```
#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 */
```

Extraction des informations de statut à l'aide de MQSTAT

Cet exemple montre comment émettre une commande MQPUT asynchrone et extraire les informations de statut avec MQSTAT.

Cette extraction est extraite de l'exemple d'application Appeler MQSTAT (programme amqsapt0) fourni avec WebSphere MQ pour les systèmes Windows . Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

```
/*********************************************************************/
/*                                                                   */
/* 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);
}
```

Exemples COBOL

Cette collection de rubriques provient des exemples d'applications WebSphere MQ for z/OS . Elles sont applicables à toutes les plateformes, sauf indication contraire.

Connexion à un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQCONN pour connecter un programme à un gestionnaire de files d'attente dans un lot z/OS .

Cet extrait est extrait du modèle d'application Browse (programme CSQ4BVA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS).

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

Déconnexion d'un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQDISC pour déconnecter un programme d'un gestionnaire de files d'attente dans un lot z/OS .

Les variables utilisées dans cette extraction de code sont celles qui ont été définies dans «Connexion à un gestionnaire de files d'attente», à la page 23. Cet extrait est extrait du modèle d'application Browse (programme CSQ4BVA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

```
⋮
*
* 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.
⋮
```

Création d'une file d'attente dynamique

Cet exemple montre comment utiliser l'appel MQOPEN pour créer une file d'attente dynamique.

Cet extrait est extrait de l'exemple d'application Credit Check (programme CSQ4CVB1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

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

Ouverture d'une file d'attente existante

Cet exemple montre comment utiliser l'appel MQOPEN pour ouvrir une file d'attente existante.

Cet extrait est extrait du modèle d'application Browse (programme CSQ4BVA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS).

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

Fermeture d'une file d'attente

Cet exemple montre comment utiliser l'appel MQCLOSE.

Les variables utilisées dans cette extraction de code sont celles qui ont été définies dans «Connexion à un gestionnaire de files d'attente», à la page 23. Cet extrait est extrait du modèle d'application Browse (programme CSQ4BVA1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

```
⋮
*
*    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.
*
```

Insertion d'un message à l'aide de MQPUT

Cet exemple montre comment utiliser l'appel MQPUT à l'aide du contexte.

Cet extrait est extrait de l'exemple d'application Credit Check (programme CSQ4CVB1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

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

Insertion d'un message à l'aide de MQPUT1

Cet exemple montre comment utiliser l'appel MQPUT1 .

Cet extrait provient de l'exemple d'application Credit Check (programme CSQ4CVB5) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

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

obtention d'un message

Cet exemple montre comment utiliser l'appel MQGET pour supprimer un message d'une file d'attente.

Cet extrait est extrait de l'exemple d'application Credit Check (programme CSQ4CVB1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

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

Obtention d'un message à l'aide de l'option d'attente

Cet exemple montre comment utiliser l'appel MQGET avec l'option d'attente et accepter les messages tronqués.

Cet extrait provient de l'exemple d'application Credit Check (programme CSQ4CVB5) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf 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-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.
⋮
```

Obtention d'un message à l'aide de la notification

Cet exemple montre comment utiliser l'appel MQGET avec la notification. Cet extrait provient de l'exemple d'application Credit Check (programme CSQ4CVB2) fourni avec WebSphere MQ for z/OS.

La signalisation est disponible uniquement avec 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
*
⋮
```

Interrogation des attributs d'un objet

Cet exemple montre comment utiliser l'appel MQINQ pour demander les attributs d'une file d'attente.

Cet extrait est extrait de l'exemple d'application Attributs de file d'attente (programme CSQ4CVC1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS) .

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

Définition des attributs d'une file d'attente

Cet exemple montre comment utiliser l'appel MQSET pour modifier les attributs d'une file d'attente.

Cet extrait est extrait de l'exemple d'application Attributs de file d'attente (programme CSQ4CVC1) fourni avec WebSphere MQ for z/OS. Pour connaître les noms et les emplacements des exemples d'application sur d'autres plateformes, voir Exemples de programmes (plateformes sauf z/OS)

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

Exemples de langage assembleur System/390

Cet ensemble de rubriques est principalement issu des modèles d'application WebSphere MQ for z/OS .

Connexion à un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQCONN pour connecter un programme à un gestionnaire de files d'attente dans un lot z/OS .

Cet extrait est extrait de l'exemple de programme Browse (CSQ4BAA1) fourni avec WebSphere MQ for z/OS.

```
⋮
WORKAREA DSECT
*
PARMLIST CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
```

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

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

Déconnexion d'un gestionnaire de files d'attente

Cet exemple montre comment utiliser l'appel MQDISC pour déconnecter un programme d'un gestionnaire de files d'attente dans un lot z/OS .

Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
⋮
*
*        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
```

Création d'une file d'attente dynamique

Cet exemple montre comment utiliser l'appel MQOPEN pour créer une file d'attente dynamique.

Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
⋮
*
*    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
```

Ouverture d'une file d'attente existante

Cet exemple montre comment utiliser l'appel MQOPEN pour ouvrir une file d'attente qui a déjà été définie.

Il montre comment spécifier deux options. Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
:
*
*    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
```

Fermeture d'une file d'attente

Cet exemple montre comment utiliser l'appel MQCLOSE pour fermer une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
⋮
*
* 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
```

Insertion d'un message à l'aide de MQPUT

Cet exemple montre comment utiliser l'appel MQPUT pour placer un message dans une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
⋮
*     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
```

Insertion d'un message à l'aide de MQPUT1

Cet exemple montre comment utiliser l'appel MQPUT1 pour ouvrir une file d'attente, insérer un message unique dans la file d'attente, puis fermer la file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
⋮
*
*     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
```

obtention d'un message

Cet exemple montre comment utiliser l'appel MQGET pour supprimer un message d'une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
⋮
*
*     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
```

Obtention d'un message à l'aide de l'option d'attente

Cet exemple montre comment utiliser l'option d'attente de l'appel MQGET.

Ce code accepte les messages tronqués. Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
.
*     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
```

Obtention d'un message à l'aide de la notification

Cet exemple montre comment utiliser l'appel MQGET pour définir un signal afin d'être averti lorsqu'un message approprié arrive dans une file d'attente.

Cet extrait n'est pas extrait des exemples d'application fournis avec WebSphere MQ.

```
⋮
*
*     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
```

Interrogation et définition des attributs d'une file d'attente

Cet exemple montre comment utiliser l'appel MQINQ pour consulter les attributs d'une file d'attente et utiliser l'appel MQSET pour modifier les attributs d'une file d'attente.

Cet extrait est extrait de l'exemple d'application Attributs de file d'attente (programme CSQ4CAC1) fourni avec WebSphere MQ for z/OS.

```
:
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),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?
⋮
```

Constantes

Utilisez les informations de référence présentées dans cette section pour accomplir les tâches adaptées à vos besoins métier.

Fichiers COPY, d'en-tête, d'inclusion et de module IBM WebSphere MQ

Ces informations sont des informations générales sur l'interface de programmation.

Cette section contient des informations pour vous aider à utiliser l'interface MQI pour différents langages de programmation, comme suit.

Fichiers d'en-tête C

Des fichiers d'en-tête sont fournis pour vous aider à écrire des programmes d'application C qui utilisent l'interface MQI. Ces fichiers d'en-tête sont récapitulés dans le tableau suivant:

Tableau 1. Fichiers d'en-tête C-appeler des prototypes, des types de données, des codes retour, des constantes et des structures					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX et Linux®**	**Windows**	**z/OS**
Appeler des prototypes, des types de données, des codes retour, des constantes et des structures					
CMQC	Définitions MQI	C	C	C	C
CMQBC	Définitions MQAI	C	C	C	
CMQEC	Définition des points d'entrée d'interface (inclut CMQC, CMQXC et CMQZC)		C	C	
CMQCFC	Définitions PCF	C	C	C	C
CMQPSC	Définitions de publication / abonnement	C	C	C	C
CMQXC	Définitions de canal et de sortie	C	C	C	C
CMQZC	Définitions de services installables	C	C	C	
Clé: C= Fichiers fournis					

Fichiers COBOL COPY

Divers fichiers COPY sont fournis pour vous aider à écrire des programmes d'application COBOL qui utilisent l'interface MQI. Ces fichiers sont récapitulés dans le tableau suivant:

Tableau 2. Fichiers de copie COBOL-codes retour, constantes et structures					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX**	**Windows**	**z/OS**
Codes retour et constantes					
CMQx	Définitions MQI	V	V	V	V
CMQCFx	Définitions PCF	V	V	V	V
CMQPSx	Définitions de publication / abonnement	V	V	V	V
CMQXx	Définitions de canal et de sortie	V	V	V	V
structures					
Récept. MQ	MQAIR-Enregistrement des informations d'authentification		V L	V L	
CMQBOx	MQBO-Options de début	V L	V L	V L	
CMQCDx	MQCD-Définition de canal	V L	V L	V L	V L
CMQCFBFx	MQCFBF-Paramètre de filtre de chaîne d'octets PCF	V L	V L	V L	V L
CMQCFBSx	MQCFBS-Paramètre de chaîne d'octets PCF	V L	V L	V L	V L
CMQCFGRx	MQCFGR-Paramètre de groupe PCF	V L	V L	V L	V L

Tableau 2. Fichiers de copie COBOL-codes retour, constantes et structures (suite)					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX**	**Windows**	**z/OS**
CMQCFHx	En-tête MQCFH-PCF	V L	V L	V L	V L
CMQCFIFx	MQCFIF-Paramètre de filtre d'entier PCF	V L	V L	V L	V L
CMQCFILx	MQCFIL-Paramètre de liste d'entiers PCF	V L	V L	V L	V L
CMQCFINx	MQCFIN-Paramètre entier PCF	V L	V L	V L	V L
CMQCFSFx	MQCFSF-Paramètre de filtre de chaîne PCF	V L	V L	V L	V L
CMQCFSLx	MQCFSL-Paramètre de liste de chaînes PCF	V L	V L	V L	V L
CMQCFSTx	MQCFST-Paramètre de chaîne PCF	V L	V L	V L	V L
CMQCFXLx	MQCFIL64 -Paramètre de liste d'entiers PCF 64 bits	V L	V L	V L	V L
CMQCFXNx	MQCFIN64 -Paramètre d'entier PCF 64 bits	V L	V L	V L	V L
CMQCHRVx	MQCHARV-Chaîne de longueur variable	V L	V L	V L	V L
CMQCIHx	MQCIH-en-tête de pont CICS	V L	V L	V L	V L
CMQCNOx	MQCNO-Options de connexion	V L	V L	V L	V L
CMQCSPx	MQCSP-Paramètres de sécurité	V L	V L	V L	V L
CMQCXPx	MQCXP-Paramètres d'exit de canal	V L			V L
CMQDHx	En-tête MQDH-Distribution	V L	V L	V L	V L
CMQDLHx	En-tête MQDLH-Dead-letter	V L	V L	V L	V L
CMQDXPx	MQDXP-Paramètres d'exit de conversion de données	V L		V L	
CMQEPHx	MQEPH-en-tête PCF imbriqué	V L	V L	V L	V L
CMQGMOx	MQGMO-Extraction des options de message	V L	V L	V L	V L
CMQIIHx	MQIIH-en-tête d'informations IMS	V L	V L	V L	V L
CMQMDx	MQMD-Descripteur de message	V L	V L	V L	V L
CMQMD1x	MQMD1 -Descripteur de message version 1	V L	V L	V L	V L
CMQMD2x	MQMD2 -Descripteur de message version 2	V L	V L	V L	V L
CMQMDEx	MQMDE-Descripteur de message étendu	V L	V L	V L	V L
CMQODx	MQOD-Descripteur d'objet	V L	V L	V L	V L
CMQORx	MQOR-Enregistrement d'objet	V L	V L	V L	V L

Tableau 2. Fichiers de copie COBOL-codes retour, constantes et structures (suite)					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX**	**Windows**	**z/OS**
CMQPMOx	MQPMO-Options de message d'insertion	V L	V L	V L	V L
CMQRFHx	MQRFH-Règles et en-tête de formatage	V L	V L	V L	V L
CMQRFH2x	MQRFH2 - En-tête 2 de règles et de formatage	V L	V L	V L	V L
CMQRMHx	MQRMH-En-tête de message de référence	V L	V L	V L	V L
CMQRRx	MQRR-Enregistrement de réponse	V L	V L	V L	
CMQSCOx	MQSCO-Options de configuration SSL		V L	V L	
CMQTMx	MQTM-Message de déclenchement	V L		V L	V L
CMQTMCx	MQTMC-Caractère du message de déclenchement	V L	V L		
CMQTMC2x	MQTMC2 -Message de déclenchement 2 caractères	V L	V L	V L	V L
CMQWIHx	MQWIH-En-tête d'informations de travail	V L	V L	V L	V L
CMQXQHx	MQXQH-En-tête de file d'attente de transmission	V L	V L	V L	V L

Clé :

- Fichiers dont les valeurs initiales sont fournies, x = V
- Fichiers sans valeurs initiales fournies, x = L

Fichiers d'inclusion PL/I

Les fichiers INCLUDE suivants sont fournis pour le langage de programmation PL/I. Ces fichiers sont disponibles sous z/OS uniquement.

Tableau 3. Fichiers d'inclusion PL/I-types de données, codes retour, constantes et structures					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX**	**Windows**	**z/OS**
Types de données, codes retour, constantes et structures					
CMQP	Définitions MQI				P
CMQCFP	Définitions PCF				P
CMQEPP	Définitions de point d'entrée				P
CMQPSP	Définitions de publication / abonnement				P
CMQXP	Définitions de canal et de sortie				P

Tableau 3. Fichiers d'inclusion PL/I-types de données, codes retour, constantes et structures (suite)					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX**	**Windows**	**z/OS**
Clé: P= Fichier fourni					

Fichiers de copie RPG

Les fichiers COPY suivants sont fournis pour le langage de programmation RPG. Ces fichiers sont disponibles uniquement sous IBM i.

Tableau 4. Fichiers de copie RPG-codes retour, constantes et structures					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX**	**Windows**	**z/OS**
Codes retour et constantes					
CMQx	Définitions MQI	G R			
CMQCFx	Définitions PCF	G			
CMQPSx	Définitions de publication / abonnement	G			
CMQXx	Définitions de canal et de sortie	G R			
structures					
CMQBOx	MQBO-Options de début	G H			
CMQCDx	MQCD-Définition de canal	G H R			
CMQCFBFx	MQCFBF-Paramètre de filtre de chaîne d'octets PCF	G H			
CMQCFBSx	MQCFBS-Paramètre de chaîne d'octets PCF	G H			
CMQCFGRx	MQCFGR-Paramètre de groupe PCF	G H			
CMQCFHx	En-tête MQCFH-PCF	G H			
CMQCFIFx	MQCFIF-Paramètre de filtre d'entier PCF	G H			
CMQCFILx	MQCFIL-Paramètre de liste d'entiers PCF	G H			
CMQCFINx	MQCFIN-Paramètre entier PCF	G H			
CMQCFSFx	MQCFSF-Paramètre de filtre de chaîne PCF	G H			
CMQCFSLx	MQCFSL-Paramètre de liste de chaînes PCF	G H			
CMQCFSTx	MQCFST-Paramètre de chaîne PCF	G H			
CMQCFXLx	MQCFIL64 -Paramètre de liste d'entiers PCF 64 bits	G H			

Tableau 4. Fichiers de copie RPG-codes retour, constantes et structures (suite)					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX**	**Windows**	**z/OS**
CMQCFXNx	MQCFIN64 -Paramètre d'entier PCF 64 bits	G H			
CMQCHARVx	MQCHARV-Chaîne de longueur variable	G H			
CMQCIHx	MQCIH-en-tête de pont CICS	G H			
CMQCNOx	MQCNO-Options de connexion	G H			
CMQCSPx	MQCSP-Paramètres de sécurité	G H			
CMQCXPx	MQCXP-Paramètres d'exit de canal	G H R			
CMQDHx	En-tête MQDH-Distribution	G H R			
CMQDLHx	En-tête MQDLH-Dead-letter	G H R			
CMQDXPx	MQDXP-Paramètres d'exit de conversion de données	G H R			
CMQEPHx	MQEPH-en-tête PCF imbriqué	G H			
CMQGMOx	MQGMO-Extraction des options de message	G H R			
CMQIIHx	MQIIH-en-tête d'informations IMS	G H R			
CMQMDx	MQMD-Descripteur de message	G H R			
CMQMD1x	MQMD1 -Descripteur de message version 1	G H R			
CMQMD2x	MQMD2 -Descripteur de message version 2	G H			
CMQMDEx	MQMDE-Descripteur de message étendu	G H R			
CMQODx	MQOD-Descripteur d'objet	G H R			
CMQORx	MQOR-Enregistrement d'objet	G H R			
CMQPMOx	MQPMO-Options de message d'insertion	G H R			
CMQPXPx	MQPXP-Paramètres d'exit de routage de publication / abonnement	G H			
CMQRFHx	MQRFH-Règles et en-tête de formatage	G H			
CMQRFH2x	MQRFH2 - En-tête 2 de règles et de formatage	G H			
CMQRMHx	MQRMH-En-tête de message de référence	G H R			

Tableau 4. Fichiers de copie RPG-codes retour, constantes et structures (suite)					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX**	**Windows**	**z/OS**
CMQRRx	MQRR-Enregistrement de réponse	G H R			
CMQTMx	MQTM-Message de déclenchement	G H R			
CMQTMCx	MQTMC-Caractère du message de déclenchement	G H R			
CMQTMC2x	MQTMC2 -Message de déclenchement 2 caractères	G H R			
CMQWIHx	MQWIH-En-tête d'informations de travail	G H			
CMQXQHx	MQXQH-En-tête de file d'attente de transmission	G H R			
Clé : • Fichier pour liaison statique, initialisé, fourni x = G • Fichier pour liaison statique, non initialisé, fourni x = H • Fichier de liaison dynamique, initialisé, fourni, x = R					

Fichiers du module Visual Basic

Des fichiers d'en-tête (ou de formulaire) sont fournis pour vous aider à écrire des programmes d'application Visual Basic qui utilisent l'interface MQI. Ces fichiers d'en-tête sont fournis dans des versions 32 bits uniquement et sont récapitulés dans le tableau suivant:

Tableau 5. Fichiers de module Visual Basic-déclarations d'appel, types de données, codes retour, constantes et structures					
Nom de fichier	**Description**	**IBM i**	**Systèmes UNIX and Linux**	**Windows**	**z/OS**
Déclarations d'appel, types de données, codes retour, constantes et structures					
CMQB	Définitions MQI			B	
CMQBB	Définitions MQAI			B	
CMQCFB	Définitions PCF			B	
CMQXB	Définitions de canal et de sortie			B	
Clé: B= Fichier fourni					

MQ_ * (longueurs de chaîne)

Tableau 6. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
LONGUEUR_CODE_ABEND_MQ_	4	X'00000004'
LONGUEUR_JETON_COMPTEUR_MQ	32	X'00000020'
MQ_APPL_IDENTITY_DATA_LENGTH	32	X'00000020'

Tableau 6. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
LONGUEUR_NOM_APPL_MQ_	28	X'0000001C'
MQ_APPL_ORIGINE_DATA_LENGTH	4	X'00000004'
LONGUEUR_BALISE_APPL_MQ_APPLICATION	28	X'0000001C'
LONGUEUR_SUFFIXE_ARM_MQ_	2	X'00000002'
LONGUEUR_ID_ATTENTION_MQ	4	X'00000004'
MQ_AUTH_INFO_NOM_CONN_LENGTH	264	X'00000108'
LONGUEUR_DESC_INFO_AUTH_MQ_MQ_AUTH_LENGTH	64	X'00000040'
LONGUEUR_NOM_INFO_AUTH_MQ_	48	X'00000030'
MQ_AUTH_INFO_OCSP_URL_LENGTH	256	X'00000100'
MQ_LONGUEUR_AUTHENTIFICATEUR	8	X'00000008'
MQ_AUTO_REORG_CATALOG_LENGTH	44	X'0000002C'
MQ_AUTO_REORG_TIME_LENGTH	4	X'00000004'
LONGUEUR_INTERFACE_ID_MQ_BATCH_	8	X'00000008'
LONGUEUR_NOM_PONT	24	X'00000018'
LONGUEUR_CODE_ANNULATION	4	X'00000004'
MQ_CF_STRUC_DESC_LENGTH	64	X'00000040'
LONGUEUR_NOM_STRUC_MQ_CF_	12	X'0000000C'
LONGUEUR_DATE_CANAL_MQ_CANAL	12	X'0000000C'
LONGUEUR_CANAL_MQ_CANAL	64	X'00000040'
LONGUEUR_NOM_CANAL_MQ_CANAL	20	X'00000014'
LONGUEUR_TEMPS_CANAL_MQ_CANAL	8	X'00000008'
LONGUEUR_PARAMÈTRE_SERVICE_MQ_CHINIT_SERVICE	32	X'00000020'
LONGUEUR_NOM_FICHIERS_MQ_CICS	8	X'00000008'
LONGUEUR_ID_CLIENT_MQ	23	X'00000017'
LONGUEUR_NOM_CLUSTER	48	X'00000030'
LONGUEUR_NOM_CONN_MQ_	264	X'00000108'
LONGUEUR_CONN_MQ_DE_CONNEXION	128	X'00000080'
LONGUEUR_ID_CONNEXION_MQM	24	X'00000018'
LONGUEUR_ID_CORREL_MQ_	24	X'00000018'
LONGUEUR_DATE_CREATION_MQ_	12	X'0000000C'
LONGUEUR_DURÉE_CREATION_MQ_	8	X'00000008'
LONGUEUR_DATE_MQ	12	X'0000000C'
LONGUEUR_NOM_DISTINCTIF	1024	X'00000400'
MQ_DNS_GROUP_NAME_LENGTH	18	X'00000012'
MQ_LONGUEUR_DONNÉES_SORTIE	32	X'00000020'
LONGUEUR_NOM_INFO_SORTIE	48	X'00000030'
LONGUEUR_NOM_SORTIE	(value differs by platform or version)	
MQ_EXIT_PD_LONGUEUR_ZONE	48	X'00000030'
MQ_EXIT_USER_LONGUEUR_ZONE	16	X'00000010'

Tableau 6. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
LONGUEUR_FACILIT_MQM	8	X'00000008'
LONGUEUR_FACILIT_MQ_LIKE_LENGTH	4	X'00000004'
LONGUEUR_FORMAT_MQ	8	X'00000008'
LONGUEUR_FONCTION_MQS	4	X'00000004'
LONGUEUR_GROUPE_MQ_ID	24	X'00000018'
LONGUEUR_MOT_DE_PASSE_LDAP_MQ_	32	X'00000020'
LONGUEUR_NOM_LISTE_MQ_ÉCOUTE	48	X'00000030'
LONGUEUR_DESC_LISTE_MQ_ÉCOUTE	64	X'00000040'
LONGUEUR_ADRESSE_LOCAL_MQ_	48	X'00000030'
MQ_LTERM_OVERRIDE_LENGTH	8	X'00000008'
LONGUEUR_NOM_MQ_LU_LENGTH	8	X'00000008'
LONGUEUR_LARGEUR_MQ_LUM	16	X'00000010'
LONGUEUR_NOM_EXIT_MQ_MAX	128	X'00000080'
MQ_MAX_MCA_LONGUEUR_ID_UTILISATEUR	64	X'00000040'
LONGUEUR_NOM_PROPRIÉTÉ_MAX_MQ_MAX	4095	X'00000FFF'
MQ_MAX_LONGUEUR_ID_UTILISATEUR	64	X'00000040'
LONGUEUR_NOM_TRAVAIL_MQ_MCA_MCA_	28	X'0000001C'
LONGUEUR_NOM_MQ_MCA_	20	X'00000014'
MQ_MCA_LONGUEUR_DONNÉES_UTILISATEUR	32	X'00000020'
MQ_MCA_LONGUEUR_ID_UTILISATEUR	(value differs by platform or version)	
MQ_MFS_MAP_NAME_LENGTH	8	X'00000008'
LONGUEUR_NOM_MODE_MQ_	8	X'00000008'
MQ_MSG_HEADER_LENGTH	4000	X'00000FA0'
LONGUEUR_MSG_MQ_ID	24	X'00000018'
LONGUEUR_JETON_MSG	16	X'00000010'
LONGUEUR_DESC_NOM_MQ_NAME	64	X'00000040'
LONGUEUR_NOM_MQ_NOM	48	X'00000030'
LONGUEUR_INSTANCE_OBJET_MQ_ID	24	X'00000018'
LONGUEUR_NOM_OBJET_MQ_	48	X'00000030'
MQ_PASS_TICKET_APPL_LENGTH	8	X'00000008'
LONGUEUR_MOT_DE_PASSE_MQM	12	X'0000000C'
LONGUEUR_ID_APP_PROCESSUS_MQM	256	X'00000100'
LONGUEUR_DES_PROCESSUS_MQ_TRAITEMENT	64	X'00000040'
MQ_PROCESS_ENV_DATA_LENGTH	128	X'00000080'
LONGUEUR_NOM_PROCESSUS	48	X'00000030'
LONGUEUR_DONNÉES_UTILISATEUR_PROCESSUS_MQ_UTILISATEUR	128	X'00000080'
LONGUEUR_NOM_PROGRAMME	20	X'00000014'
MQ_PUT_APPL_NAME_LENGTH	28	X'0000001C'

Tableau 6. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
LONGUEUR_DATE_PUT_MQ_	8	X'00000008'
LONGUEUR_TEMPS_PUT_MQ_LONG	8	X'00000008'
LONGUEUR_DESC_Q_MQ_LONG	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'
LONGUEUR_NOM_Q_MQ_	48	X'00000030'
LONGUEUR_NOM_QSG_MQ_	4	X'00000004'
LONGUEUR_SYS_DISTANT_MQ_ID	4	X'00000004'
LONGUEUR_ID_SÉCURITÉ_MQ	40	X'00000028'
LONGUEUR_SÉLECTEUR_MQM	10240	X'00002800'
LONGUEUR_ARGS_SERVICE_MQ_LONG	255	X'000000FF'
LONGUEUR_COMMANDE_SERVICE_MQ_SERVICE	255	X'000000FF'
LONGUEUR_DESC_SERVICE_MQ_SERVICE	64	X'00000040'
LONGUEUR_NOM_SERVICE_MQ_	32	X'00000020'
LONGUEUR_CHEMIN_SERVICE_MQ_LONG	255	X'000000FF'
LONGUEUR_ÉTAPE_SERVICE_MQ_SERVICE	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_LONGUEUR_MATÉRIEL	256	X'00000100'
MQ_SSL_HANDSHAKE_STAGE_LENGTH	32	X'00000020'
LONGUEUR_BIBLIOTHÈQUE_SSL_MQ_SSL_CLÉ	44	X'0000002C'
MQ_SSL_KEY_MEMBER_LENGTH	8	X'00000008'
LONGUEUR_RÉFÉRENTIEL_CLÉ_SSL_MQ_SSL	256	X'00000100'
MQ_SSL_PEER_NAME_LENGTH	1024	X'00000400'
MQ_SSL_SHORT_PEER_NAME_LENGTH	256	X'00000100'
LONGUEUR_CODE_DÉMARRAGE	4	X'00000004'
LONGUEUR_CLASSE_STOCKAGE_MQ_DESC_LENGTH	64	X'00000040'
LONGUEUR_CLASSE_STOCKAGE_MQM	8	X'00000008'
MQ_SUB_IDENTITY_LENGTH	128	X'00000080'
LONGUEUR_POINT_SOUS-FILE	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'
LONGUEUR_TEMPS_MQ_ATTENTE	8	X'00000008'
LONGUEUR_DESC_TOPIC_MQ_LONG	64	X'00000040'

Tableau 6. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
LONGUEUR_NOM_TOPIC_MQ_	48	X'00000030'
LONGUEUR_TOPIC_MQ_STR_LENGTH	10240	X'00002800'
LONGUEUR_EXIT_TOTAL_MQ_DONNÉES	999	X'000003E7'
LONGUEUR_NOM_EXIT_TOTAL_MQ_	999	X'000003E7'
LONGUEUR_NOM_MQ_TP_LENGTH	64	X'00000040'
MQ_TPIPE_NAME_LENGTH	8	X'00000008'
LONGUEUR_ID_INSTANCE_MQ_TRANS	16	X'00000010'
LONGUEUR_ID_TRANSACTION_MQM	4	X'00000004'
LONGUEUR_DONNÉES_TRIGGER_MQ_	64	X'00000040'
LONGUEUR_NOM_PROGRAMME_TRIGGER_MQ_	8	X'00000008'
MQ_TRIGGER_TERM_ID_LENGTH	4	X'00000004'
MQ_TRIGGER_TRANS_ID_LENGTH	4	X'00000004'
LONGUEUR_ID_UTILISATEUR	12	X'0000000C'
LONGUEUR_VERSION_MQQ	8	X'00000008'
LONGUEUR_NOM_GROUPE_XCF_MQ_XCF	8	X'00000008'
MQ_XCF_NOM_MEMBRE_LENGTH	16	X'00000010'

MQ_ * (Longueur de chaîne de format de commande)

Tableau 7. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQ_ARCHIVE_PFX_LENGTH	36	X'00000024'
LONGUEUR_UNITÉ_ARCHIVE_MQ_M	8	X'00000008'
MQ_ASID_LENGTH	4	X'00000004'
LONGUEUR_NOM_PROFIL_AUTH_MQ_	48	X'00000030'
LONGUEUR_ID_MQ_CF_LEID	12	X'0000000C'
LONGUEUR_MQ_COMMANDE_COMMANDE	32768	X'00008000'
LONGUEUR_NOM_JEU_DONNÉES_MQ_DONNÉES	44	X'0000002C'
MQ_DB2_NAME_LENGTH	4	X'00000004'
LONGUEUR_NOM_DSG_MQ_	8	X'00000008'
LONGUEUR_NOM_ENTITÉ	64	X'00000040'
MQ_ENV_INFO_LENGTH	96	X'00000060'
LONGUEUR_ADRESSE_IP_MQ_	48	X'00000030'
LONGUEUR_ID_CORREL_LOG_MQ_	8	X'00000008'
LONGUEUR_NOM_EXT_JOURNAL_MQ_LOG	24	X'00000018'
LONGUEUR_CHEMIN_JOURNAL_MQ	1024	X'00000400'
MQ_LRSN_LENGTH	12	X'0000000C'
LONGUEUR_NOM_ORIGINE_MQ_	8	X'00000008'
LONGUEUR_NOM_MQ_PSB	8	X'00000008'
LONGUEUR_ID_INSTRUCTION	8	X'00000008'

Tableau 7. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQ_Q_MGR_CPF_LENGTH	4	X'00000004'
LONGUEUR_RÉPONSE_MQ_ID	24	X'00000018'
MQ_RBA_LENGTH	12	X'0000000C'
LONGUEUR_PROFIL_SÉCURITÉ_MQM	40	X'00000028'
LONGUEUR_COMPOSANT_SERVICE_MQ_SERVICE	48	X'00000030'
LONGUEUR_NOM_SOUS-FILE	10240	X'00002800'
LONGUEUR_SERVICE_MQ_SYSP_SERVICE	32	X'00000020'
LONGUEUR_NOM_SYSTÈME	8	X'00000008'
NOMBRE_TÂCHE_MQ_LONGUEUR	8	X'00000008'
MQ_TPIPE_PFX_LENGTH	4	X'00000004'
LONGUEUR_ID_UOW_MQ_	256	X'00000100'
LONGUEUR_DONNÉES_UTILISATEUR_MQ	10240	X'00002800'
LONGUEUR_VOL_MQM	6	X'00000006'

MQACH_ * (structure d'en-tête de zone de chaîne d'exit d'API)

Tableau 8. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQACH_	"ACH¬"
MQACH_STRUC_ID_ARRAY	'A','C','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 9. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQACH_VERSION_1	1	X'00000001'
VERSION_MQACH_CURRENT_VERSION	1	X'00000001'
MQACH_LENGTH_1	(value differs by platform or version)	
MQACH_LONGUEUR_ACTUELLE	(value differs by platform or version)	

MQACT_ * (jeton de comptabilité)

Tableau 10. Noms et valeurs de constante	
Nom	**Valeur**
MQACT_AUCUN	X'00...00' (32 valeurs nulles)
MQACT_NON_ARRAY	'\0','\0',... (32 valeurs nulles)

MQACT_ * (Options d'action de format de commande)

Tableau 11. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQACT_FORCE_REMOVE	1	X'00000001'
MQACT_ADVANCE_LOG	2	X'00000002'
MQACT_COLLECT_STATISTICS	3	X'00000003'

Tableau 11. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQACT_PUBSUB	4	X'00000004'

MQACTP_ * (Action)

Tableau 12. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQACTP_NOUVEAU	0	X'00000000'
MQACTP_FORWARD	1	X'00000001'
MQACTP_REPLY	2	X'00000002'
RAPPORT MQACTP_RAPPORT	3	X'00000003'

MQACTT_ * (types de jeton de comptabilité)

Tableau 13. Valeurs des constantes	
Nom	**Valeur hexadécimale**
MQACTT_INCONNU	X'00'
MQACTT_CICS_LUOW_ID	X'01'
MQACTT_OS2_DEFAULT	X'04'
MQACTT_DOS_DEFAULT	X'05'
MQACTT_UNIX_NUMERIC_ID	X'06'
MQACTT_OS400_ACCOUNT_TOKEN	X'08'
MQACTT_WINDOWS_DEFAULT	X'09'
MQACTT_NT_SECURITY_ID	X'0B'
MQACTT_UTILISATEUR	X'19'

MQADOPT_ * (adoption de nouveaux contrôles MCA et adoption de nouveaux types MCA)

Adopter les nouvelles vérifications MCA

Tableau 14. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'

Adopter de nouveaux types MCA

Tableau 15. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQADOPT_TYPE_NO	0	X'00000000'
MQADOPT_TYPE_ALL	1	X'00000001'
MQADOPT_TYPE_SVR	2	X'00000002'

Tableau 15. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQADOPT_TYPE_SDR	4	X'00000004'
MQADOPT_TYPE_RCVR	8	X'00000008'
MQADOPT_TYPE_CLUSRCVR	16	X'00000010'

MQAIR_ * (Structure d'enregistrement des informations d'authentification)

Tableau 16. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQAIR_	"AIR¬"
MQAIR_STRUC_ID_ARRAY	'A','I','R','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 17. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQAIR_VERSION_1	1	X'00000001'
MQAIR_VERSION_2	2	X'00000002'
MQAIR_CURRENT_VERSION	2	X'00000002'

MQAIT_ * (type d'informations d'authentification)

Tableau 18. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQAIT_ALL	0	X'00000000'
MQAIT_CRL_LDAP	1	X'00000001'
MQAIT_OCSP	2	X'00000002'

MQAS_ * (Valeurs d'état asynchrones du format de commande)

Tableau 19. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQAS_AUCUN	0	X'00000000'
MQAS_DEMARRE	1	X'00000001'
MQAS_START_WAIT	2	X'00000002'
MQAS_ARRETEM	3	X'00000003'
MQAS_SUSPENDU	4	X'00000004'
MQAS_SUSPENDU_TEMPORAIRE	5	X'00000005'
MQAS_ACTIF	6	X'00000006'
MQAS_XX_ENCODE_CASE_ONE inactif	7	X'00000007'

MQAT_ * (types d'application d'insertion)

Tableau 20. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQAT_INCONNU	-1	X'FFFFFFFF'

Tableau 20. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQAT_NO_CONTEXT	0	X'00000000'
MQAT_CICS	1	X'00000001'
MQAT_MVS	2	X'00000002'
MQAT_OS390	2	X'00000002'
MQAT_ZOS	2	X'00000002'
MQAT_IMS	3	X'00000003'
MQAT_OS2	4	X'00000004'
MQAT_DOS	5	X'00000005'
MQAT_AIX	6	X'00000006'
MQAT_UNIX	6	X'00000006'
MQAT_QMGR	7	X'00000007'
MQAT_OS400	8	X'00000008'
MQAT_WINDOWS	9	X'00000009'
MQAT_CICS_VSE	10	X'0000000A'
MQAT_WINDOWS_NT	11	X'0000000B'
MQAT_VMS	12	X'0000000C'
MQAT_GARDIEN	13	X'0000000D'
MQAT_NSK	13	X'0000000D'
MQAT_VOS	14	X'0000000E'
MQAT_OPEN_TP1	15	X'0000000F'
MQAT_MV	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'
Utilisateur_MQAT	25	X'00000019'
MQAT_COURTIER	26	X'0000001A'
MQAT_QMGR_PUBLISH	26	X'0000001A'
MQAT_JAVA	28	X'0000001C'
MQAT_DQM	29	X'0000001D'
MQAT_CHANNEL_INITIATOR	30	X'0000001E'
MQAT_WLM	31	X'0000001F'
MQAT_BATCH	32	X'00000020'
MQAT_RRS_BATCH	33	X'00000021'
MQAT_SIB	34	X'00000022'
MQAT_PAR DEFAUT	(value differs by platform or version)	
MQAT_USER_FIRST	65536	X'00010000'
MQAT_USER_LAST	999999999	X'3B9AC9FF'

MQAUTH_ * (Valeurs des droits de format de commande)

Tableau 21. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQAUTH_NONE	0	X'00000000'
MQAUTH_ALT_USER_AUTHORITY	1	X'00000001'
MQAUTH_BROWSE	2	X'00000002'
CHANGE_MQAUTH_CHANGE	3	X'00000003'
MQAUTH_CLEAR	4	X'00000004'
MQAUTH_CONNECT	5	X'00000005'
CREER 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'
SET MQAUTH_SET	14	X'0000000E'
MQAUTH_SET_ALL_CONTEXT	15	X'0000000F'
MQAUTH_SET_IDENTITY_CONTEXT	16	X'00000010'
CONTROLE MQAUTH_DE CONTROLE	17	X'00000011'
MQAUTH_CONTRÔLE_ÉTENDU	18	X'00000012'
MQAUTH_PUBLICATION	19	X'00000013'
MQAUTH_ABONNEMENT	20	X'00000014'
MQAUTH_RESUME	21	X'00000015'
SYSTEME MQAUTH_	22	X'00000016'

MQAUTHOPT_ * (Options des droits de format de commande)

Tableau 22. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQAUTHOPT_CUMULATIVE	256	X'00000100'
MQAUTHOPT_ENTITY_EXPLICIT	1	X'00000001'
MQAUTHOPT_ENTITY_SET	2	X'00000002'
MQAUTHOPT_NOM_ALL_MATCHING	32	X'00000020'
MQAUTHOPT_NAME_AS_WILDCARD	64	X'00000040'
MQAUTHOPT_NOM_EXPLICITE	16	X'00000010'

MQAXC_ * (structure de contexte d'exit d'API)

Tableau 23. Structures de constantes

Nom	Structure
ID_STRUC_MQAXC	"AXC¬"

Tableau 23. Structures de constantes (suite)	
Nom	**Structure**
MQAXC_STRUC_ID_ARRAY	'A','X','C','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 24. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQAXC_VERSION_1	1	X'00000001'
MQAXC_CURRENT_VERSION	1	X'00000001'

MQAXP_ * (structure des paramètres d'exit d'API)

Tableau 25. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQAXP	"AXP¬"
MQAXP_STRUC_ID_ARRAY	'A','X','P','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 26. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQAXP_VERSION_1	1	X'00000001'
MQAXP_VERSION_2	2	X'00000002'
MQAXP_CURRENT_VERSION	2	X'00000002'

MQBA_ * (Sélecteurs d'attribut d'octet)

Tableau 27. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBA_FIRST	6001	X'00001771'
MQBA_LAST	8000	X'00001F40'

MQBACF_ * (types de paramètre d'octet de format de commande)

Tableau 28. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBACF_PREMIER	7001	X'00001B59'
MQBACF_EVENT_ACCOUNTING_TOKEN	7001	X'00001B59'
ID_SÉCURITÉ_ÉVÉNEMENT_MQBACF	7002	X'00001B5A'
MQBACF_RESPONSE_SET	7003	X'00001B5B'
ID_RÉPONSE_MQBACF	7004	X'00001B5C'
ID_UOW_EXTERNAL_MQBACF	7005	X'00001B5D'
ID_CONNEXION_MQBACF	7006	X'00001B5E'
ID_CONNEXION_GENERIC_MQBACF	7007	X'00001B5F'
MQBACF_ORIGIN_UOW_ID	7008	X'00001B60'
MQBACF_Q_MGR_UOW_ID	7009	X'00001B61'

Tableau 28. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
JETON_COMPTEUR_MQBACF	7010	X'00001B62'
ID_CORREL_MQBACF	7011	X'00001B63'
ID_GROUPE_MQBACF	7012	X'00001B64'
ID_MSG_MQBACF	7013	X'00001B65'
MQBACF_CF_LEID	7014	X'00001B66'
ID_CORREL_DESTINATION_MQBACF	7015	X'00001B67'
ID_SUB_MQBACF	7016	X'00001B68'
MQBACF_LAST_UTILISÉ	7016	X'00001B68'

MQBL_ * (Longueur de la mémoire tampon pour la chaîne mqAddet la chaîne mqSet)

Tableau 29. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBL_NULL_CLOS	-1	X'FFFFFFFF'

MQBMHO_ * (options et structure de la mémoire tampon pour le traitement des messages)

Structure des options de gestion de la mémoire tampon vers le message

Tableau 30. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQBMHO_	"BMHO"
MQBMHO_STRUC_ID_ARRAY	'B','M','H','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 31. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBMHO_VERSION_1	1	X'00000001'
MQBMHO_VERSION	1	X'00000001'

Options de traitement de la mémoire tampon vers le message

Tableau 32. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBMHO_AUCUN	0	X'00000000'
MQBMHO_DELETE_PROPERTIES	1	X'00000001'

MQBND_ * (liaisons par défaut)

Tableau 33. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBND_BIND_ON_OPEN	0	X'00000000'

Tableau 33. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBND_BIND_NOT_FIXED	1	X'00000001'
MQBND_BIND_ON_GROUPE	2	X'00000002'

MQBO_ * (Options de début et structure)

Structure des options de début

Tableau 34. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQBO_	"BO¬¬"
MQBO_STRUC_ID_ARRAY	'B','O','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 35. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBO_VERSION_1	1	X'00000001'
MQBO_CURRENT_VERSION	1	X'00000001'

Options de début

Tableau 36. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBO_AUCUN	0	X'00000000'

MQBT_ * (types de pont de format de commande)

Tableau 37. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQBT_OTMA	1	X'00000001'

MQCA_ * (Sélecteurs d'attribut de caractère)

Tableau 38. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'

Tableau 38. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
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'

Tableau 38. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
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'

Tableau 38. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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_ * (Types de paramètre de caractère de format de commande)

Tableau 39. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCACF_FIRST	3001	X'00000BB9'
MQCACF_NOM_Q_SOURCE	3001	X'00000BB9'
MQCACF_TO_NOM_FILE	3002	X'00000BBA'
MQCACF_NOM_PROCESSUS_SOURCE	3003	X'00000BBB'
MQCACF_TO_NOM_PROCESSUS	3004	X'00000BBC'
MQCACF_NOM_NAMELIST_SOURCE	3005	X'00000BBD'
MQCACF_TO_NOM_LISTE	3006	X'00000BBE'
MQCACF_NOM_CANAL_SOURCE	3007	X'00000BBF'
MQCACF_TO_CHANNEL_NAME	3008	X'00000BC0'
MQCACF_NOM_INFO_AUTH_SOURCE	3009	X'00000BC1'
MQCACF_TO_AUTH_INFO_NAME	3010	X'00000BC2'
MQCACF_Q_NOMS	3011	X'00000BC3'
NOMS_PROCESSUS_MQCACF	3012	X'00000BC4'
MQCACF_NAMELIST_NAMES	3013	X'00000BC5'
TEXTE_ÉCHAPPATION_MQCACF	3014	X'00000BC6'
MQCACF_LOCAL_Q_NAMES	3015	X'00000BC7'
MQCACF_MODEL_Q_NAMES	3016	X'00000BC8'
MQCACF_ALIAS_Q_NAMES	3017	X'00000BC9'
MQCACF_REMOTE_Q_NAMES	3018	X'00000BCA'
MQCACF_SENDER_CHANNEL_NAMES	3019	X'00000BCB'
MQCACF_SERVER_CHANNEL_NAMES	3020	X'00000BCC'
MQCACF_REQUESTER_CHANNEL_NAMES	3021	X'00000BCD'

Tableau 39. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCACF_NOM_CANAL_RÉCEPTEUR	3022	X'00000BCE'
MQCACF_OBJECT_Q_MGR_NAME	3023	X'00000BCF'
NOM_APPL_MQCACF	3024	X'00000BD0'
MQCACF_IDENTIFICATEUR_UTILISATEUR	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'
NOM_BRIDGE_MQCACF	3029	X'00000BD5'
NOM_FLOT_MQCACF	3030	X'00000BD6'
MQCACF_SUJET	3031	X'00000BD7'
MQCACF_PARENT_Q_MGR_NAME	3032	X'00000BD8'
ID_CORREL_MQCACF	3033	X'00000BD9'
MQCACF_HORODATAGE_PUBLICATION	3034	X'00000BDA'
DONNÉES_CHAÎNE_MQCACF	3035	X'00000BDB'
NOM_FLOT_SUPPORT_MQCACF	3036	X'00000BDC'
MQCACF_REG_TOPIC	3037	X'00000BDD'
HEURE DE REG_MQCACF	3038	X'00000BDE'
ID_UTILISATEUR_REG_MQCACF	3039	X'00000BDF'
MQCACF_CHILD_Q_MGR_NAME	3040	X'00000BE0'
MQCACF_REG_NOM_FLOT_TRAVAUX	3041	X'00000BE1'
MQCACF_REG_Q_MGR_NAME	3042	X'00000BE2'
MQCACF_REG_Q_NAME	3043	X'00000BE3'
ID_CORREL_REG_MQCACF	3044	X'00000BE4'
ID_UTILISATEUR MQCACF_EVENT_USER_ID	3045	X'00000BE5'
MQCACF_NOM_OBJET	3046	X'00000BE6'
MQCACF_EVENT_Q_MGR	3047	X'00000BE7'
MQCACF_INFO_AUTH_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_NOM_ABONNEMENT	3052	X'00000BEC'
MQCACF_REG_SOUS-NOM	3053	X'00000BED'
MQCACF_IDENTITÉ_ABONNEMENT	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'
NOM_ENSEMBLE_DONNÉES_MQCACF	3059	X'00000BF3'
DATE DE DEMARRAGE MQCACF_UOW	3060	X'00000BF4'

Tableau 39. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCACF_UOW_HEURE DE DEMARRAGE	3061	X'00000BF5'
DATE DE DEMARRAGE 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'
NOM_PROFIL_AUTH_MQCACF	3067	X'00000BFB'
NOM_ENTITÉ_MQCACF	3068	X'00000BFC'
COMPOSANT_SERVICE_MQCACF_SERVICE	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'
CHEMIN_LOG_MQCACF	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'
Portée de la commande MQCACF_COMMAND_SCOPE	3080	X'00000C08'
ID_AS_MQCACF	3081	X'00000C09'
MQCACF_PSB_NAME	3082	X'00000C0A'
ID_STP_MQCACF	3083	X'00000C0B'
NUMÉRO_TÂCHE_MQCACF	3084	X'00000C0C'
ID_TRANSACTION_MQCACF	3085	X'00000C0D'
MQCACF_Q_MGR_UOW_ID	3086	X'00000C0E'
MQCACF_NOM_ORIGINE	3088	X'00000C10'
MQCACF_ENV_INFO	3089	X'00000C11'
MQCACF_PROFIL_SÉCURITÉ_DE_SÉCURITÉ	3090	X'00000C12'
DATE_CONFIGURATION_MQCACF	3091	X'00000C13'
MQCACF_HEURE de configuration	3092	X'00000C14'
MQCACF_FROM_CF_NOM_STRUCTURE	3093	X'00000C15'
MQCACF_TO_CF_STRUC_NAME	3094	X'00000C16'
MQCACF_CF_NOMS_STRUCTURES	3095	X'00000C17'
DATE D'ECHO_MQCACF_DATE	3096	X'00000C18'
DURE_ECHEC_MQCACF_HEURE	3097	X'00000C19'
DATE_SAUVEGARDE_MQCACF	3098	X'00000C1A'
MQCACF_HEURE de sauvegarde	3099	X'00000C1B'
NOM_SYSTÈME_MQCACF	3100	X'00000C1C'
MQCACF_CF_STRUC_BACKUP_START	3101	X'00000C1D'

Tableau 39. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCACF_CF_STRUC_BACKUP_END	3102	X'00000C1E'
MQCACF_CF_STRUC_LOG_Q_MGRS	3103	X'00000C1F'
CLASSE DE STOCKAGE MQCACF_FROM_	3104	X'00000C20'
MQCACF_TO_STORAGE_CLASS	3105	X'00000C21'
MQCACF_STORAGE_NOMS_CLASSE_STOCKAGE	3106	X'00000C22'
NOM_DSG_MQCACF	3108	X'00000C24'
MQCACF_DB2_NAME	3109	X'00000C25'
MQCACF_SYSP_CMD_USER_ID	3110	X'00000C26'
GROUPE MQCACF_SYSP_OTMA_GROUP	3111	X'00000C27'
MQCACF_SYSP_OTMA_MEMBRE	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'
MQCACF_SYSP_Q_MGR_RBA	3121	X'00000C31'
MQCACF_SYSP_LOG_RBA	3122	X'00000C32'
SERVICE MQCACF_SYSP_SERVICE	3123	X'00000C33'
NOM_LISTE_MQCACF_FROM_	3124	X'00000C34'
MQCACF_TO_LISTENER_NAME	3125	X'00000C35'
MQCACF_NOM_SERVICE_SOURCE	3126	X'00000C36'
MQCACF_TO_NOM_SERVICE	3127	X'00000C37'
MQCACF_DATE_DERNIÈRE_PUT_DATE	3128	X'00000C38'
MQCACF_DERNIER_TEMPS_PUT_TIME	3129	X'00000C39'
MQCACF_LAST_GET_DATE	3130	X'00000C3A'
MQCACF_LAST_GET_TIME	3131	X'00000C3B'
DATE_OPÉRATION_MQCACF	3132	X'00000C3C'
TEMPS-opération_mqcacf	3133	X'00000C3D'
MQCACF_ACTIVITY_DESC	3134	X'00000C3E'
MQCACF_APPL_IDENTITY_DATA	3135	X'00000C3F'
MQCACF_APPL_ORIGIN_DATA	3136	X'00000C40'
DATE MQCACF_PUT_DATE	3137	X'00000C41'
MQCACF_HEURE	3138	X'00000C42'
MQCACF_RÉPONSE_À_Q	3139	X'00000C43'
MQCACF_REPLY_TO_Q_MGR	3140	X'00000C44'
MQCACF_RESOLVED_Q_NAME	3141	X'00000C45'

Tableau 39. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
ID_STRUCF_MQCACF	3142	X'00000C46'
NOM_VALEUR_MQCACF	3143	X'00000C47'
DATE MQCACF_SERVICE_START_DATE	3144	X'00000C48'
MQCACF_SERVICE_START_TIME	3145	X'00000C49'
MQCACF_SYSP_OFFLINE_RBA	3146	X'00000C4A'
MQCACF_SYSP_ARCHIVE_PFX2	3147	X'00000C4B'
MQCACF_SYSP_ARCHIVE_UNIT2	3148	X'00000C4C'
MQCACF_TO_TOPIC_NAME	3149	X'00000C4D'
MQCACF_FROM_TOPIC_NAME	3150	X'00000C4E'
MQCACF_TOPIC_NAMES	3151	X'00000C4F'
MQCACF_NOM_SOUS-SYSTÈME	3152	X'00000C50'
MQCACF_DESTINATION_Q_MGR	3153	X'00000C51'
DESTINATION_MQCACF	3154	X'00000C52'
MQCACF_SUB_USER_ID	3156	X'00000C54'
DONNEES_SOUS-UTILISATEUR_MQCACF	3159	X'00000C57'
MQCACF_SUB_SELECTOR	3160	X'00000C58'
MQCACF_LAST_PUB_DATE	3161	X'00000C59'
MQCACF_DERNIER_HEURE_PUBLICATION	3162	X'00000C5A'
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_FILTRE	3170	X'00000C62'
MQCACF_NONE	3171	X'00000C63'
MQCACF_ADMIN_TOPIC_NAMES	3172	X'00000C64'
MQCACF_LAST_USED	3172	X'00000C64'

MQCACH_ * (types de paramètre de canal de caractères de format de commande)

Tableau 40. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCACH_FIRST	3501	X'00000DAD'
MQCACH_CHANNEL_NAME	3501	X'00000DAD'
DES_MQCACH_DESC	3502	X'00000DAE'
MQCACH_MODE_NAME	3503	X'00000DAF'
MQCACH_NOM_TP	3504	X'00000DB0'
MQCACH_NOM_Q_XMIT_CACHE	3505	X'00000DB1'
MQCACH_NOM_CONNEXION	3506	X'00000DB2'
MQCACH_MCA_NAME	3507	X'00000DB3'

Tableau 40. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCACH_SEC_NOM_SORTIE	3508	X'00000DB4'
MQCACH_MSG_EXIT_NAME	3509	X'00000DB5'
MQCACH_SEND_NOM_SORTIE	3510	X'00000DB6'
MQCACH_RCV_NOM_SORTIE	3511	X'00000DB7'
MQCACH_CHANNEL_NAMES	3512	X'00000DB8'
DONNEES_UTILISATEUR_EXIT_MQCACH_SEC_	3513	X'00000DB9'
DONNEES_USER_EXIT_MQCACH_MSG_	3514	X'00000DBA'
DONNEES_UTILISATEUR_EXIT_MQCACH_SEND_	3515	X'00000DBB'
DONNEES_UTILISATEUR_EXIT_MQCACH_RCV_	3516	X'00000DBC'
ID_UTILISATEUR_MQCACHE	3517	X'00000DBD'
MOT_DE_PASSE_MQCACH_CACHE	3518	X'00000DBE'
ADRESSE_LOCAL_MQCACH_LOCAL_ADDRESS	3520	X'00000DC0'
MQCACH_NOM_LOCAL_LOCAL	3521	X'00000DC1'
MQCACH_LAST_MSG_TIME	3524	X'00000DC4'
MQCACH_LAST_MSG_DATE	3525	X'00000DC5'
ID_UTILISATEUR_MCA_MQCACH_	3527	X'00000DC7'
MQCACH_CHANNEL_START_TIME	3528	X'00000DC8'
MQCACH_CHANNEL_START_DATE	3529	X'00000DC9'
MQCACH_MCA_NOM_TRAVAIL	3530	X'00000DCA'
MQCACH_LAST_LUWID	3531	X'00000DCB'
MQCACH_CURRENT_LUWID	3532	X'00000DCC'
MQCACH_FORMAT_NAME	3533	X'00000DCD'
NOM_EXIT_MR_MQCACH_	3534	X'00000DCE'
DONNEES_UTILISATEUR_EXIT_MQCACH_MR_DONNEES_UTILISATEUR	3535	X'00000DCF'
MQCACH_SSL_CIPHER_SPEC	3544	X'00000DD8'
NOM_PE_SSL_MQCACH_	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_NOM DE DIFFUSION	3550	X'00000DDE'
MQCACH_LU_NAME	3551	X'00000DDF'
ADRESSE_IP_MQCACH_IP	3552	X'00000DE0'
NOM_TCP_MQCACHE	3553	X'00000DE1'
NOM_LISTE_MQCACHE	3554	X'00000DE2'
MQCACH_LISTENER_DESC	3555	X'00000DE3'
DATE MQCACH_LISTENER_START_DATE	3556	X'00000DE4'
HEURE-DEMARRAGE-LISTE_MQCACHE	3557	X'00000DE5'
MQCACH_SSL_KEY_RESET_DATE	3558	X'00000DE6'

Tableau 40. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCACH_SSL_KEY_RESET_TIME	3559	X'00000DE7'
MQCACH_LAST_USED	3559	X'00000DE7'

MQCADSD_ * (Descripteurs ADS d'en-tête d'informations CICS)

Tableau 41. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCADSD_NONE	0	X'00000000'
MQCADSD_SEND	1	X'00000001'
MQCADSD_RECV	16	X'00000010'
MQCADSD_MSGFORMAT	256	X'00000100'

MQCAFTY_ * (Valeurs d'affinité de connexion)

Tableau 42. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCAFTY_NONE	0	X'00000000'
MQCAFTY_PREFERENTE_PRÉFÉRÉE	1	X'00000001'

MQCAMO_ * (format de commande-Types de paramètre de surveillance des caractères)

Tableau 43. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCAMO_FIRST	2701	X'00000A8D'
DATE MQCAMO_CLOSE_DATE	2701	X'00000A8D'
HEURE MQCAMO_CLOSE	2702	X'00000A8E'
DATE MQCAMO_CONN_DATE	2703	X'00000A8F'
HEURE_CONNEXION MQCAMO_MQ	2704	X'00000A90'
DATE MQCAMO_DISC_DATE	2705	X'00000A91'
HEURE-DISC_MQCAMO_	2706	X'00000A92'
DATE MQCAMO_FIN	2707	X'00000A93'
Heure_END_MQCAMO	2708	X'00000A94'
DATE MQCAMO_OPEN_DATE	2709	X'00000A95'
HEURE DE L'OPEN_MQCAMO	2710	X'00000A96'
DATE MQCAMO_START_DATE	2711	X'00000A97'
HEURE DE DEMARRAGE MQCAMO_DEMARRAGE	2712	X'00000A98'
MQCAMO_LAST_USED	2712	X'00000A98'

MQCBC_ * (structure des constantes MQCBC)

Tableau 44. Structures de constantes	
Nom	**Structure**
MQCBC_STRUC_ID	"CBC¬"

Tableau 44. Structures de constantes (suite)	
Nom	**Structure**
MQCBC_STRUC_ID_ARRAY	'C','B','C','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 45. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCBC_VERSION_1	1	X'00000001'
MQCBC_CURRENT_VERSION	1	X'00000001'

MQCBCF_ * (indicateurs de constantes MQCBC)

Tableau 46. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCBCF_AUCUN	0	X'00000000'
MQCBCF_READA_BUFFER_EMPTY	1	X'00000001'

MQCBCT_ * (type de rappel de constantes MQCBC)

Tableau 47. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
Appel MQCBCT_START_CALL	1	X'00000001'
MQCBCT_STOP_CALL	2	X'00000002'
APPEL MQCBCT_REGISTER_CALL	3	X'00000003'
MQCBCT_DEREGISTER_CALL	4	X'00000004'
MQCBCT_EVENT_CALL	5	X'00000005'
MQCBCT_MSG_REMOVED	6	X'00000006'
MQCBCT_MSG_NOT_REMOVED	7	X'00000007'

MQCBD_ * (structure des constantes MQCBD)

Tableau 48. Structures de constantes	
Nom	**Structure**
ID MQCBD_STRUC_ID	"CBD¬"
MQCBD_STRUC_ID_ARRAY	'C','B','D','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 49. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCBD_VERSION_1	1	X'00000001'
MQCBD_CURRENT_VERSION	1	X'00000001'

MQCBDO_ * (Options de rappel des constantes MQCBD)

Tableau 50. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCBDO_AUCUN	0	X'00000000'
Appel MQCBDO_START_CALL	1	X'00000001'
Appel MQCBDO_STOP_CALL	4	X'00000004'
APPEL MQCBDO_REGISTER_CALL	256	X'00000100'
MQCBDO_DEREGISTER_CALL	512	X'00000200'
MQCBDO_ECHEC_IF_QUIESCING	8192	X'00002000'

MQCBO_ * (options de création de sac pour mqCreateBag)

Tableau 51. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCBO_AUCUN	0	X'00000000'
MQCBO_USER_BAG	0	X'00000000'
MQCBO_ADMIN_BAG	1	X'00000001'
SAC de commande mqcbo_commande	16	X'00000010'
MQCBO_SYSTEM_BAG	32	X'00000020'
MQCBO_GROUP_BAG	64	X'00000040'
MQCBO_LIST_FORM_ALLOWED	2	X'00000002'
MQCBO_LIST_FORM_INHIBÉ	0	X'00000000'
MQCBO_REORDER_AS_REQUIRED	4	X'00000004'
MQCBO_DO_NOT_REORDER	0	X'00000000'
MQCBO_CHECK_SELTEURS	8	X'00000008'
MQCBO_DO_NOT_CHECK_SELECTEURS	0	X'00000000'

MQCBT_ * (constantes MQCBD Il s'agit du type de la fonction de rappel)

Tableau 52. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCBT_MESSAGE_CONSUMER	1	X'00000001'
MQCBT_EVENT_HANDLER	2	X'00000002'

MQCC_ * (codes achèvement)

Tableau 53. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCC_OK	0	X'00000000'
MQCC_WARNING	1	X'00000001'
MQCC_FAILED	2	X'00000002'
MQCC_INCONNU	-1	X'FFFFFFFF'

MQCCSI_ * (Identificateurs de jeu de caractères codés)

Tableau 54. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCCSI_UNDEFINI	0	X'00000000'
MQCCSI_DEFAULT	0	X'00000000'
MQCCSI_Q_DIR	0	X'00000000'
MQCCSI_HÉRITER	-2	X'FFFFFFFE'
MQCCSI_IMBRIQUÉ	-1	X'FFFFFFFF'
MQCCSI_APPL	-3	X'FFFFFFFD'

MQCCT_ * (en-tête d'informations CICS-Options de tâche conversationnelle)

Tableau 55. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCCT_OUI	1	X'00000001'
MQCCT_NO	0	X'00000000'

MQCD_ * (structure de définition de canal)

Tableau 56. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCD_VERSION_1	1	X'00000001'
MQCD_VERSION_2	2	X'00000002'
MQCD_VERSION_3	3	X'00000003'
MQCD_VERSION_4	4	X'00000004'
MQCD_VERSION_5	5	X'00000005'
MQCD_VERSION_6	6	X'00000006'
MQCD_VERSION_7	7	X'00000007'
MQCD_VERSION_8	8	X'00000008'
MQCD_VERSION_9	9	X'00000009'
MQCD_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_LONGUEUR_EN_COURS	(value differs by platform or version)	

MQCDC_ * (Conversion de données de canal)

Tableau 57. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCDC_SENDER_CONVERSION	1	X'00000001'
MQCDC_NO_SENDER_CONVERSION	0	X'00000000'

MQCERT_ * (Type de règle de validation de certificat)

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_ * (indicateurs de capacité)

Tableau 58. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCF_AUCUN	0	X'00000000'
LIST_LISTES MQCF_DIST	1	X'00000001'

MQCFAC_ * (fonction d'en-tête d'informations CICS)

Tableau 59. Noms et valeurs de constante

Nom	Valeur hexadécimale
MQCFAC_NONE	X'00...00' (8 valeurs nulles)
MQCFAC_NON_ARRAY	'\0','\0',... (8 valeurs nulles)

MQCFBF_ * (structure de paramètre de filtre de chaîne d'octet de format de commande)

Tableau 60. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFBF_STRUC_LENGTH_FIXED	20	X'00000014'

MQCFBS_ * (structure des paramètres de chaîne d'octets de format de commande)

Tableau 61. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFBS_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFC_ * (options de contrôle d'en-tête de format de commande)

Tableau 62. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFC_LAST	1	X'00000001'
MQCFC_NOT_LAST	0	X'00000000'

MQCFGR_ * (Structure des paramètres de groupe de format de commande)

Tableau 63. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFGR_LONGUEUR_STRUCTURE	16	X'00000010'

MQCFH_ * (structure d'en-tête de format de commande)

Tableau 64. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
LONGUEUR_STRUC_MQCFH	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_ * (structure de paramètre de filtre d'entier de format de commande)

Tableau 65. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
LONGUEUR_STRUC_MQCFIF_LENGTH	20	X'00000014'

MQCFIL_ * (Structure des paramètres de liste d'entiers de format de commande)

Tableau 66. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFIL_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFIL64_* (structure de paramètres de liste d'entiers 64 bits de format de commande)

Tableau 67. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFIL64_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFIN_ * (Structure des paramètres de type entier de format de commande)

Tableau 68. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFIN_LONGUEUR_STRUCTURE	16	X'00000010'

MQCFIN64_* (structure de paramètres d'entier 64 bits de format de commande)

Tableau 69. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFIN64_STRUC_LENGTH	24	X'00000018'

MQCFO_ * (Options de régénération du format de commande et options de suppression de files d'attente)

Options du référentiel d'actualisation de format de commande

Tableau 70. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCFO_REFRESH_REPOSITORY_YES	1	X'00000001'
MQCFO_REFRESH_REPOSITORY_NO	0	X'00000000'

Format de commande-Options de suppression de files d'attente

Tableau 71. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCFO_REMOVE_QUEUES_YES	1	X'00000001'
MQCFO_REMOVE_QUEUES_NO	0	X'00000000'

MQCFOP_ * (Opérateurs de filtre de format de commande)

Tableau 72. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCFOP_LESS	1	X'00000001'
MQCFOP_EQUAL	2	X'00000002'
MQCFOP_SUPÉRIEUR	4	X'00000004'
MQCFOP_NOT_LESS	6	X'00000006'
MQCFOP_NOT_EQUAL	5	X'00000005'
MQCFOP_NON_SUPÉRIEUR	3	X'00000003'
MQCFOP_LIKE	18	X'00000012'
MQCFOP_NOT_LIKE	21	X'00000015'
MQCFOP_CONTIENT	10	X'0000000A'
MQCFOP_EXCLUDES	13	X'0000000D'
MQCFOP_CONTAINS_GEN	26	X'0000001A'
MQCFOP_EXCLUDES_GEN	29	X'0000001D'

MQCFR_ * (capacité de reprise de l'unité de couplage)

Tableau 73. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCFR_OUI	1	X'00000001'
MQCFR_NO	0	X'00000000'

MQCFSF_ * (Structure de paramètre de filtre de chaîne de format de commande)

Tableau 74. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCFSF_STRUC_LENGTH_FIXED	24	X'00000018'

MQCFSL_ * (Structure de paramètre de liste de chaînes de format de commande)

Tableau 75. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFSL_STRUC_LENGTH_FIXED	24	X'00000018'

MQCFST_ * (Structure de paramètre de chaîne de format de commande)

Tableau 76. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFST_STRUC_LENGTH_FIXED	20	X'00000014'

MQCFSTATUS_ * (Statut de l'unité de couplage de format de commande)

Tableau 77. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFSTATUS_NOT_FOUND	0	X'00000000'
MQCFSTATUS_ACTIVE	1	X'00000001'
MQCFSTATUS_DANS_RECUPERER	2	X'00000002'
MQCFSTATUS_DANS_LA sauvegarde	3	X'00000003'
MQCFSTATUS_ECHEC	4	X'00000004'
MQCFSTATUS_NONE	5	X'00000005'
MQCFSTATUS_INCONNU	6	X'00000006'
MQCFSTATUS_ADMIN_INCOMPLET	20	X'00000014'
MQCFSTATUS_NEVER_USED	21	X'00000015'
MQCFSTATUS_NO_BACKUP	22	X'00000016'
MQCFSTATUS_NOT_ECHEC	23	X'00000017'
MQCFSTATUS_NON_RÉCUPÉRABLE	24	X'00000018'
MQCFSTATUS_XES_ERREUR	25	X'00000019'

MQCFT_ * (types de format de commande de la structure)

Tableau 78. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCFT_AUCUN	0	X'00000000'
MQCFT_COMMAND (COMMANDE MQ)	1	X'00000001'
MQCFT_REPONSE	2	X'00000002'
MQCFT_ENTIER	3	X'00000003'
MQCFT_CHAINE	4	X'00000004'
MQCFT_INTEGER_LIST	5	X'00000005'
MQCFT_STRING_LIST	6	X'00000006'
MQCFT_EVENT	7	X'00000007'
Utilisateur_MQCF	8	X'00000008'
MQCFT_BYTE_CHAINE	9	X'00000009'

Tableau 78. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCFT_TRACE_ROUTE	10	X'0000000A'
RAPPORT MQCF	12	X'0000000C'
MQCFT_INTEGER_FILTER	13	X'0000000D'
FILTRE_CHAÎNE_MQCFT_FILTER	14	X'0000000E'
MQCFT_BYTE_STRING_FILTER	15	X'0000000F'
MQCFT_COMMAND_XR	16	X'00000010'
MQCFT_XR_MSG	17	X'00000011'
MQCFT_XR_ITEM	18	X'00000012'
MQCFT_XR_SUMMARY	19	X'00000013'
GROUPE MQCF	20	X'00000014'
STATISTIQUES MQCF	21	X'00000015'
MQCFT_ACCOUNTING	22	X'00000016'
MQCFT_INTEGER64	23	X'00000017'
MQCFT_INTEGER64_LIST	25	X'00000019'

MQCFTYPE_ * (types d'unité de couplage de format de commande)

Tableau 79. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCFTYPE_APPL	0	X'00000000'
MQCFTYPE_ADMIN	1	X'00000001'

MQCFUNC_ * (fonctions d'en-tête d'informations CICS)

Tableau 80. Structures de constantes	
Nom	**Structure**
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_NON_ARRAY	'¬','¬','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQCGWI_ * (en-tête d'information CICS-Intervalle d'attente d'obtention)

Tableau 81. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCGWI_VALEUR PAR DEFAUT	-2	X'FFFFFFFE'

MQCHAD_ * (définition automatique de canal)

Tableau 82. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCHAD_DISABLED	0	X'00000000'
MQCHAD_ENABLED	1	X'00000001'

MQCHIDS_ * (Format de commande-Etat en attente de validation)

Tableau 83. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCHIDS_NOT_INDOUBT	0	X'00000000'
MQCHIDS_INDOUBT	1	X'00000001'

MQCHLD_ * (Dispositions de canal de format de commande)

Tableau 84. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCHLD_ALL	-1	X'FFFFFFFF'
MQCHLD_VALEUR PAR DEFAUT	1	X'00000001'
MQCHLD_SHARED	2	X'00000002'
MQCHLD_PRIVE	4	X'00000004'
MQCHLD_FIXSHARED	5	X'00000005'

MQCHS_* (Statut du canal du format de commande)

Tableau 85. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_ * (options de redémarrage partagé du canal de format de commande)

Tableau 86. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCHSH_RESTART_NO	0	X'00000000'
MQCHSH_RESTART_YES	1	X'00000001'

MQCHSR_ * (Options d'arrêt du canal de format de commande)

Tableau 87. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCHSR_STOP_NOT_REQUESTED	0	X'00000000'
MQCHSR_STOP_DEMANDÉ	1	X'00000001'

MQCHSSTATE_ * (Sous-états du canal de format de commande)

Tableau 88. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCHSSTATE_AUTRES	0	X'00000000'
MQCHSSTATE_END_OF_BATCH	100	X'00000064'
MQCHSSTATE_ENVOI	200	X'000000C8'
MQCHSSTATE_RECEIVING	300	X'0000012C'
MQCHSSTATE_SÉRIALISATION	400	X'00000190'
MQCHSSTATE_RESYNCHING	500	X'000001F4'
MQCHSSTATE_PULSATION	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_DANS_MREXIT	1100	X'0000044C'
MQCHSSTATE_IN_CHADEXIT	1200	X'000004B0'
MQCHSSTATE_NET_CONNEXION	1250	X'000004E2'
MQCHSSTATE_SSL_HANDSHAKE	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_COMPRESSION	1800	X'00000708'

MQCHT_ * (types de canal)

Tableau 89. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
EXPÉDITEUR_MQCH	1	X'00000001'

Tableau 89. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
SERVEUR_MQ	2	X'00000002'
MQCHT_RECEIVER	3	X'00000003'
DEMANDE_MQCHT_DEMANDEUR	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_ * (Types de table de canal de format de commande)

Tableau 90. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCHTAB_Q_MGR	1	X'00000001'
MQCHTAB_CLNTCONN	2	X'00000002'

MQCI_ * (identificateur de corrélation)

Tableau 91. Noms et valeurs de constante	
Nom	**Valeur**
MQCI_NONE	X'00...00' (24 valeurs nulles)
MQCI_NON_ARRAY	'\0','\0',... (24 valeurs nulles)
MQCI_NOUVEAU_SESSION	X'414D5121...'
MQCI_NOUVELLE_MATRICE de sessions	'\x41','\x4D','\51','\x21',...

MQCIH_ * (indicateurs et structure d'en-tête d'informations CICS)

Structure d'en-tête d'informations CICS

Tableau 92. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQCIH	"CIH¬"
MQCIH_STRUC_ID_ARRAY	'C','I','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 93. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCIH_VERSION_1	1	X'00000001'
MQCIH_VERSION_2	2	X'00000002'
VERSION_MQCIH_CURRENT_VERSION	2	X'00000002'
MQCIH_LENGTH_1	164	X'000000A4'
MQCIH_LENGTH_2	180	X'000000B4'
MQCIH_LONGUEUR_EN_COURS	180	X'000000B4'

Indicateurs d'en-tête d'informations CICS

Tableau 94. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCIH_NONE	0	X'00000000'
EXPIRATION_MOT_DE_PASSE_MQCIH	1	X'00000001'
MQCIH_EXPIRATION non limitée	0	X'00000000'
MQCIH_REPLY_WITHOUT_NULLS	2	X'00000002'
MQCIH_RÉPONSE_AVEC_NULS	0	X'00000000'
MQCIH_SYNC_ON_RETURN	4	X'00000004'
MQCIH_NO_SYNC_ON_RETURN	0	X'00000000'

MQCLCT_ * (types de cache de cluster)

Tableau 95. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCLCT_STATIC	0	X'00000000'
MQCLCT_DYNAMIQUE	1	X'00000001'

MQCLRS_ * (format de commande-Effacement de la portée de la chaîne de rubrique)

Tableau 96. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCLRS_LOCAL	1	X'00000001'
MQCLRS_GLOBAL	2	X'00000002'

MQCLRT_ * (Format de commande-Type de chaîne de rubrique d'effacement)

Tableau 97. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCLRT_RETENU	1	X'00000001'

MQCLT_ * (types de lien d'en-tête d'informations CICS)

Tableau 98. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
PROGRAMME MQCL	1	X'00000001'
TRANSACTION MQCL	2	X'00000002'

MQCLWL_ * (charge de travail du cluster)

Tableau 99. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCLWL_USEQ_LOCAL	0	X'00000000'
MQCLWL_USEQ_ANY	1	X'00000001'
MQCLWL_USEQ_AS_Q_MGR	-3	X'FFFFFFFD'

MQCLXQ_ * (Type de file d'attente de transmission de cluster)

MQCLXQ_* sont les valeurs que vous pouvez définir dans l'attribut de gestionnaire de files d'attente DEFCLXQ . L'attribut DEFCLXQ contrôle la file d'attente de transmission qui est sélectionnée par défaut par les canaux émetteurs de cluster pour extraire les messages et les envoyer aux canaux récepteurs de cluster.

Tableau 100. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCLXQ_SCTQ	0	X'00000000'
MQCLXQ_CHANNEL	1	X'00000001'

Référence associée

«DefClusterXmitQueueType (MQLONG)», à la page 804
Attribut DefClusterXmitQueue contrôle la file d'attente de transmission qui est sélectionnée par défaut par les canaux émetteurs de cluster pour extraire les messages et les envoyer aux canaux récepteurs de cluster.

Modifier un gestionnaire de files d'attente

Consulter les gestionnaires de files d'attente

Interroger le gestionnaire de files d'attente (réponse)

«MQINQ-Attributs d'objet Inquire», à la page 691
L'appel MQINQ renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'un objet.

MQCMD_ * (codes de commande)

Tableau 101. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCMD_NONE	0	X'00000000'
MQCMD_CHANGE_Q_MGR	1	X'00000001'
MQCMD_INQUIRE_Q_MGR	2	X'00000002'
MQCMD_CHANGE_PROCESSUS	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'

Tableau 101. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
CANAL_COPIE_MQCMD_	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'
CANAL_REINITIALISATION_MQCMD_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_ECHAPPEMENT	38	X'00000026'
MQCMD_RESOLVE_CHANNEL	39	X'00000027'
MQCMD_PING_Q_MGR	40	X'00000028'
MQCMD_INQUIRE_Q_STATUS	41	X'00000029'
MQCMD_INQUIRE_CHANNEL_STATUS	42	X'0000002A'
MQCMD_CONFIG_EVENT	43	X'0000002B'
MQCMD_Q_MGR_EVENT	44	X'0000002C'
MQCMD_PERFM_EVENT	45	X'0000002D'
MQCMD_CHANNEL_EVENT	46	X'0000002E'
MQCMD_DELETE_PUBLICATION	60	X'0000003C'
MQCMD_DÉSENREGISTRE_DIFFUSEUR de publications	61	X'0000003D'
MQCMD_DÉSENREGISTRE_ABONNÉ	62	X'0000003E'
MQCMD_PUBLICATION	63	X'0000003F'
MQCMD_REGISTER_PUBLISHER	64	X'00000040'
ABONNÉ_REGISTER_MQCMD_	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'

Tableau 101. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQCMD_TRACE_ROUTE	75	X'0000004B'
SECURITE MQCMD_REFRESH_SECURITE	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'
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'
ÉVÉNEMENT_COMMANDE_MQCMD_	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'
CLASSE MQCMD_COPY_STG_CLASS	111	X'0000006F'
MQCMD_DELETE_CF_STRUC	112	X'00000070'
CLASSE MQCMD_DELETE_STG_CLASS	113	X'00000071'
MQCMD_INQUIRE_ARCHIVE	114	X'00000072'
MQCMD_INQUIRE_CF_STRUC	115	X'00000073'

Tableau 101. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
MQCMD_SET_ARCHIVE	134	X'00000086'
MQCMD_SET_LOG	136	X'00000088'
MQCMD_SET_SYSTEM	137	X'00000089'
MQCMD_START_CMD_SERVEUR	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'
SERVICE 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'

Tableau 101. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
SERVICE 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'
MQCMD_INQUIRE_TOPIC_NAMES	175	X'000000AF'
MQCMD_INQUIRE_ABONNEMENT	176	X'000000B0'
MQCMD_CREATE_ABONNEMENT	177	X'000000B1'
ABONNEMENT_CHANGE_MQCMD_	178	X'000000B2'
MQCMD_DELETE_ABONNEMENT	179	X'000000B3'
MQCMD_ABONNEMENT_COPIE	181	X'000000B5'
MQCMD_INQUIRE_SUB_STATUS	182	X'000000B6'
MQCMD_INQUIRE_TOPIC_STATUS	183	X'000000B7'
MQCMD_CLEAR_TOPIC_CHAINE	184	X'000000B8'
MQCMD_INQUIRE_PUBSUB_STATUS	185	X'000000B9'
CANAL_PURGE_MQCMD_	195	X'000000C3'

MQCMDI_ * (Valeurs d'informations de commande de format de commande)

Tableau 102. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCMDI_CMDSCOPE_ACCEPTED	1	X'00000001'
MQCMDI_CMDSCOPE_GENERATED	2	X'00000002'
MQCMDI_CMDSCOPE_COMPLETED	3	X'00000003'
MQCMDI_QSG_DISP_COMPLETED	4	X'00000004'
MQCMDI_COMMAND_ACCEPTÉ	5	X'00000005'
MQCMDI_CLUSTER_REQUEST_QUEUED	6	X'00000006'

Tableau 102. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCMDI_CHANNEL_INIT_STARTED	7	X'00000007'
MQCMDI_RECOVER_STARTED	11	X'0000000B'
MQCMDI_BACKUP_STARTED	12	X'0000000C'
MQCMDI_RECOVER_COMPLETED	13	X'0000000D'
MQCMDI_SEC_TIMER_ZERO	14	X'0000000E'
MQCMDI_REFRESH_CONFIGURATION	16	X'00000010'
MQCMDI_SEC_SIGNOFF_ERREUR	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_MAJUSCULE	21	X'00000015'
MQCMDI_SEC_MIXEDCASE	22	X'00000016'

MQCMDL_* (Niveaux de commande)

Tableau 103. Noms et valeurs de constante	
Nom	**Valeur**
MQCMDL_LEVEL_1	100
MQCMDL_LEVEL_101	101
MQCMDL_LEVEL_110	110
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

Tableau 103. Noms et valeurs de constante (suite)	
Nom	**Valeur**
MQCMDL_LEVEL_711	711
MQCMDL_LEVEL_750	750

MQCMHO_ * (Créer des options et une structure de descripteur de message)

Créer une structure d'options de descripteur de message

Tableau 104. Structures de constantes	
Nom	**Structure**
MQCMHO_STRUC_ID	"CMHO"
MQCMHO_STRUC_ID_ARRAY	'C','M','H','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 105. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCMHO_VERSION_1	1	X'00000001'
MQCMHO_CURRENT_VERSION	1	X'00000001'

Options de création de descripteur de message

Tableau 106. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCMHO_VALEUR_PAR_DÉFAUT	0	X'00000000'
MQCMHO_NO_VALIDATION	1	X'00000001'
MQCMHO_VALIDATION	2	X'00000002'
MQCMHO_AUCUN	0	X'00000000'

MQCNO_ * (options et structure de connexion)

Structure des options de connexion

Tableau 107. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQCNO	"CNO¬"
MQCNO_STRUC_ID_ARRAY	'C','N','O','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 108. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'

Tableau 108. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCNO_CURRENT_VERSION	5	X'00000005'

Options de connexion

Tableau 109. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCNO_STANDARD_BINDING	0	X'00000000'
MQCNO_FASTPATH_BINDING	1	X'00000001'
MQCNO_SERIALIZE_CONN_TAG_Q_MGR	2	X'00000002'
MQCNO_SERIALIZE_CONN_TAG_QSG	4	X'00000004'
MQCNO_RESTRICT_CONN_TAG_Q_MGR	8	X'00000008'
MQCNO_RESTRICT_CONN_TAG_QSG	16	X'00000010'
MQCNO_HANDLE_SHARE_NONE	32	X'00000020'
MQCNO_HANDLE_SHARE_BLOCK	64	X'00000040'
MQCNO_HANDLE_SHARE_NO_BLOCK	128	X'00000080'
MQCNO_LIEN_PARTAGE	256	X'00000100'
MQCNO_LIEN_ISOLÉ_LIAISON	512	X'00000200'
MQCNO_LIEN_LOCAL_LOCAL	1024	X'00000400'
LIEN_CLIENT_MQCNO_BINDING	2048	X'00000800'
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_TOUS_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'
MQCNO_RECONNECT_DISABLED	33554432	X'02000000'
MQCNO_RECONNECT_Q_MGR	67108864	X'04000000'
MQCNO_ACTIVITY_TRACE_ENABLED	134217728	X'08000000'
MQCNO_ACTIVITY_TRACE_DISABLED	268435456	X'10000000'
MQCNO_AUCUN	0	X'00000000'

MQCO_ * (options de fermeture)

Tableau 110. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCO_IMMEDIATE	0	X'00000000'
MQCO_AUCUN	0	X'00000000'
MQCO_DELETE	1	X'00000001'

Tableau 110. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCO_DELETE_PURGE	2	X'00000002'
MQCO_KEEP_SUB	4	X'00000004'
SOUS-TITRE_REMOVE_MQQUE	8	X'00000008'
MQCO_QUIESCE	32	X'00000020'

MQCODL_ * (longueur des données de sortie de l'en-tête d'informations CICS)

Tableau 111. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCODL_AS_ENTREE	-1	X'FFFFFFFF'

MQCOMPRESS_ * (compression de canal)

Tableau 112. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCOMPRESS_NON_DISPONIBLE	-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'
MQCOMPRESS_ANY	268435455	X'0FFFFFFF'

MQCONNID_ * (identificateur de connexion)

Tableau 113. Noms et valeurs de constante	
Nom	**Valeur**
MQCONNID_NONE	X'00...00' (24 valeurs nulles)
MQCONNID_NONE_ARRAY	'\0','\0',... (24 valeurs nulles)

MQCOPY_ * (Options de copie de propriété)

Tableau 114. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCOPY_NONE	0	X'00000000'
MQCOPY_ALL	1	X'00000001'
MQCOPY_FORWARD	2	X'00000002'
MQCOPY_PUBLICATION	4	X'00000004'
REPONSE_MQCOPY_RÉPONSE	8	X'00000008'
RAPPORT MQCOPY_REPORT	16	X'00000010'
MQCOPY_DEFAULT	22	X'00000016'

MQCQT_ * (types de file d'attente de cluster)

Tableau 115. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCQT_LOCAL_Q	1	X'00000001'
MQCQT_ALIAS_Q	2	X'00000002'
MQCQT_REMOTE_Q	3	X'00000003'
MQCQT_Q_MGR_ALIAS	4	X'00000004'

MQCRC_ * (codes retour d'en-tête d'informations CICS)

Tableau 116. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCRC_OK	0	X'00000000'
MQCRC_CICS_EXEC_ERROR	1	X'00000001'
MQCRC_MQ_API_ERREUR	2	X'00000002'
MQCRC_BRIDGE_ERREUR	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'
MQCRC_DÉLAI_BRIDGE_ATTENTE	8	X'00000008'
MQCRC_TRANSID_NON DISPONIBLE	9	X'00000009'

MQCS_ * (constantes MQCBC-Etat du consommateur)

Tableau 117. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCS_AUCUN	0	X'00000000'
MQCS_SUSPENDU_TEMPORAIRE	1	X'00000001'
ACTION MQCS_SUSPENDED_USER_ACTION	2	X'00000002'
MQCS_SUSPENDU	3	X'00000003'
MQCS_ARRETÉE	4	X'00000004'

MQCSC_ * (codes de démarrage d'en-tête d'informations CICS)

Tableau 118. Structures de constantes

Nom	Structure
MQCSC_DEBUT	"S¬¬¬"
Données MQCSC_STARTDATA	"SD¬¬"
PUT MQCSC_TERMINE	"TD¬¬"
MQCSC_NONE	"¬¬¬¬"
MQCSC_START_ARRAY	'S','¬','¬','¬'
MQCSC_STARTDATA_ARRAY	'S','D','¬','¬'
MQCSC_TERMINPUT_ARRAY	'T','D','¬','¬'

Tableau 118. Structures de constantes (suite)	
Nom	**Structure**
MQCSC_NONE_ARRAY	'¬','¬','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQCSP_ * (structure des paramètres de sécurité de connexion et types d'authentification)

Structure des paramètres de sécurité de connexion

Tableau 119. Structures de constantes	
Nom	**Structure**
MQCSP_STRUC_ID	"CSP¬"
MQCSP_STRUC_ID_ARRAY	'C','S','P','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 120. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCSP_VERSION_1	1	X'00000001'
MQCSP_CURRENT_VERSION	1	X'00000001'

Paramètres de sécurité de connexion Types d'authentification

Tableau 121. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQCSP_AUTH_NONE	0	X'00000000'
MQCSP_AUTH_USER_ID_AND_PWD	1	X'00000001'

MQCSRV_ * (Options du serveur de commandes)

Tableau 122. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_ * (balise de connexion du gestionnaire de files d'attente)

Tableau 123. Noms et valeurs de constante	
Nom	**Valeur**
MQCT_AUCUN	X'00...00' (128 valeurs nulles)
MQCT_NON_ARRAY	'\0','\0',... (128 valeurs nulles)

MQCTES_ * (Statut de fin de tâche de l'en-tête d'informations CICS)

Tableau 124. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCTES_NOSYNC	0	X'00000000'
MQCTES_COMMIT	256	X'00000100'
MQCTES_BACKOUT	4352	X'00001100'
MQCTES_ENDTASK	65536	X'00010000'

MQCTLO_ * (structure des options MQCTL et options de contrôle de consommateur)

Structure des options MQCTL

Tableau 125. Structures de constantes

Nom	Structure
MQCTLO_STRUC_ID	"CTLO"
MQCTLO_STRUC_ID_ARRAY	'C','T','L','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 126. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCTLO_VERSION_1	1	X'00000001'
MQCTLO_CURRENT_VERSION	1	X'00000001'

Options MQCTL-Options de contrôle du consommateur

Tableau 127. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCTLO_NONE	0	X'00000000'
MQCTLO_AFFINITÉ_UNITÉS d'exécution	1	X'00000001'
MQCTLO_FAIL_IF_QUIESCING	8192	X'00002000'

MQCUOWC_ * (en-tête d'informations CICS-Contrôles d'unité de travail)

Tableau 128. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQCUOWC_ONLY	273	X'00000111'
MQCUOWC_CONTINUE	65536	X'00010000'
MQCUOWC_FIRST	17	X'00000011'
MQCUOWC_MILIEU	16	X'00000010'
MQCUOWC_LAST	272	X'00000110'
MQCUOWC_COMMIT	256	X'00000100'
MQCUOWC_BACKOUT	4352	X'00001100'

MQCXP_ * (structure des paramètres d'exit de canal)

Tableau 129. Structures de constantes	
Nom	**Structure**
MQCXP_STRUC_ID	"CXP¬"
MQCXP_STRUC_ID_ARRAY	'C','X','P','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 130. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_ * (classe de destination)

Tableau 131. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDC_GERE	1	X'00000001'
MQDC_FOURNI	2	X'00000002'

MQDCC_ * (options de conversion, masques et facteurs)

Options de conversion

Tableau 132. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDCC_CONVERSION_PAR_DÉFAUT_PAR_DÉFAUT	1	X'00000001'
MQDCC_FILL_TARGET_BUFFER	2	X'00000002'
MQDCC_INT_DEFAULT_CONVERSION	4	X'00000004'
MQDCC_SOURCE_ENC_NATIVE	(value differs by platform or version)	
MQDCC_SOURCE_ENC_NORMAL	16	X'00000010'
MQDCC_SOURCE_ENC_INVERSE	32	X'00000020'
MQDCC_SOURCE_ENC_UNDEFINED	0	X'00000000'
MQDCC_TARGET_ENC_NATIVE	(value differs by platform or version)	
MQDCC_CIBLE_ENC_NORMAL	256	X'00000100'
MQDCC_CIBLE_INVERSÉ	512	X'00000200'
MQDCC_TARGET_ENC_UNDEFINED	0	X'00000000'

Tableau 132. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDCC_AUCUN	0	X'00000000'

Masques et facteurs des options de conversion

Tableau 133. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MASQUE_SOURCE_MQDCC_CACHE	240	X'000000F0'
MQDCC_MASQUE_CIBLE_CACHE	3840	X'00000F00'
FACTEUR_SOURCE_MQDCC_SOURCE	16	X'00000010'
MQDCC_FACTEUR_CIBLE	256	X'00000100'

MQDELO_ * (Options de suppression de publication / abonnement)

Tableau 134. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDELO_AUCUN	0	X'00000000'
MQDELO_LOCAL	4	X'00000004'

MQDH_ * (Structure d'en-tête de distribution)

Tableau 135. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQDH_	"DH¬¬"
MQDH_STRUC_ID_ARRAY	'D','H','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 136. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDH_VERSION_1	1	X'00000001'
MQDH_CURRENT_VERSION	1	X'00000001'

MQDHF_ * (indicateurs d'en-tête de distribution)

Tableau 137. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDHF_NEW_MSG_IDS	1	X'00000001'
MQDHF_NONE	0	X'00000000'

MQDISCONNECT_ * (types de déconnexion du format de commande)

Tableau 138. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDISCONNECT_NORMAL	0	X'00000000'
MQDISCONNECT_IMPLICITE	1	X'00000001'
MQDISCONNECT_Q_MGR	2	X'00000002'

MQDL_ * (Listes de distribution)

Tableau 139. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQDL_SUPPORTED	1	X'00000001'
MQDL_NOT_SUPPORTED	0	X'00000000'

MQDLH_ * (structure d'en-tête de rebut)

Tableau 140. Structures de constantes

Nom	Structure
ID_STRUC_MQDLH	"DLH¬"
MQDLH_STRUC_ID_ARRAY	'D','L','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 141. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQDLH_VERSION_1	1	X'00000001'
MQDLH_CURRENT_VERSION	1	X'00000001'

MQDLV_ * (Distribution des messages persistants / non persistants)

Tableau 142. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQDLV_AS_PARENT	0	X'00000000'
MQDLV_ALL	1	X'00000001'
MQDLV_ALL_DUR	2	X'00000002'
MQDLV_ALL_AVAIL	3	X'00000003'

MQDMHO_ * (Supprimer les options et la structure de descripteur de message)

Supprimer la structure des options de descripteur de message

Tableau 143. Structures de constantes

Nom	Structure
MQDMHO_STRUC_ID	"DMHO"
MQDMHO_STRUC_ID_ARRAY	'D','M','H','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 144. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQDMHO_VERSION_1	1	X'00000001'
MQDMHO_CURRENT_VERSION	1	X'00000001'

Options de suppression de descripteur de message

Tableau 145. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDMHO_AUCUN	0	X'00000000'

MQDMPO_ * (suppression des options et de la structure des propriétés de message)

Supprimer la structure des options de propriété de message

Tableau 146. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQDMPO_	"DMPO"
MQDMPO_STRUC_ID_ARRAY	'D','M','P','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 147. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDMPO_VERSION_1	1	X'00000001'
MQDMPO_VERSION	1	X'00000001'

Options de suppression de propriété de message

Tableau 148. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDMPO_DEL_FIRST	0	X'00000000'
MQDMPO_DEL_PROP_UNDER_CURSOR	1	X'00000001'
MQDMPO_AUCUN	0	X'00000000'

MQDNSWLM_ * (WLM DNS)

Tableau 149. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDNSWLM_NO	0	X'00000000'
MQDNSWLM_OUI	1	X'00000001'

MQDT_ * (types de destination)

Tableau 150. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDT_APPL	1	X'00000001'
COURTIER MQDT_BROKER	2	X'00000002'

MQDXP_ * (structure des paramètres d'exit de conversion)

Tableau 151. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQDXP	"DXP¬"
MQDXP_STRUC_ID_ARRAY	'D','X','P','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 152. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQDXP_VERSION_1	1	X'00000001'
MQDXP_VERSION_2	2	X'00000002'
MQDXP_CURRENT_VERSION	2	X'00000002'

MQEC_ * (Valeurs de signal)

Tableau 153. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQEC_MSG_ARRIVÉ	2	X'00000002'
MQEC_INTERVALLE_ATTENTE_EXPIRÉE	3	X'00000003'
MQEC_ATTENTE_ANNULE	4	X'00000004'
MQEC_Q_MGR_QUIESCING	5	X'00000005'
MQEC_CONNEXION_MISE au repos	6	X'00000006'

MQEI_ * (expiration)

Tableau 154. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQEI_UNLIMITED	-1	X'FFFFFFFF'

MQENC_ * (codage)

MQENC_ * (codage)

Tableau 155. Valeurs des constantes par plateforme			
Nom	**Plateforme**	**Valeur décimale**	**Valeur hexadécimale**
MQENC_NATIVE	IBM i	273	X'00000111'
	Linux	546	X'00000222'
	Linux sur SPARC	273	X'00000111'
	Linux sur x86	546	X'00000222'
	Solaris sur SPARC	273	X'00000111'
	systèmes UNIX	273	X'00000111'
	Windows	546	X'00000222'
	Micro-Focus COBOL sous Windows	17	X'00000011'
	z/OS	785	X'00000311'

MQENC_ * (masques de codage)

Tableau 156. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQENC_INTEGER_MASK	15	X'0000000F'
MQENC_DECIMAL_MASK	240	X'000000F0'
MQENC_FLOAT_MASK	3840	X'00000F00'
MASQUE_RÉSERVÉ_MQENC_MASQUE	-4096	X'FFFFF000'

MQENC_ * (Encodages pour les entiers binaires)

Tableau 157. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQENC_INTEGER_UNDEFINI	0	X'00000000'
MQENC_INTEGER_NORMAL	1	X'00000001'
MQENC_INTEGER_REVERSED	2	X'00000002'

MQENC_ * (Codages pour les entiers décimaux condensés)

Tableau 158. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQENC_DECIMAL_UNDEFINI	0	X'00000000'
MQENC_DECIMAL_NORMAL	16	X'00000010'
MQENC_DECIMAL_REVERSED	32	X'00000020'

MQENC_ * (Encodages pour les nombres à virgule flottante)

Tableau 159. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQENC_FLOAT_NON DEFINI	0	X'00000000'
MQENC_FLOAT_IEEE_NORMAL	256	X'00000100'
MQENC_FLOAT_IEEE_REVERSED	512	X'00000200'
MQENC_FLOAT_S390	768	X'00000300'
MQENC_FLOAT_TNS	1024	X'00000400'

MQEPH_ * (indicateurs et structure d'en-tête de format de commande imbriquée)

Structure d'en-tête de format de commande intégrée

Tableau 160. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQEPH_ID	"EPH¬"
MQEPH_STRUC_ID_ARRAY	'E','P','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 161. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQEPH_STRUC_LENGTH_FIXED	68	X'00000044'
MQEPH_VERSION_1	1	X'00000001'
MQEPH_CURRENT_VERSION	1	X'00000001'

Indicateurs d'en-tête de format de commande imbriquée

Tableau 162. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQEPH_NONE	0	X'00000000'
MQEPH_CCSID_IMBRIQUÉ	1	X'00000001'

MQET_ * (types d'échappement de format de commande)

Tableau 163. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQET_MQSC	1	X'00000001'

MQEVO_ * (origines d'événement de format de commande)

Tableau 164. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQEVO_AUTRES	0	X'00000000'
MQEVO_CONSOLE	1	X'00000001'
MQEVO_INIT	2	X'00000002'
MSG MQEVO_MQ	3	X'00000003'
MQEVO_MQSET	4	X'00000004'
MQEVO_INTERNE	5	X'00000005'

MQEVR_ * (format de commande-Enregistrement d'événements)

Tableau 165. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQEVR_DISABLED	0	X'00000000'
MQEVR_ENABLED	1	X'00000001'
MQEVR_EXCEPTION	2	X'00000002'
MQEVR_NO_DISPLAY	3	X'00000003'

MQEXPI_ * (Intervalle d'analyse de l'expiration)

Tableau 166. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQEXPI_OFF	0	X'00000000'

MQFB_ * (valeurs de commentaires en retour)

Tableau 167. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQFB_AUCUN	0	X'00000000'
MQFB_SYSTEM_FIRST	1	X'00000001'
MQFB_QUIT	256	X'00000100'
EXPIRATION_MQFB	258	X'00000102'
MQFB_COA	259	X'00000103'
MQFB_COD	260	X'00000104'
MQFB_CHANNEL_COMPLETED	262	X'00000106'
MQFB_CHANNEL_FAIL_RETRY	263	X'00000107'
MQFB_CHANNEL_FAIL	264	X'00000108'
MQFB_APPL_CANNOT_BE_STARTED	265	X'00000109'
MQFB_TM_ERREUR	266	X'0000010A'
MQFB_APPL_TYPE_ERREUR	267	X'0000010B'
MQFB_STOPPED_BY_MSG_EXIT	268	X'0000010C'
ACTIVITE MQFB_MQ	269	X'0000010D'
MQFB_XMIT_Q_MSG_ERROR	271	X'0000010F'
MQFB_PAN	275	X'00000113'
MQFB_NAN	276	X'00000114'
MQFB_STOPPED_BY_CHAD_EXIT	277	X'00000115'
MQFB_STOPPED_BY_PUBSUB_EXIT	279	X'00000117'
MQFB_NOT_REPOSITORY_MSG	280	X'00000118'
MQFB_BIND_OPEN_CLUSRCVR_DEL	281	X'00000119'
ACTIVITÉS MQFB_MAX_ACTIVITÉS	282	X'0000011A'
MQFB_NON_RÉACHEMINEMENT	283	X'0000011B'
MQFB_NOT_DELIVERED	284	X'0000011C'
MQFB_UNSUPPORT_FORWARDING	285	X'0000011D'
MQFB_UNSUPPORTED_DISTRIBUTION	286	X'0000011E'
MQFB_DATA_LENGTH_ZERO	291	X'00000123'
MQFB_DONNÉES_LONGUEUR_NÉGATIVE	292	X'00000124'
MQFB_DATA_LENGTH_TOO_BIG	293	X'00000125'
MQFB_BUFFER_OVERFLOW	294	X'00000126'
MQFB_LENGTH_OFF_BY_ONE	295	X'00000127'
MQFB_IIH_ERREUR	296	X'00000128'
MQFB_NOT_AUTHORIZED_FOR_IMS	298	X'0000012A'
MQFB_IMS_ERREUR	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'

Tableau 167. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQFB_CICS_BRIDGE_FAILURE	403	X'00000193'
MQFB_CICS_CORREL_ID_ERREUR	404	X'00000194'
MQFB_CICS_CCSID_ERREUR	405	X'00000195'
MQFB_CICS_ENCODING_ERROR	406	X'00000196'
MQFB_CICS_CIH_ERREUR	407	X'00000197'
MQFB_ERREUR UOW_CICS_	408	X'00000198'
MQFB_CICS_COMMAREA_ERREUR	409	X'00000199'
MQFB_CICS_APPL_NOT_STARTED	410	X'0000019A'
MQFB_CICS_APPL_ABENDED	411	X'0000019B'
MQFB_CICS_DLQ_ERREUR	412	X'0000019C'
MQFB_CICS_UOW_BACKED_OUT	413	X'0000019D'
MQFB_PUBLICATIONS_ON_REQUEST	501	X'000001F5'
MQFB_SUBSCRIBER_IS_PUBLISHER	502	X'000001F6'
MQFB_MSG_SCOPE_MISMATCH	503	X'000001F7'
MQFB_SELECTOR_MISMATCH	504	X'000001F8'
MQFB_IMS_NACK_1A_REASON_FIRST	600	X'00000258'
MQFB_IMS_NACK_1A_REASON_LAST	855	X'00000357'
MQFB_SYSTEM_LAST	65535	X'0000FFFF'
MQFB_APPL_FIRST	65536	X'00010000'
MQFB_APPL_LAST	999999999	X'3B9AC9FF'

MQFC_ * (Options de force de format de commande)

Tableau 168. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQFC_OUI	1	X'00000001'
MQFC_NO	0	X'00000000'

MQFMT_ * (formats)

Tableau 169. Noms et valeurs de constante	
Nom	**Valeur**
MQFMT_AUCUN	"¬¬¬¬¬¬¬¬"
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¬"

Tableau 169. Noms et valeurs de constante (suite)

Nom	Valeur
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','¬','¬','¬'
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','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQGA_ * (Sélecteurs d'attributs de groupe)

Tableau 170. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQGA_FIRST	8001	X'00001F41'
MQGA_LAST	9000	X'00002328'

MQGACF_ * (Types de paramètre de groupe de format de commande)

Tableau 171. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQGACF_FIRST	8001	X'00001F41'
CONTEXTE_COMMANDE_MQGACF_COMMANDE	8001	X'00001F41'
DONNÉES_COMMANDE_MQGF_ACF	8002	X'00001F42'
MQGACF_TRACE_ROUTE	8003	X'00001F43'
MQGACF_OPERATION	8004	X'00001F44'
ACTIVITÉ_MQGACF_MQG	8005	X'00001F45'
MQGACF_EMBEDDED_MQMD	8006	X'00001F46'
MQGACF_MESSAGE	8007	X'00001F47'
MQGACF_MQMD	8008	X'00001F48'
MQGACF_NOM_VALEUR_	8009	X'00001F49'
MQGACF_Q_DONNEES_COMPTE	8010	X'00001F4A'
MQGACF_Q_STATISTICS_DATA	8011	X'00001F4B'
MQGACF_CHL_STATISTICS_DATA	8012	X'00001F4C'
MQGACF_LAST_USED	8012	X'00001F4C'

MQGI_ * (identificateur de groupe)

Tableau 172. Noms et valeurs de constante	
Nom	**Valeur**
MQGI_AUCUN	X'00...00' (24 valeurs nulles)
MQGI_NON_ARRAY	'\0','\0',... (24 valeurs nulles)

MQGMO_ * (Obtenir les options et la structure du message)

Obtenir la structure des options de message

Tableau 173. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQGMO_	"GMO¬"
MQGMO_STRUC_ID_ARRAY	'G','M','O','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 174. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQGMO_VERSION_1	1	X'00000001'

Tableau 174. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQGMO_VERSION_2	2	X'00000002'
MQGMO_VERSION_3	3	X'00000003'
MQGMO_VERSION_4	4	X'00000004'
MQGMO_CURRENT_VERSION	4	X'00000004'

Options d'obtention de message

Tableau 175. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQGMO_WAIT	1	X'00000001'
MQGMO_NO_WAIT	0	X'00000000'
MQGMO_SET_SIGNAL	8	X'00000008'
MQGMO_FAIL_IF_QUIESCING	8192	X'00002000'
MQGMO_SYNCPOINT	2	X'00000002'
MQGMO_SYNCPOINT_IF_PERSISTENT	4096	X'00001000'
MQGMO_NO_SYNCPOINT	4	X'00000004'
MQGMO_MARK_SKIP_BACKOUT	128	X'00000080'
MQGMO_BROWSE_FIRST	16	X'00000010'
MQGMO_BROWSE_NEXT	32	X'00000020'
MQGMO_BROWSE_MSG_UNDER_CURSOR	2048	X'00000800'
MQGMO_BROWSE_HANDLE	17825808	X'01100010'
MQGMO_BROWSE_CO_OP	18874384	X'01200010'
MQGMO_MSG_UNDER_CURSOR	256	X'00000100'
MQGMO_LOCK	512	X'00000200'
MQGMO_UNLOCK	1024	X'00000400'
MQGMO_ACCEPT_TRUNCATED_MSG	64	X'00000040'
MQGMO_CONVERT	16384	X'00004000'
MQGMO_LOGICAL_ORDER	32768	X'00008000'
MQGMO_COMPLETE_MSG	65536	X'00010000'
MQGMO_ALL_MSGS_AVAILABLE	131072	X'00020000'
MQGMO_ALL_SEGMENTS_AVAILABLE	262144	X'00040000'
MQGMO_MARK_BROWSE_HANDLE	1048576	X'00100000'
MQGMO_MARK_BROWSE_CO_OP	2097152	X'00200000'
MQGMO_UNMARK_BROWSE_CO_OP	4194304	X'00400000'
MQGMO_UNMARK_BROWSE_HANDLE	8388608	X'00800000'
MQGMO_UNMARKED_BROWSE_MSG	16777216	X'01000000'
MQGMO_PROPERTIES_FORCE_MQRFH2	33554432	X'02000000'
MQGMO_NO_PROPERTIES	67108864	X'04000000'
MQGMO_PROPERTIES_IN_HANDLE	134217728	X'08000000'
MQGMO_PROPERTIES_COMPATIBILITY	268435456	X'10000000'

Tableau 175. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQGMO_PROPERTIES_AS_Q_DEF	0	X'00000000'
MQGMO_NONE	0	X'00000000'

MQGS_ * (statut du groupe)

Tableau 176. Noms et valeurs de constante	
Nom	**Valeur**
MQGS_NOT_IN_GROUPE	'¬'
MQGS_MSG_IN_GROUPE	'G'
MQGS_LAST_MSG_IN_GROUP	'L'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQHA_ * (Sélecteurs de descripteur)

Tableau 177. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQHA_FIRST	4001	X'00000FA1'
MQHA_BAG_HANDLE	4001	X'00000FA1'
MQHA_LAST_USED	4001	X'00000FA1'
MQHA_LAST	6000	X'00001770'

MQHB_ * (descripteurs de sac)

Tableau 178. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQHB_UNUSABLE_HBAG	-1	X'FFFFFFFF'
MQHB_AUCUN	-2	X'FFFFFFFE'

MQHC_ * (descripteurs de connexion)

Tableau 179. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQHC_DEF_HCONN	0	X'00000000'
MQHC_UNUSABLE_HCONN	-1	X'FFFFFFFF'
MQHC_UNASSOCIATED_HCONN	-3	X'FFFFFFFD'

MQHM_ * (descripteur de message)

Tableau 180. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQHM_UNUSABLE_HMSG	-1	X'FFFFFFFF'
MQHM_NONE	0	X'00000000'

MQHO_ * (Descripteur d'objet)

Tableau 181. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQHO_UNUSABLE_HOBJ	-1	X'FFFFFFFF'
MQHO_AUCUN	0	X'00000000'

MQHSTATE_ * (Etats de descripteur de format de commande)

Tableau 182. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQHSTATE_INACTIVE	0	X'00000000'
MQHSTATE_ACTIVE	1	X'00000001'

MQIA_ * (Sélecteurs d'attribut d'entier)

Tableau 183. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'
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'

Tableau 183. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
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'

Tableau 183. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
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'

Tableau 183. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
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'

Tableau 183. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
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'

Tableau 183. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_ * (format de commande-Types de paramètre entier)

Tableau 184. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIACF_FIRST	1001	X'000003E9'
MQIACF_Q_MGR_ATTRS	1001	X'000003E9'
MQIACF_Q_ATTRS	1002	X'000003EA'
MQIACF_PROCESS_ATTRS	1003	X'000003EB'
MQIACF_NAMELIST_ATTRS	1004	X'000003EC'
MQIACF_FORCE	1005	X'000003ED'
REMPLACEMENT_MQIACF_ACF	1006	X'000003EE'
PURGE_MQIACF_ACF	1007	X'000003EF'
MQIACF_QUIESCE	1008	X'000003F0'
MODE MQIACF	1008	X'000003F0'
MQIACF_ALL	1009	X'000003F1'
MQIACF_EVENT_APPL_TYPE	1010	X'000003F2'
ORIGINE_ÉVÉNEMENT_MQIACF_ACF	1011	X'000003F3'
ID_PARAMÈTRE_MQIACF_ID	1012	X'000003F4'
ID_ERREUR_MQIACF	1013	X'000003F5'
IDENTIFICATEUR_ERREUR_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1013	X'000003F5'
MQIACF_SELECTOR	1014	X'000003F6'
MQIACF_ATTRIBUTS de canal	1015	X'000003F7'
TYPE_OBS_MQIACF	1016	X'000003F8'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIACF_TYPE_ÉCHAPPEMENT	1017	X'000003F9'
MQIACF_DÉCALAGE_ERREUR_ERREUR	1018	X'000003FA'
MQIACF_AUTH_INFO_ATTRS	1019	X'000003FB'
QUALIFICATION_RAISON_MQIACF_ANOMALIE	1020	X'000003FC'
Commande MQIACF_COMMAND	1021	X'000003FD'
OPTIONS_OPEN_MQIACF_ACF	1022	X'000003FE'
MQIACF_OPEN_TYPE	1023	X'000003FF'
ID_PROCESSUS_MQIACF_ID	1024	X'00000400'
ID_UNITÉ_UNITÉ_FILE_MQIACF	1025	X'00000401'
MQIACF_Q_STATUS_ATTRS	1026	X'00000402'
MQIACF_UNCOMMITTED_MSGS	1027	X'00000403'
ETAT_HANDLE_MQIACF	1028	X'00000404'
MQIACF_AUX_ERROR_DATA_INT_1	1070	X'0000042E'
MQIACF_AUX_ERROR_DATA_INT_2	1071	X'0000042F'
MQIACF_CONV_CODE_RAISON	1072	X'00000430'
TYPE_BRIDGE_MQIACF_	1073	X'00000431'
MQIACF_INTERROGATION	1074	X'00000432'
INTERVALLE_ATTENTE_MQIACF_	1075	X'00000433'
OPTIONS MQIACF	1076	X'00000434'
MQIACF_BROKER_OPTIONS	1077	X'00000435'
MQIACF_REFRESH_TYPE	1078	X'00000436'
NUMÉRO_SÉQUENCE_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1079	X'00000437'
Données_INTEGER_MQIACF_	1080	X'00000438'
MQIACF_OPTIONS d'enregistrement	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'
ACTION MQIACF	1086	X'0000043E'
MQIACF_INTERRUPTION	1087	X'0000043F'
MQIACF_NOMBRE_COURTIERS	1088	X'00000440'
NOMBRE_APPL_MQIACF_NOMBRE	1089	X'00000441'
NOMBRE_ANONYMISES_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1090	X'00000442'
OPTIONS REG_REG_MQIACF	1091	X'00000443'
OPTIONS DE DELETE_MQIACF	1092	X'00000444'
MQIACF_CLUSTER_Q_MGR_ATTRS	1093	X'00000445'
INTERVALLE_ACTUALISATION_MQIACF_	1094	X'00000446'
MQIACF_REFRESH_REPOSITORY	1095	X'00000447'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIACF_FILES d'attente de suppression	1096	X'00000448'
MQIACF_TYPE_ENTRÉE_OUVERTE	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_DESCRIPTEUR	1104	X'00000450'
STATUT Q_MQIACF	1105	X'00000451'
TYPE_SÉCURITÉ_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	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_PROFIL_ATTRS	1114	X'0000045A'
LISTE_AUTORISATION_MQIACF_AUTORISATION	1115	X'0000045B'
MQIACF_AUTH_ADD_AUTHS	1116	X'0000045C'
MQIACF_AUTH_REMOVE_AUTHS	1117	X'0000045D'
TY_ENTIER_MQIACF_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'
ETAT_UOW_MQIACF	1128	X'00000468'
MQIACF_SECURITY_ITEM	1129	X'00000469'
MQIACF_CF_STATUT_STRUCTURE	1130	X'0000046A'
TYPE_UOW_MQIACF_MQ	1132	X'0000046C'
MQIACF_CF_STRUC_ATTRS	1133	X'0000046D'
MQIACF_INTERVALDE_EXCLUSION	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_TYPE_STRUCTURE	1139	X'00000473'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
DÉLAI_SÉCURITÉ_MQIACF_MQI	1152	X'00000480'
INTERVALLE_SÉCURITÉ_MQIACF_MQI	1153	X'00000481'
MQIACF_SECURITY_SWITCH	1154	X'00000482'
PARAMÈTRE_SÉCURITÉ_MQI	1155	X'00000483'
MQIACF_STORAGE_CLASS_ATTRS	1156	X'00000484'
TYPE MQIACF_USAGE_	1157	X'00000485'
ID_POOL_BUFFER_MQIACF_	1158	X'00000486'
MQIACF_USAGE_TOTAL_PAGES	1159	X'00000487'
PAGES_UTILISATION_MQIACF_UNUSAGE_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_NOMBRE_EXTENSIF_UTILISATION	1164	X'0000048C'
STATUT DE LA PAGE 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_POOL_BUFFER_USAGE_MQIACF_	1170	X'00000492'
MQIACF_MOVE_COUNT	1171	X'00000493'
MQIACF_EXPIRY_Q_COUNT	1172	X'00000494'
Objets de configuration MQIACF_CONFIGURATION_OBJECTS	1173	X'00000495'
MQIACF_CONFIGURATION_EVENTS	1174	X'00000496'
TYPE_SYSP_MQIACF	1175	X'00000497'
MQIACF_SYSP_DEALLOC_INTERVAL	1176	X'00000498'
MQIACF_SYSP_MAX_ARCHIVE	1177	X'00000499'
MQIACF_SYSP_MAX_READ_TAPES	1178	X'0000049A'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIACF_SYSP_IN_BUFFER_SIZE	1179	X'0000049B'
MQIACF_SYSP_OUT_BUFFER_SIZE	1180	X'0000049C'
MQIACF_SYSP_OUT_BUFFER_COUNT	1181	X'0000049D'
MQIACF_SYSP_ARCHIVE	1182	X'0000049E'
MQIACF_SYSP_DUAL_ACTIVE	1183	X'0000049F'
MQIACF_SYSP_DUAL_ARCHIVE	1184	X'000004A0'
MQIACF_SYSP_DUAL_BSDS	1185	X'000004A1'
MQIACF_SYSP_MAX_CONNS	1186	X'000004A2'
MQIACF_SYSP_MAX_CONNS_AVANT	1187	X'000004A3'
MQIACF_SYSP_MAX_CONNS_BACK	1188	X'000004A4'
INTERVALLE_EXIT_SYSP_MQIACF_XX_ENCODE_CASE_CAPS_LOCK _OFF	1189	X'000004A5'
MQIACF_SYSP_EXIT_TASKS	1190	X'000004A6'
MQIACF_SYSP_CHKPOINT_COUNT	1191	X'000004A7'
INTERVALLE_OTMA_SYSP_MQIACF_	1192	X'000004A8'
MQIACF_SYSP_Q_INDEX_DIFFÉRER	1193	X'000004A9'
MQIACF_SYSP_DB2_TASKS	1194	X'000004AA'
MQIACF_SYSP_RESLEVEL_AUDIT	1195	X'000004AB'
CODE MQIACF_SYSP_ROUTING_CODE	1196	X'000004AC'
MQIACF_SYSP_SMF_COMPTABILITÉ	1197	X'000004AD'
MQIACF_SYSP_SMF_STATS	1198	X'000004AE'
MQIACF_SYSP_SMF_INTERVAL	1199	X'000004AF'
MQIACF_SYSP_TRACE_CLASS	1200	X'000004B0'
TAILLE DU FICHIER MQIACF_SYSP_TRACE_SIZE	1201	X'000004B1'
MQIACF_SYSP_INTERVALLE_WLM	1202	X'000004B2'
MQIACF_SYSP_ALLOC_UNIT	1203	X'000004B3'
MQIACF_SYSP_ARCHIVE_RETAIN	1204	X'000004B4'
MQIACF_SYSP_ARCHIVE_WTOR	1205	X'000004B5'
TAILLE_BLOC_SYSP_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OFF	1206	X'000004B6'
MQIACF_SYSP_CATALOGUE	1207	X'000004B7'
MQIACF_SYSP_COMPACT	1208	X'000004B8'
MQIACF_SYSP_ALLOC_PRIMARY	1209	X'000004B9'
MQIACF_SYSP_ALLOC_SECONDARY	1210	X'000004BA'
MQIACF_SYSP_PROTECT	1211	X'000004BB'
MQIACF_SYSP_QUIESCE_INTERVAL	1212	X'000004BC'
HORODATAGE_SYSP_MQIACF_XX_ENCODE_CASE_CAPS_LOCK_OF F	1213	X'000004BD'
ADRESSE_UNIT_SYSP_MQIACF	1214	X'000004BE'
STATUT DU SYSTEME MQIACF_SYSP_UNIT_STATUS	1215	X'000004BF'
MQIACF_SYSP_LOG_COPY	1216	X'000004C0'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIACF_SYSP_LOG_UTILISÉ	1217	X'000004C1'
MQIACF_SYSP_LOG_SUSPEND	1218	X'000004C2'
MQIACF_SYSP_OFFLOAD_STATUS	1219	X'000004C3'
MQIACF_SYSP_TOTAL_LOGS	1220	X'000004C4'
MQIACF_SYSP_FULL_LOGS	1221	X'000004C5'
MQIACF_LISTENER_ATTRS	1222	X'000004C6'
MQIACF_LISTENER_STATUS_ATTRS	1223	X'000004C7'
MQIACF_SERVICE_ATTRS	1224	X'000004C8'
MQIACF_SERVICE_STATUS_ATTRS	1225	X'000004C9'
INDICATEUR_HEURE_Q_MQIACF_FILE	1226	X'000004CA'
MQIACF_OLDEST_MSG_AGE	1227	X'000004CB'
OPTIONS D'AUTEUR MQIACF	1228	X'000004CC'
MQIACF_Q_MGR_STATUS_ATTRS	1229	X'000004CD'
MQIACF_CONNECTION_COUNT	1230	X'000004CE'
MQIACF_Q_MGR_FACILITY	1231	X'000004CF'
MQIACF_CHINIT_STATUS	1232	X'000004D0'
MQIACF_CMD_SERVER_STATUS	1233	X'000004D1'
DÉTAILLEMENT_ROUTE_MQIACF_ACF	1234	X'000004D2'
ACTIVITÉS MQIACF_RECORDED_ACTIVITÉS	1235	X'000004D3'
ACTIVITÉS MQIACF_MAX_ACTIVITÉS	1236	X'000004D4'
MQIACF_NOMBRE_DISCONTINUITÉS	1237	X'000004D5'
ACCUMULATION_ROUTE_MQIACF_XX_ENCODE_CASE_CAPS_LOCK _OFF	1238	X'000004D6'
MQIACF_ROUTE_DISTRIBUTION	1239	X'000004D7'
TYPE_OPÉRATION_MQIACF_OPÉRATION	1240	X'000004D8'
MQIACF_BACKOUT_COUNT	1241	X'000004D9'
CODE COMP_MQIACF	1242	X'000004DA'
CODAGE_MQIACF_CODAGE	1243	X'000004DB'
EXPIRATION_MQIACF_EXPIRATION	1244	X'000004DC'
MQIACF_FEEDBACK	1245	X'000004DD'
MQIACF_MSG_FLAGS	1247	X'000004DF'
LONGUE_MSG_MQIACF_MESSAGE	1248	X'000004E0'
TYPE MQIACF_MSG_	1249	X'000004E1'
DÉCALAGE_MQIACF_ACF	1250	X'000004E2'
MQIACF_ORIGINAL_LENGTH	1251	X'000004E3'
PERSISTENCE MQIACF_PERSISTANCE	1252	X'000004E4'
PRIO_MQIACF_ACF	1253	X'000004E5'
MQIACF_CODE_RAISON	1254	X'000004E6'
RAPPORT MQIACF_RAPPORT	1255	X'000004E7'
VERSION MQIACF_MQ	1256	X'000004E8'

Tableau 184. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
ACTIVITÉS MQIACF_UNRECORDED_ACTIVITÉS	1257	X'000004E9'
MQIACF_SURVEILLANCE	1258	X'000004EA'
RÉACHEMINEMENT_ROUTE_MQIACF_ACF	1259	X'000004EB'
STATUT DU SERVICE MQIACF_STATUT	1260	X'000004EC'
MQIACF_Q_TYPES	1261	X'000004ED'
MQIACF_XX_ENCODE_CASE_ONE id_utilisateur_support	1262	X'000004EE'
VERSION INTERFACF_MQIACF	1263	X'000004EF'
MQIACF_AUTH_SERVICE_ATTRS	1264	X'000004F0'
TYPE_EXTENSIF_MQIACF_USAGE_	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'
CLASSE_DESTINATION_MQIACF_ACF	1273	X'000004F9'
MQIACF_DURABLE_ABONNEMENT	1274	X'000004FA'
MQIACF_SUBSCRIPTION_SCOPE	1275	X'000004FB'
ID_UTILISATEUR MQIACF_VARIABLE_	1277	X'000004FD'
MQIACF_REQUEST_ONLY	1280	X'00000500'
MQIACF_PRIORITE de publication	1283	X'00000503'
MQIACF_SUB_ATTRS	1287	X'00000507'
SCHÉMA_GÉNÉRIQUES-MQIACF_WILDCARD_SCHEMA	1288	X'00000508'
SOUS-TYPE MQIACF_	1289	X'00000509'
NOMBRE_MESSAGE_MQIACF_MESSAGE	1290	X'0000050A'
MQIACF_Q_MGR_PUBSUB	1291	X'0000050B'
MQIACF_Q_MGR_VERSION	1292	X'0000050C'
MQIACF_SUB_STATUS_ATTRS	1294	X'0000050E'
STATUT DE LA COMMANDE MQIACF_TOPIC_STATUS	1295	X'0000050F'
MQIACF_TOPIC_SUB	1296	X'00000510'
MQIACF_TOPIC_PUB	1297	X'00000511'
MQIACF_RETAINED_PUBLICATION	1300	X'00000514'
MQIACF_TOPIC_STATUS_ATTRS	1301	X'00000515'
MQIACF_TOPIC_STATUS_TYPE	1302	X'00000516'
MQIACF_SUB_OPTIONS	1303	X'00000517'
NOMBRE_DIFF_MQIACF_PUBLICATION	1304	X'00000518'
MQIACF_CLEAR_TYPE	1305	X'00000519'
Portée MQIACF_CLEAR_SCOPE	1306	X'0000051A'
MQIACF_SOUS-NIVEAU	1307	X'0000051B'
MQIACF_ASYNC_STATE	1308	X'0000051C'

Tableau 184. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIACF_SYNTHéSE	1309	X'0000051D'
MSGS_OBSOLÈTES_MQIACF_	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_INDICATEUR	1351	X'00000547'
MQIACF_LAST_USED	1351	X'00000547'

MQIACH_ * (types de canal de type entier au format de commande)

Tableau 185. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'
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'

Tableau 185. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
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'

Tableau 185. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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'
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'

Tableau 185. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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_ * (format de commande-Types de paramètre de surveillance des entiers)

Tableau 186. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIAMO_FIRST	701	X'000002BD'
TAILLE DE LA COMMANDE MQIAMO_AVG_BATCH_SIZE	702	X'000002BE'
MQIAMO_AVG_Q_HEURE	703	X'000002BF'
MQIAMO_ANNULATIONS	704	X'000002C0'
MQIAMO_NAVIGATIONS	705	X'000002C1'
MQIAMO_BROWSE_MAX_BYTES	706	X'000002C2'
MQIAMO_BROWSE_MIN_BYTES	707	X'000002C3'
Echec de MQIAMO_BROWSES_FAILED	708	X'000002C4'
MQIAMO_FERMETURES	709	X'000002C5'
MQIAMO_VALIDATIONS	710	X'000002C6'
Echec de MQIAMO_COMMITS_FAILED	711	X'000002C7'
MQIAMO_CONNS	712	X'000002C8'

Tableau 186. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIAMO_CONNS_MAX	713	X'000002C9'
MQIAMO_DISQUES	714	X'000002CA'
MQIAMO_DISCS_IMPLICITE	715	X'000002CB'
TYPE_DISC_MQIAMO	716	X'000002CC'
MQIAMO_EXIT_TIME_AVG	717	X'000002CD'
MQIAMO_EXIT_TIME_MAX	718	X'000002CE'
MQIAMO_EXIT_TEMPS_MIN	719	X'000002CF'
MQIAMO_LOTS	720	X'000002D0'
MQIAMO_MSGS	721	X'000002D1'
MQIAMO_GETS	722	X'000002D2'
MQIAMO_GET_MAX_BYTES	723	X'000002D3'
MQIAMO_GET_MIN_BYTES	724	X'000002D4'
Echec de MQIAMO_GETS_FAILED	725	X'000002D5'
MQIAMO_INCOMPLETE_LOTS	726	X'000002D6'
MQIAMO_INQS	727	X'000002D7'
MQIAMO_MSGS	728	X'000002D8'
MQIAMO_NET_TIME_AVG	729	X'000002D9'
MQIAMO_TEMPS_NET_MAX	730	X'000002DA'
MQIAMO_HEURE_NET_MIN	731	X'000002DB'
NOMBRE_OBJETS_MQIAMO_OBJET	732	X'000002DC'
MQIAMO_OUVERTURES	733	X'000002DD'
MQIAMO_PUT1S	734	X'000002DE'
MQIAMO_INSERTIONS	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_PROFONDEUR	739	X'000002E3'
MQIAMO_Q_MIN_DEPTH	740	X'000002E4'
MQIAMO_Q_TIME_MOY	741	X'000002E5'
MQIAMO_Q_TIME_MAX	742	X'000002E6'
MQIAMO_HEURE_Q_MIN	743	X'000002E7'
MQIAMO_SETS	744	X'000002E8'
Echec de MQIAMO_CONNS_FAILED	749	X'000002ED'
Echec de MQIAMO_OPENS_FAILED	751	X'000002EF'
Echec de MQIAMO_INQS_FAILED	752	X'000002F0'
MQIAMO_SETS_FAILED	753	X'000002F1'
MQIAMO_PUTS_FAILED	754	X'000002F2'
MQIAMO_PUT1S_FAILED	755	X'000002F3'
Echec de MQIAMO_CLOSES_FAILED	757	X'000002F5'

Tableau 186. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIAMO_MSGS_EXPIRED	758	X'000002F6'
MQIAMO_MSGS_NOT_QUEUED	759	X'000002F7'
MQIAMO_MSGS_PURGED	760	X'000002F8'
MQIAMO_SUBS_DUR	764	X'000002FC'
MQIAMO_SUBS_NDUR	765	X'000002FD'
Echec de MQIAMO_SUBS_FAILED	766	X'000002FE'
MQIAMO_SUBRQS	767	X'000002FF'
Echec de MQIAMO_SUBRQS_FAILED	768	X'00000300'
MQIAMO_CBS	769	X'00000301'
MQIAMO_CBS_ECHEC	770	X'00000302'
MQIAMO_CTLS	771	X'00000303'
MQIAMO_CTLS_FAILED	772	X'00000304'
MQIAMO_STATS	773	X'00000305'
MQIAMO_STATS_ECHEC	774	X'00000306'
MQIAMO_SUB_DUR_HIGHWATER	775	X'00000307'
MQIAMO_SUB_DUR_LOWWATER	776	X'00000308'
MQIAMO_SUB_NDUR_HIGHWATER	777	X'00000309'
MQIAMO_SUB_NDUR_LOWWATER	778	X'0000030A'
MQIAMO_TOPIC_PUTS	779	X'0000030B'
MQIAMO_TOPIC_PUTS_FAILED	780	X'0000030C'
MQIAMO_TOPIC_PUT1S	781	X'0000030D'
MQIAMO_TOPIC_PUT1S_FAILED	782	X'0000030E'
MQIAMO_NOMBRE_MSG_PUBLICATION	784	X'00000310'
MQIAMO_UNSUBS_DUR	786	X'00000312'
MQIAMO_UNSUBS_NDUR	787	X'00000313'
MQIAMO_UNSUBS_FAILED	788	X'00000314'
INTERVALLE_MQIAMO_	789	X'00000315'
MQIAMO_MSGS_SENT	790	X'00000316'
MQIAMO_BYTES_SENT	791	X'00000317'
MQIAMO_REPAIR_BYTES	792	X'00000318'
MODE_PASSE_ALIMENTATION_MQIAMO_MODE	793	X'00000319'
MQIAMO_TYPE_FIABILITÉ	794	X'0000031A'
MQIAMO_LATE_JOIN_MARK	795	X'0000031B'
MQIAMO_NACKS_RCVD	796	X'0000031C'
MQIAMO_REPAIR_PKTS	797	X'0000031D'
MQIAMO_HISTORY_PKTS	798	X'0000031E'
MQIAMO_PENDING_PKTS	799	X'0000031F'
MQIAMO_PKT_RATE	800	X'00000320'
MQIAMO_MCAST_XMIT_RATE	801	X'00000321'

Tableau 186. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQIAMO_MCAST_BATCH_HEURE	802	X'00000322'
MQIAMO_MCAST_HEARTBEAT	803	X'00000323'
PORT MQIAMO_DEST_DATA_PORT	804	X'00000324'
PORT MQIAMO_DEST_REPAIR_PORT	805	X'00000325'
MQIAMO_ACKS_RCVD	806	X'00000326'
MQIAMO_ACTIVE_ACKERS	807	X'00000327'
MQIAMO_PKTS_SENT	808	X'00000328'
MQIAMO_TOTAL_REPAIR_PKTS	809	X'00000329'
MQIAMO_TOTAL_PKTS_SENT	810	X'0000032A'
MQIAMO_TOTAL_MSGS_SENT	811	X'0000032B'
MQIAMO_TOTAL_BYTES_SENT	812	X'0000032C'
MQIAMO_NUM_STREAMS	813	X'0000032D'
MQIAMO_ACK_FEEDBACK	814	X'0000032E'
MQIAMO_NACK_FEEDBACK	815	X'0000032F'
MQIAMO_PKTS_LOST	816	X'00000330'
MQIAMO_MSGS_RCVD	817	X'00000331'
MQIAMO_MSG_BYTES_RCVD	818	X'00000332'
MQIAMO_MSGS_DELIVERED	819	X'00000333'
MQIAMO_PKTS_PROCESSED	820	X'00000334'
MQIAMO_PKTS_DLVD	821	X'00000335'
MQIAMO_PKTS_SUPPRIMÉ	822	X'00000336'
MQIAMO_PKTS_DUPLIQUÉ	823	X'00000337'
MQIAMO_NACKS_CREATED	824	X'00000338'
MQIAMO_NACK_PKTS_SENT	825	X'00000339'
MQIAMO_REPAIR_PKTS_RQSTD	826	X'0000033A'
MQIAMO_REPAIR_PKTS_RCVD	827	X'0000033B'
MQIAMO_PKTS_RÉPARÉ	828	X'0000033C'
MQIAMO_TOTAL_MSGS_RCVD	829	X'0000033D'
MQIAMO_TOTAL_MSGS_BYTES_RCVD	830	X'0000033E'
MQIAMO_TOTAL_REPAIR_PKTS_RCVD	831	X'0000033F'
MQIAMO_TOTAL_REPAIR_PKTS_RQSTD	832	X'00000340'
MQIAMO_TOTAL_MSGS_PROCESSED	833	X'00000341'
MQIAMO_TOTAL_MSGS_SELECTED	834	X'00000342'
MQIAMO_TOTAL_MSGS_EXPIRED	835	X'00000343'
MQIAMO_TOTAL_MSGS_DELIVERED	836	X'00000344'
MQIAMO_TOTAL_MSGS_RETURNED	837	X'00000345'
MQIAMO_LAST_USED	837	X'00000345'

MQIAMO64_* (format de commande-Types de paramètre de surveillance des entiers 64 bits)

Tableau 187. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_ * (Sélecteurs système entiers)

Tableau 188. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIASY_FIRST	-1	X'FFFFFFFF'
MQIASY_CODED_CHAR_SET_ID	-1	X'FFFFFFFF'
TYPE_MQIAS	-2	X'FFFFFFFE'
MQIASY_COMMAND	-3	X'FFFFFFFD'
NUMERO MQIASY_MSG_SEQ_NO	-4	X'FFFFFFFC'
CONTROLE MQIASY_	-5	X'FFFFFFFB'
CODE COMP_MQIASY_	-6	X'FFFFFFFA'
MQIASY_MOTIF	-7	X'FFFFFFF9'
OPTIONS MQIASY_BAG_OPTIONS	-8	X'FFFFFFF8'
VERSION MQIASY_VERSION	-9	X'FFFFFFF7'
MQIASY_LAST_USED	-9	X'FFFFFFF7'
MQIASY_LAST	-2000	X'FFFFF830'

MQIAUT_ * (Authentificateur de l'en-tête d'informationsIMS)

Tableau 189. Noms et valeurs de constante

Nom	Valeur
MQIAUT_AUCUN	"¬¬¬¬¬¬¬¬"
MQIAUT_NON_ARRAY	'¬','¬','¬','¬','¬','¬','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQIAV_ * (Valeurs d'attribut d'entier)

Tableau 190. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIAV_NOT_APPLICABLE	-1	X'FFFFFFFF'
MQIAV_NON DEFINI	-2	X'FFFFFFFE'

MQICM_ * (modes de validation de l'en-tête d'informationsIMS)

Tableau 191. Noms et valeurs de constante

Nom	Valeur
MQICM_COMMIT_THEN_SEND	'0'
MQICM_SEND_THEN_COMMIT	'1'

MQIDO_ * (Options d'attente de validation du format de commande)

Tableau 192. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
VALIDATION_MQIDO_VALIDATION	1	X'00000001'
MQIDO_BACKOUT	2	X'00000002'

MQIEP_ * (points d'entrée d'interface)

Structure des paramètres de sécurité de connexion

Tableau 193. Structures de constantes

Nom	Structure
ID_STRUC_MQIPE	"IEP¬"
MQIEP_STRUC_ID_ARRAY	'I','E','P','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 194. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIEP_VERSION_1	1	X'00000001'
MQDXP_CURRENT_VERSION	1	X'00000001'

MQIGQ_ * (Mise en file d'attente intra-groupe)

Tableau 195. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIGQ_DISABLED	0	X'00000000'
MQIGQ_ENABLED	1	X'00000001'

MQIGQPA_ * (droits d'insertion de la mise en file d'attente intra-groupe)

Tableau 196. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIGQPA_PAR DEFAUT	1	X'00000001'

Tableau 196. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIGQPA_CONTEXT	2	X'00000002'
MQIGQPA_ONLY_IGQ	3	X'00000003'
MQIGQPA_ALTERNATE_OR_IGQ	4	X'00000004'

MQIIH_ * (indicateurs et structure d'en-tête d'informationIMS)

Structure d'en-tête d'informations IMS

Tableau 197. Structures de constantes	
Nom	**Structure**
ID MQIIH_STRUC_ID	"IIH¬"
MQIIH_STRUC_ID_ARRAY	'I','I','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 198. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIIH_VERSION_1	1	X'00000001'
MQIIH_VERSION_CURRENT_	1	X'00000001'
MQIIH_LENGTH_1	84	X'00000054'

Indicateurs d'en-tête d'informations IMS

Tableau 199. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIIH_AUCUN	0	X'00000000'
MQIIH_PASS_EXPIRATION	1	X'00000001'
MQIIH_EXPIRATION_ILLIMITÉE	0	X'00000000'
MQIIH_REPLY_FORMAT_NONE	8	X'00000008'
MQIIH_IGNORE_PURG	16	X'00000010'
MQIIH_CM0_REQUEST_RESPONSE	32	X'00000020'

MQIMPO_ * (interrogation des options et de la structure des propriétés de message)

Consulter la structure des options de propriété de message

Tableau 200. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQIMPO_	"IMPO"
MQIMPO_STRUC_ID_ARRAY	'I','M','P','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 201. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIMPO_VERSION_1	1	X'00000001'
MQIMPO_CURRENT_VERSION	1	X'00000001'

Interroger les options de propriété de message

Tableau 202. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIMPO_TYPE_CONVERT	2	X'00000002'
LONGUEUR_REQUÊTE_MQIMPO_LONG	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_ * (Dispositions entrantes du format de commande)

Tableau 203. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQINBD_Q_DIR	0	X'00000000'
MQINBD_GROUPE	3	X'00000003'

MQIND_ * (Valeurs d'index spéciales)

Tableau 204. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIND_AUCUN	-1	X'FFFFFFFF'
MQIND_ALL	-2	X'FFFFFFFE'

MQIPADDR_ * (versions d'adresse IP)

Tableau 205. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQIPADDR_IPV4	0	X'00000000'
MQIPADDR_IPV6	1	X'00000001'

MQISS_ * (portées de sécurité de l'en-tête d'informationsIMS)

Tableau 206. Noms et valeurs de constante	
Nom	**Valeur**
CONTROLE MQISS_CHECK	'C'
MQISS_FULL	'F'

MQIT_ * (types d'index)

Tableau 207. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQIT_AUCUN	0	X'00000000'
ID_MSG_MQ	1	X'00000001'
ID_CORREL_MQ	2	X'00000002'
JETON_MSG_MQIT_TOKEN	4	X'00000004'
ID_GROUPE_MQIT	5	X'00000005'

MQITEM_ * (Type d'élément pour mqInquireItemInfo)

Tableau 208. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQITEM_ENTIER	1	X'00000001'
MQITEM_CHAINE	2	X'00000002'
COUSSIN MQITEM_BAG	3	X'00000003'
MQITEM_BYTE_STRING	4	X'00000004'
FILTRE_ENTIER_MQITEM_ENTIER	5	X'00000005'
FILTRE_CHAÎNE_MQITEM_FILTER	6	X'00000006'
MQITEM_INTEGER64	7	X'00000007'
MQITEM_BYTE_STRING_FILTER	8	X'00000008'

MQITII_ * (identificateur d'instance de transaction de l'en-tête d'informationIMS)

Tableau 209. Noms et valeurs de constante

Nom	Valeur
MQITII_AUCUN	X'00...00' (16 valeurs nulles)
MQITII_NON_ARRAY	'\0','\0',... (16 valeurs nulles)

MQITS_ * (Etats de transaction de l'en-tête d'informationIMS)

Tableau 210. Noms et valeurs de constante

Nom	Valeur
MQITS_DANS_CONVERSATION	'C'
MQITS_NON_CONVERSATION	'¬'
MQITS_ARCHITECTURE	'A'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQKAI_ * (intervalleKeepAlive)

Tableau 211. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQKAI_AUTO	-1	X'FFFFFFFF'

MQMASTER_ * (administration maître)

Tableau 212. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQMASTER_NO	0	X'00000000'
MQMASTER_OUI	1	X'00000001'

MQMCAS_ * (Statut de l'agent de canal de message de format de commande)

Tableau 213. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQMCAS_ARRETA	0	X'00000000'
MQMCAS_EN COURS D'EXECUTION	3	X'00000003'

MQMCAT_ * (types MCA)

Tableau 214. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
PROCESSUS MQMCAT_PROCESS	1	X'00000001'
MQMCAT_THREAD	2	X'00000002'

MQMCD_ * (Informations sur la balise des options de publication / abonnement)

Options de publication / abonnement Balise Descripteur de contenu de message (mcd) Etiquettes

Tableau 215. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQMCD_FOLDER_VERSION	1	X'00000001'

Noms de balise d'options de publication / abonnement

Tableau 216. Noms et valeurs de constante

Nom	Valeur
DOMAINE_MSG_MQMCD_MESSAGE	"Msd"
MQMCD_MSG_SET	"Set"
TYPE MQMCD_MSG_TYPE	"Type"
FORMAT MQMCD_MSG_FORMAT	"Fmt"

Noms de balise XML des options de publication / abonnement

Tableau 217. Noms et valeurs de constante

Nom	Valeur
MQMCD_MSG_DOMAINE_B	"<Msd>"
MQMCD_MSG_DOMAINE_E	"</Msd>"
MQMCD_MSG_SET_B	"<Set>"
MQMCD_MSG_SET_E	"</Set>"

Tableau 217. Noms et valeurs de constante (suite)	
Nom	**Valeur**
MQMCD_MSG_TYPE_B	"<Type>"
MQMCD_MSG_TYPE_E	"</Type>"
MQMCD_MSG_FORMAT_B	"<Fmt>"
MQMCD_MSG_FORMAT_E	"</Fmt>"

Valeurs de balise d'options de publication / abonnement

Tableau 218. Noms et valeurs de constante	
Nom	**Valeur**
MQMCD_DOMAINE_AUCUN	"none"
MQMCD_DOMAIN_NEON	"neon"
MQMCD_DOMAIN_MRM	"mrm"
MQMCD_DOMAIN_JMS_NONE	"jms_none"
MQMCD_DOMAIN_JMS_TEXT	"jms_text"
OBJET_JMS_DOMAINE_MQMCD	"jms_object"
MAPPE_JMS_DOMAINE_MQMCD	"jms_map"
MQMCD_DOMAIN_JMS_STREAM	"jms_stream"
MQMCD_DOMAIN_JMS_BYTES	"jms_bytes"

MQMD_ * (structure de descripteur de message)

Tableau 219. Structures de constantes	
Nom	**Structure**
ID_STRUCD_MQM	"MD¬¬"
MQMD_STRUC_ID_ARRAY	'M','D','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 220. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMD_VERSION_1	1	X'00000001'
MQMD_VERSION_2	2	X'00000002'
MQMD_CURRENT_VERSION	2	X'00000002'

MQMDE_ * (structure d'extension de descripteur de message)

Tableau 221. Structures de constantes	
Nom	**Structure**
ID_STRUCDE_MQM	"MDE¬"
MQMDE_STRUC_ID_ARRAY	'M','D','E','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 222. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMDE_VERSION_2	2	X'00000002'
MQMDE_CURRENT_VERSION	2	X'00000002'
MQMDE_LENGTH_2	72	X'00000048'

MQMDEF_ * (indicateurs d'extension de descripteur de message)

Tableau 223. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMDEF_AUCUN	0	X'00000000'

MQMDS_ * (Séquence de distribution des messages)

Tableau 224. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMDS_PRIORITE	0	X'00000000'
MQMDS_FIFO	1	X'00000001'

MQMF_ * (indicateurs de message)

Tableau 225. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMF_SEGMENTATION_INHIBÉE	0	X'00000000'
MQMF_SEGMENTATION_ALLOWED	1	X'00000001'
GROUPE MQMF_MSG_IN_GROUP	8	X'00000008'
MQMF_LAST_MSG_IN_GROUP	16	X'00000010'
SEGMENT_MQMF	2	X'00000002'
SEGMENT_DERNIER_MQMF_SEGMENT	4	X'00000004'
MQMF_AUCUN	0	X'00000000'

MQMHBO_ * (descripteur de message vers les options et la structure de la mémoire tampon)

Descripteur de message dans la structure des options de mémoire tampon

Tableau 226. Structures de constantes	
Nom	**Structure**
MQMHBO_STRUC_ID	"MHBO"
MQMHBO_STRUC_ID_ARRAY	'M','H','B','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 227. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMHBO_VERSION_1	1	X'00000001'
MQMHBO_CURRENT_VERSION	1	X'00000001'

Options de traitement des messages vers la mémoire tampon

Tableau 228. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMHBO_PROPERTIES_IN_MQRFH2	1	X'00000001'
MQMHBO_DELETE_PROPERTIES	2	X'00000002'
MQMHBO_AUCUN	0	X'00000000'

MQMI_ * (identificateur de message)

Tableau 229. Noms et valeurs de constante	
Nom	**Valeur**
MQMI_AUCUN	X'00...00' (24 valeurs nulles)
MQMI_NON_ARRAY	'\0','\0',... (24 valeurs nulles)

MQMMBI_ * (Intervalle entre les marques de message)

Tableau 230. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMMBI_UNLIMITED	-1	X'FFFFFFFF'

MQMO_ * (options de correspondance)

Tableau 231. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
ID MQMO_MATCH_MSG_ID	1	X'00000001'
ID_CORREL_MQMO_MATCH_	2	X'00000002'
ID_GROUPE_MQMO_MATCH	4	X'00000004'
NUMERO MQMO_MATCH_MSG_SEQ_NO	8	X'00000008'
MQMO_MATCH_OFFSET	16	X'00000010'
MQMO_MATCH_MSG_TOKEN	32	X'00000020'
MQMO_AUCUN	0	X'00000000'

MQMODE_ * (Options de mode de format de commande)

Tableau 232. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
FORCE MQMODE	0	X'00000000'
MQMODE_QUIESCE	1	X'00000001'
MQMODE_TERMINATE	2	X'00000002'

MQMON_ * (Valeurs de surveillance)

Tableau 233. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMON_NON DISPONIBLE	-1	X'FFFFFFFF'
MQMON_AUCUN	-1	X'FFFFFFFF'

Tableau 233. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQMON_Q_DIR	-3	X'FFFFFFFD'
MQMON_OFF	0	X'00000000'
MQMON_ON	1	X'00000001'
MQMON_DISABLED	0	X'00000000'
MQMON_ENABLED	1	X'00000001'
MQMON_FAIBLE	17	X'00000011'
MQMON_MEDIUM	33	X'00000021'
MQMON_ELEVE	65	X'00000041'

MQMT_ * (types de message)

Tableau 234. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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 (CHAMP_MQT)	113	X'00000071'
MQMT_SYSTEM_LAST	65535	X'0000FFFF'
MQMT_APPL_FIRST	65536	X'00010000'
MQMT_APPL_LAST	999999999	X'3B9AC9FF'

MQMTOK_ * (jeton de message)

Tableau 235. Noms et valeurs de constante	
Nom	**Valeur**
MQMTOK_NONE	X'00...00' (16 valeurs nulles)
MQMTOK_NON_ARRAY	'\0','\0',... (16 valeurs nulles)

MQNC_ * (nombre de noms)

Tableau 236. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQNC_MAX_NAMELIST_NAME_COUNT	256	X'00000100'

MQNPM_ * (classe de message non persistant)

Tableau 237. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQNPM_CLASS_NORMAL	0	X'00000000'
MQNPM_CLASS_HIGH	10	X'0000000A'

MQNPMS_ * (NonPersistent-Messages)

Tableau 238. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQNPMS_NORMAL	1	X'00000001'
MQNPMS_FAST	2	X'00000002'

MQNT_ * (Types de liste de noms)

Tableau 239. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQNT_AUCUN	0	X'00000000'
MQNT_Q	1	X'00000001'
MQNT_CLUSTER	2	X'00000002'
INFO MQNT_AUTH_INFO	4	X'00000004'
MQNT_ALL	1001	X'000003E9'

MQNVS_ * (Noms de la chaîne de nom / valeur)

Tableau 240. Noms et valeurs de constante

Nom	Valeur
MQNVS_TYPE D'APPLICATION	"OPT_APP_GRP¬"
TYPE MQNVS_MSG_	"OPT_MSG_TYPE¬"

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQOA_ * (Limites des sélecteurs pour les attributs d'objet)

Tableau 241. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQOA_PREMIER	1	X'00000001'
MQOA_LAST	9000	X'00002328'

MQOD_ * (structure de descripteur d'objet)

Tableau 242. Structures de constantes

Nom	Structure
ID_STRUC_MQODE	"OD¬¬"
MQOD_STRUC_ID_ARRAY	'O','D','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 243. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQOD_VERSION_1	1	X'00000001'
MQOD_VERSION_2	2	X'00000002'
MQOD_VERSION_3	3	X'00000003'
MQOD_VERSION_4	4	X'00000004'

Tableau 243. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
VERSION MQOD_CURRENT_VERSION	4	X'00000004'
MQOD_LONGUEUR_ACTUELLE	(value differs by platform or version)	

MQOII_ * (identificateur d'instance d'objet)

Tableau 244. Noms et valeurs de constante	
Nom	**Valeur**
MQOII_AUCUN	X'00...00' (24 valeurs nulles)
MQOII_NONE_ARRAY	'\0','\0',... (24 valeurs nulles)

MQOL_ * (Longueur d'origine)

Tableau 245. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQOL_UNDEFINI	-1	X'FFFFFFFF'

MQOM_ * (options des messages Db2 obsolètes sur le groupe Inquire)

Tableau 246. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQOM_NO	0	X'00000000'
MQOM_OUI	1	X'00000001'

MQOO_ * (options d'ouverture)

Tableau 247. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQOO_BIND_AS_Q_DEF	0	X'00000000'
MQOO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQOO_INPUT_AS_Q_DEF	1	X'00000001'
MQOO_INPUT_SHARED	2	X'00000002'
MQOO_INPUT_EXCLUSIVE	4	X'00000004'
MQOO_BROWSE	8	X'00000008'
MQOO_SORTIE	16	X'00000010'
MQOO_INTERROGATION	32	X'00000020'
MQOO_SET	64	X'00000040'
MQOO_SAVE_ALL_CONTEXT	128	X'00000080'
MQOO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQOO_PASS_ALL_CONTEXT	512	X'00000200'
MQOO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQOO_SET_ALL_CONTEXT	2048	X'00000800'
MQOO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'
MQOO_FAIL_IF_QUIESCING	8192	X'00002000'
MQOO_BIND_ON_OPEN	16384	X'00004000'

Tableau 247. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQOO_BIND_NOT_FIXED	32768	X'00008000'
MQOO_CO_OP	131072	X'00020000'
MQOO_RESOLVE_LOCAL_TOPIC	262144	X'00040000'
MQOO_NO_READ_AHEAD	524288	X'00080000'
MQOO_READ_AHEAD	1048576	X'00100000'
MQOO_BIND_ON_GROUPE	4194304	X'00400000'

MQOO_ * (suivi utilisé en C++ uniquement)

Tableau 248. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQOO_NOMS_RÉSOLUTION	65536	X'00010000'
MQOO_RÉSOLVE_LOCAL_Q	262144	X'00040000'

MQOP_ * (codes opération pour MQCTL et MQCB)

Codes opération pour MQCTL

Tableau 249. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
DEMARRAGE MQOP	1	X'00000001'
MQOP_START_WAIT	2	X'00000002'
MQOP_ARRET	4	X'00000004'

Codes opération pour MQCB

Tableau 250. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQOP_REGISTER	256	X'00000100'
MQOP_DÉSENREGISTREMENT	512	X'00000200'

Codes opération pour MQCTL et MQCB

Tableau 251. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQOP_SUSPENSION	65536	X'00010000'
MQOP_RESUME	131072	X'00020000'

MQOPEN_ * (valeurs associées à la structure MQOPEN_PRIV)

Tableau 252. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQOPEN_PRIV_VERSION_1	1	X'00000001'
MQOPEN_PRIV_CURRENT_VERSION	1	X'00000001'

MQOPER_ * (Opérations d'activité)

Tableau 253. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQOPER_SYSTEM_FIRST	0	X'00000000'
MQOPER_INCONNU	0	X'00000000'
MQOPER_PARCOURIR	1	X'00000001'
Carte MQOPER_DISCARD	2	X'00000002'
MQOPER_GET	3	X'00000003'
MQOPER_PUT	4	X'00000004'
MQOPER_PUT_REPLY	5	X'00000005'
RAPPORT MQOPER_PUT_REPORT	6	X'00000006'
MQOPER_RECEIVE	7	X'00000007'
MQOPER_SEND	8	X'00000008'
MQOPER_TRANSFORM	9	X'00000009'
MQOPER_PUBLICATION	10	X'0000000A'
MQOPER_EXCLUDED_PUBLISH	11	X'0000000B'
MQOPER_DISCARDED_PUBLISH	12	X'0000000C'
MQOPER_SYSTEM_LAST	65535	X'0000FFFF'
MQOPER_APPL_FIRST	65536	X'00010000'
MQOPER_APPL_LAST	999999999	X'3B9AC9FF'

MQOT_ * (types d'objet et types d'objet étendu)

Types d'objet

Tableau 254. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQOT_AUCUN	0	X'00000000'
MQOT_Q	1	X'00000001'
MQOT_NAMELIST	2	X'00000002'
PROCESSUS MQ	3	X'00000003'
CLASSE DE STOCKAGE MQOT_STORAGE_CLASS	4	X'00000004'
MQOT_Q_DIR	5	X'00000005'
CANAL_MQTON	6	X'00000006'
INFO MQOT_AUTH_INFO	7	X'00000007'
MQOT_TOPIC	8	X'00000008'
MQOT_CF_STRUC	10	X'0000000A'
MQOT_PROGRAMME d'écoute	11	X'0000000B'
SERVICE MQOT	12	X'0000000C'
MQOT_RESERVED_1	999	X'000003E7'

Types d'objet étendus

Tableau 255. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'
CANAL_SERVEUR_MQOT_CANAL	1008	X'000003F0'
CANAL_DEMANDE_MQOT_CANAL	1009	X'000003F1'
CANAL_RÉCEPTEUR_MQOT_CANAL	1010	X'000003F2'
MQOT_CANAL_EN_COURS	1011	X'000003F3'
CANAL_SAUVEGARDE_MQOT_SAVED_ENREGISTREMENT	1012	X'000003F4'
CANAL_SVRCONN_MQOT_SVRCONN_CHANNEL	1013	X'000003F5'
MQOT_CLNTCONN_CHANNEL	1014	X'000003F6'
MQOT_SHORT_CHANNEL	1015	X'000003F7'

MQPA_ * (droits d'insertion)

Tableau 256. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPA_PAR DEFAUT	1	X'00000001'
MQPA_CONTEXT	2	X'00000002'
MQPA_ONLY_MCA	3	X'00000003'
MQPA_ALTERNATE_OR_MCA	4	X'00000004'

MQPD_ * (descripteur de propriété, support et contexte)

Structure du descripteur de propriété

Tableau 257. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQD	"PD¬¬"
MQPD_STRUC_ID_ARRAY	'P','D','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 258. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPD_VERSION_1	1	X'00000001'
MQPD_CURRENT_VERSION	1	X'00000001'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Options de descripteur de propriété

Tableau 259. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPD_AUCUN	0	X'00000000'

Options de prise en charge des propriétés

Tableau 260. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPD_SUPPORT_XX_ENCODE_CASE_ONE facultatif	1	X'00000001'
MQPD_SUPPORT_REQUIS	1048576	X'00100000'
MQPD_SUPPORT_REQUIRED_IF_LOCAL	1024	X'00000400'
MQPD_REJECT_UNSUP_MASK	-1048576	X'FFF00000'
MQPD_ACCEPT_UNSUP_IF_XMIT_MASK	1047552	X'000FFC00'
MQPD_ACCEPT_UNSUP_MASK	1023	X'000003FF'

Contexte de propriété

Tableau 261. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPD_NO_CONTEXT	0	X'00000000'
CONTEXTE MQPD_USER_	1	X'00000001'

MQPER_ * (Valeurs de persistance)

Tableau 262. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPER_PERSISTENCE_AS_PARENT	-1	X'FFFFFFFF'
MQPER_NON_PERSISTENT	0	X'00000000'
MQPER_PERSISTANT	1	X'00000001'
MQPER_PERSISTENCE_AS_Q_DEF	2	X'00000002'
MQPER_PERSISTENCE_AS_TOPIC_DEF	2	X'00000002'

MQPL_ * (plateformes)

MQPL_MVS	1	X'00000001'
MQPL_OS390	1	X'00000001'
MQPL_ZOS	1	X'00000001'
MQPL_OS2	2	X'00000002'
MQPL_AIX	3	X'00000003'
MQPL_UNIX	3	X'00000003'
MQPL_OS400	4	X'00000004'
MQPL_WINDOWS	5	X'00000005'
MQPL_WINDOWS_NT	11	X'0000000B'

MQPL_VMS	12	X'0000000C'
MQPL_NSK	13	X'0000000D'
MQPL_NSS	13	X'0000000D'
MQPL_OPEN_TP1	15	X'0000000F'
MQPL_MV	18	X'00000012'
MQPL_TPF	23	X'00000017'
MQPL_VSE	27	X'0000001B'
MQPL_NATIVE	1	X'00000001'

MQPMO_ * (Options de message d'insertion et structure pour le masque de publication)

Structure des options de message d'insertion

Tableau 263. Structures de constantes

Nom	Structure
ID_STRUC_MQPMO_	"PMO¬"
MQPMO_STRUC_ID_ARRAY	'P','M','O','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 264. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_LONGUEUR_ACTUELLE	(value differs by platform or version)	(value differs by platform or version)

Options d'insertion de message

Tableau 265. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPMO_SYNCPOINT	2	X'00000002'
MQPMO_NO_SYNCPOINT	4	X'00000004'
MQPMO_CONTEXTE_PAR_DÉFAUT	32	X'00000020'
ID MQPMO_NEW_MSG_ID	64	X'00000040'
MQPMO_NEW_CORREL_ID	128	X'00000080'
MQPMO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQPMO_PASS_ALL_CONTEXT	512	X'00000200'
MQPMO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQPMO_SET_ALL_CONTEXT	2048	X'00000800'
MQPMO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'

Tableau 265. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_REPONSE	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_REPONSE_AS_Q_DEF	0	X'00000000'
MQPMO_REPONSE_AS_TOPIC_DEF	0	X'00000000'
MQPMO_NONE	0	X'00000000'

Options d'insertion de message pour le masque de publication

Tableau 266. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPMO_PUB_OPTIONS_MASK	2097152	X'00200000'

MQPMRF_ * (Zones d'enregistrement de message d'insertion)

Tableau 267. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
ID MQPMRF_MSG_ID	1	X'00000001'
ID_CORREL_MQPMRF_	2	X'00000002'
ID_GROUPE_MQPMRF_ID	4	X'00000004'
MQPMRF_FEEDBACK	8	X'00000008'
MQPMRF_COMPTING_TOKEN	16	X'00000010'
MQPMRF_AUCUN	0	X'00000000'

MQPO_ * (options de purge de format de commande)

Tableau 268. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPO_OUI	1	X'00000001'
MQPO_NON	0	X'00000000'

MQPRI_ * (priorité)

Tableau 269. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPRI_PRIORITY_AS_Q_DEF	-1	X'FFFFFFFF'
MQPRI_PRIORITY_AS_PARENT	-2	X'FFFFFFFE'
MQPRI_PRIORITY_AS_PUBLISHED	-3	X'FFFFFFFD'
MQPRI_PRIORITY_AS_TOPIC_DEF	-1	X'FFFFFFFF'

MQPROP_ * (Valeurs de contrôle de propriété de file d'attente et de canal et longueur maximale des propriétés)

Valeurs de contrôle de propriété de file d'attente et de canal

Tableau 270. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
COMPATIBILITE_MQPROP_COMPATIBILITÉ	0	X'00000000'
MQPROP_NONE	1	X'00000001'
MQPROP_ALL	2	X'00000002'
MQPROP_FORCE_MQRFH2	3	X'00000003'

Longueur maximale des propriétés

Tableau 271. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPROP_UNRESTRICTED_LENGTH	-1	X'FFFFFFFF'

MQPRT_ * (Valeurs de réponse d'insertion)

Tableau 272. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPRT_RESPONSE_AS_PARENT	0	X'00000000'
MQPRT_SYNC_REPONSE	1	X'00000001'
MQPRT_ASYNC_RESPONSE	2	X'00000002'

MQPS_ * (Publier/S' abonner)

Statut de publication / d'abonnement du format de commande

Tableau 273. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPS_STATUS_INACTIVE	0	X'00000000'
MQPS_STATUS_DEMARRAGE	1	X'00000001'
MQPS_STATUS_STOPPING	2	X'00000002'
STATUS_MQPS ACTIF	3	X'00000003'
MQPS_STATUS_COMPAT	4	X'00000004'
MQPS_STATUS_ERREUR	5	X'00000005'

Tableau 273. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPS_STATUS_REFUSE	6	X'00000006'

Balises de publication / abonnement en tant que chaînes

Tableau 274. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
COMMANDE MQPS	"MQPSCommand"	
CODE COMP_MQPS	"MQPSCompCode"	
ID_CORREL_MQPS	"MQPSCorrelId"	
OPTIONS DE DELETE_MQPS	"MQPSDelOpts"	
ID_ERREUR_MQPS	"MQPSErrorId"	
MQPS_ERREUR_POS	"MQPSErrorPos"	
DONNEES_INTEGER_MQPS	"MQPSIntData"	
MQPS_PARAMETER_ID	"MQPSParmId"	
OPTIONS DE PUBLICATION_MQPS	"MQPSPubOpts"	
MQPS_PUBLISH_TIMESTAMP	"MQPSPubTime"	
MQPS_Q_MGR_NAME	"MQPSQMgrName"	
NOM_Q_MQPS	"MQPSQName"	
MQPS_MOTIF	"MQPSReason"	
MQPS_MOTIF	"MQPSReasonText"	
OPTIONS D'ENREGISTREMENT MQPS	"MQPSRegOpts"	
NUMÉRO_SÉQUENCE_MQPS_	"MQPSSeqNum"	
NOM_FLOT_MQPS_	"MQPSStreamName"	
DONNEES_STRING_MQPS	"MQPSStringData"	
MQPS_SUBSCRIPTION_IDENTITY	"MQPSSubIdentity"	
NOM_SUBSCRIPTION_MQPS	"MQPSSubName"	
MQPS_SUBSCRIPTION_USER_DATA	"MQPSSubUserData"	
MQPS_TOPIC	"MQPSTopic"	
ID_UTILISATEUR_MQPS	"MQPSUserId"	

Balises de publication / abonnement sous forme de chaînes vides incluses

Tableau 275. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPS_COMMAND_B	"bMQPSCommandb"	
MQPS_COMP_CODE_B	"bMQPSCompCodeb"	
MQPS_CORREL_ID_B	"bMQPSCorrelIdb"	
MQPS_DELETE_OPTIONS_B	"bMQPSDelOptsb"	
ID_ERREUR_MQPS B	"bMQPSErrorIdb"	
MQPS_ERRE_POS_B	"bMQPSErrorPosb"	
MQPS_INTEGER_DATA_B	"bMQPSIntDatab"	

Tableau 275. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPS_PARAMETER_ID_B	"bMQPSParmIdb"	
MQPS_PUBLICATION_OPTIONS_B	"bMQPSPubOptsb"	
MQPS_PUBLISH_HORODATAGE_B	"bMQPSPubTimeb"	
MQPS_Q_MGR_NAME_B	"bMQPSQMgrNameb"	
MQPS_Q_NOM_B	"bMQPSQNameb"	
MQPS_MOTIF B	"bMQPSReasonb"	
MQPS_REASON_TEXT_B	"bMQPSReasonTextb"	
MQPS_REGISTRATION_OPTIONS_B	"bMQPSRegOptsb"	
NOMBRE_SÉQUENCE_MQPS_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"	
ID_utilisateur_MQPS B	"bMQPSUserIdb"	

Valeurs de balise de commande de publication / abonnement sous forme de chaînes

Tableau 276. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPS_DELETE_PUBLICATION	"DeletePub"	
MQPS_DÉSENREGISTRE_PUBLIEUR	"DeregPub"	
ABONNEMENT_DÉSENREGISTRE_MQP	"DeregSub"	
MQPS_PUBLICATION	"Publish"	
MQPS_REGISTER_DIFFUSEUR de publications	"RegPub"	
ABONNÉ_REGISTER_MQPS_	"RegSub"	
MQPS_REQUEST_UPDATE	"ReqUpdate"	

Valeurs de balise de commande de publication / abonnement sous forme de chaînes encadrées par des blancs

Tableau 277. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPS_DELETE_PUBLICATION_B	"bDeletePubb"	
MQPS_DÉSENREGISTRE_PUBLISHER_B	"bDeregPubb"	
MQPS_DEREGISTER_SUBSCRIBER_B	"bDeregSubb"	
MQPS_PUBLISH_B	"bPublishb"	
MQPS_REGISTER_PUBLISHER_B	"bRegPubb"	
MQPS_REGISTER_SUBSCRIBER_B	"bRegSubb"	
MQPS_REQUEST_UPDATE_B	"bReqUpdateb"	

Valeurs de balise d'options de publication / abonnement sous forme de chaînes

Tableau 278. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
NOM_ADD_MQ	"AddName"	
MQPS_ANONYME	"Anon"	
MQPS_CORREL_ID_AS_IDENTITE	"CorrelAsId"	
MQPS_DEREGISTER_ALL	"DeregAll"	
MQPS_DIRECT_REQUESTS	"DirectReq"	
MQPS_DUPLICATES_OK	"DupsOK"	
MQPS_RÉPONSE_COMPLET_RÉPONSE	"FullResp"	
MQPS_INCLUDE_STREAM_NAME	"InclStreamName"	
MQPS_INFORM_SI_RETENU	"InformIfRet"	
MQPS_IS_RETAINED_PUBLICATION	"IsRetainedPub"	
MQPS_JOIN_EXCLUSIF	"JoinExcl"	
MQPS_JOIN_SHARED	"JoinShared"	
MQPS_LEAVE_ONLY	"LeaveOnly"	
MQPS_LOCAL	"Local"	
MQPS_VERROUILLÉ	"Locked"	
MQPS_NEW_PUBLICATIONS_ONLY	"NewPubsOnly"	
MQPS_NON_MODIFICATION	"NoAlter"	
MQPS_NO_REGISTRATION	"NoReg"	
MQPS_NON_PERSISTENT	"NonPers"	
MQPS_AUCUN	"None"	
MQPS_AUTRE_ABONNEMENTS uniquement	"OtherSubsOnly"	
MQPS_PERSISTENT	"Pers"	
MQPS_PERSISTENT_AS_PUBLISH	"PersAsPub"	
MQPS_XX_ENCODE_CASE_ONE persistant-AS_Q	"PersAsQueue"	
MQPS_PUBLISH_ON_REQUEST_ONLY	"PubOnReqOnly"	
PUBLICATION MQPS_RETAIN_	"RetainPub"	
ID_UTILISATEUR_MQPS_VARIABLE_	"VariableUserId"	

Valeurs de balise d'options de publication / d'abonnement sous forme de chaînes entourées de blancs

Tableau 279. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_DUPLICES_OK_B	"bDupsOKb"	

Tableau 279. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_AUTRE_ABONNEMENTS_ONLY_B	"bOtherSubsOnlyb"	
MQPS_XX_ENCODE_CASE_ONE persistant _b	"bPersb"	
MQPS_PERSISTENT_AS_PUBLISH_B	"bPersAsPubb"	
MQPS_XX_ENCODE_CASE_ONE persistant AS_q_b	"bPersAsQueueb"	
MQPS_PUBLISH_ON_REQUEST_ONLY_B	"bPubOnReqOnlyb"	
MQPS_RETAIN_PUBLICATION_B	"bRetainPubb"	
ID_UTILISATEUR_MQPS_VARIABLE_B	"bVariableUserIdb"	

MQPSC_ * (balises d'options de publication / d'abonnement-Dossier de commandes de publication / d'abonnement (psc))

Tableau 280. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPSC_FOLDER_VERSION	1	X'00000001'

MQPSC_ * (noms de balise d'options de publication / abonnement)

Tableau 281. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPSC_COMMAND	"Command"	
OPTION D'ENREGISTREMENT MQPS	"RegOpt"	
OPTION MQPSC_PUBLICATION_OPTION	"PubOpt"	
OPTION DELETE_MQPS	"DelOpt"	
MQPSC_TOPIC	"Topic"	
MQPSC_SUBSCRIPTION_POINT	"SubPoint"	
MQPSC_FILTER	"Filter"	
MQPSC_Q_MGR_NAME	"QMgrName"	
MQPSC_Q_NAME	"QName"	

Tableau 281. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPSC_HORODATAGE de publication	"PubTime"	
NUMERO_SEQUENCE_MQPS	"SeqNum"	
NOM_SOUS-SCRIPTION_MQPS	"SubName"	
MQPSC_SUBSCRIPTION_IDENTITY	"SubIdentity"	
MQPSC_SUBSCRIPTION_DONNEES_UTILISATEUR	"SubUserData"	
ID_CORREL_MQPS	"CorrelId"	

MQPSC_ * (noms de balise XML des options de publication / abonnement)

Tableau 282. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_HORODATAGE_B	"<PubTime>"	
MQPSC_PUBLISH_TIMESTAMP_E	"</PubTime>"	
MQPSC_NUMÉRO_SÉQUENCE_B	"<SeqNum>"	
MQPSC_NUMÉRO_SÉQUENCE_E	"</SeqNum>"	
MQPSC_SUBSCRIPTION_NAME_B	"<SubName>"	
MQPSC_SUBSCRIPTION_NAME_E	"</SubName>"	
MQPSC_SUBSCRIPTION_IDENTITY_B	"<SubIdentity>"	
MQPSC_SUBSCRIPTION_IDENTITY_E	"</SubIdentity>"	
MQPSC_SUBSCRIPTION_USER_DATA_B	"<SubUserData>"	
MQPSC_SUBSCRIPTION_USER_DATA_E	"</SubUserData>"	
MQPSC_CORREL_ID_B	"<CorrelId>"	

Tableau 282. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
ID_CORREL_MQPS E	"</CorrelId>"	

MQPSC_ * (Valeurs de balise d'options de publication / abonnement sous forme de chaînes)

Tableau 283. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQPSC_DELETE_PUBLICATION	"DeletePub"	
MQPSC_DÉSENREGISTRE_ABONNÉ	"DeregSub"	
MQPSC_PUBLISH	"Publish"	
MQPSC_REGISTER_ABONNÉ	"RegSub"	
MQPSC_REQUEST_UPDATE	"ReqUpdate"	

MQPSC_ * (Valeurs de balise d'options de publication / abonnement sous forme de chaînes)

Tableau 284. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
NOM_ADD_MQPS	"AddName"	
MQPSC_CORREL_ID_AS_IDENTITY	"CorrelAsId"	
MQPSC_DEREGISTER_ALL	"DeregAll"	
MQPSC_DUPLICATES_OK	"DupsOK"	
MQPSC_FULL_RESPONSE	"FullResp"	
MQPSC_INFORM_IF_RETAINED	"InformIfRet"	
MQPSC_IS_RETAINED_PUB	"IsRetainedPub"	
MQPSC_JOIN_SHARED	"JoinShared"	
MQPSC_JOIN_EXCLUSIVE	"JoinExcl"	
MQPSC_LEAVE_ONLY	"LeaveOnly"	
MQPSC_LOCAL	"Local"	
MQPSC_LOCKED	"Locked"	
MQPSC_NEW_PUBS_ONLY	"NewPubsOnly"	
MQPSC_NO_ALTERATION	"NoAlter"	
MQPSC_NON_PERSISTENT	"NonPers"	
MQPSC_OTHER_SUBS_ONLY	"OtherSubsOnly"	
MQPSC_XX_ENCODE_CASE_ONE persistant	"Pers"	
MQPSC_PERSISTENT_AS_PUBLISH	"PersAsPub"	
MQPSC_XX_ENCODE_CASE_ONE persistENT_AS_q	"PersAsQueue"	
MQPSC_NONE	"None"	
MQPSC_PUB_ON_REQUEST_ONLY	"PubOnReqOnly"	
MQPSC_RETAIN_PUB	"RetainPub"	
ID_UTILISATEUR MQPSC_VARIABLE_	"VariableUserId"	

MQPSCR_ * (options de publication / abonnement)

Options de publication / abonnement Etiquette Dossier de réponses de publication / abonnement (pscr) Etiquettes

Tableau 285. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPSCR_FOLDER_VERSION	1	X'00000001'

Noms de balise d'options de publication / abonnement

Tableau 286. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPSCR_COMPLÉTION	"Completion"	
MQPSCR_REPONSE	"Response"	
MQPSCR_MOTIF	"Reason"	

Noms de balise XML des options de publication / abonnement

Tableau 287. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPSCR_COMPLETION_B	"<Completion>"	
MQPSCR_COMPLETION_E	"</Completion>"	
REPONSE_MQPSCR_B	"<Response>"	
REPONSE_MQPSCR_E	"</Response>"	
MQPSCR_MOTIF B	"<Reason>"	
MQPSCR_MOTIF	"</Reason>"	

Valeurs de balise d'options de publication / abonnement

Tableau 288. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPSCR_OK	"ok"	
AVERTISSEMENT MQPSCR_avertissement	"warning"	
MQPSCR_ERREUR	"error"	

MQPSM_ * (mode de publication / abonnement)

Tableau 289. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPSM_DISABLED	0	X'00000000'
MQPSM_COMPAT	1	X'00000001'
MQPSM_ENABLED	2	X'00000002'

MQPSPROP_ * (Propriétés de message de publication / abonnement)

Tableau 290. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPSPROP_NONE	0	X'00000000'
MQPSPROP_COMPAT	1	X'00000001'
MQPSPROP_RFH2	2	X'00000002'
MQPSPROP_MSGPROP	3	X'00000003'

MQPSST_ * (format de commande-Type de statut de publication / d'abonnement)

Tableau 291. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPSST_ALL	0	X'00000000'
MQPSST_LOCAL	1	X'00000001'
MQPSST_PARENT	2	X'00000002'
MQPSST_CHILD	3	X'00000003'

MQPUBO_ * (Options de publication / abonnement)

Tableau 292. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_REGISTRATION	8	X'00000008'
MQPUBO_IS_RETAINED_PUBLICATION	16	X'00000010'

MQPXP_ * (structure des paramètres d'exit de routage de publication / abonnement)

Tableau 293. Structures de constantes

Nom	Structure
MQPXP_STRUC_ID	"PXP¬"
MQPXP_STRUC_ID_ARRAY	'P','X','P','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 294. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQPXP_VERSION_1	1	X'00000001'
MQPXP_CURRENT_VERSION	1	X'00000001'

MQQA_ * (attributs de file d'attente)

Interdire l'obtention de valeurs

Tableau 295. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQA_GET_INHIBÉE	1	X'00000001'
MQQA_GET_ALLOWED	0	X'00000000'

Interdire les valeurs d'insertion

Tableau 296. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQA_PUT_INHIBÉ	1	X'00000001'
MQQA_PUT_ALLOWED	0	X'00000000'

Partage de file d'attente

Tableau 297. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQA_SHAREABLE	1	X'00000001'
MQQA_NOT_SHAREABLE	0	X'00000000'

Renforcement de l'arrière-système

Tableau 298. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQA_BACKOUT_HARDENED	1	X'00000001'
MQQA_BACKOUT_NOT_HARDENED	0	X'00000000'

MQQDT_ * (types de définition de file d'attente)

Tableau 299. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQDT_PRÉDÉFINI	1	X'00000001'
MQQDT_PERMANENT_DYNAMIQUE	2	X'00000002'
MQQDT_TEMPORARY_DYNAMIQUE	3	X'00000003'
MQQDT_PARTAGE_DYNAMIQUE	4	X'00000004'

MQQF_ * (indicateurs de file d'attente)

Tableau 300. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQF_LOCAL_Q	1	X'00000001'
MQQF_CLWL_USEQ_ANY	64	X'00000040'
MQQF_CLWL_USEQ_LOCAL	128	X'00000080'

MQQMDT_ * (format de commande-Types de définition de gestionnaire de files d'attente)

Tableau 301. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_ * (indicateurs de gestionnaire de files d'attente)

Tableau 302. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQMF_REPOSITORY_Q_MGR	2	X'00000002'
MQQMF_CLUSSDR_USER_DÉFINI	8	X'00000008'
MQQMF_CLUSSDR_AUTO_DEFINED	16	X'00000010'
MQQMF_XX_ENCODE_CASE_ONE disponible	32	X'00000020'

MQQMFAC_ * (fonction de gestionnaire de files d'attente de format de commande)

Tableau 303. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQMFAC_IMS_BRIDGE	1	X'00000001'
MQQMFAC_DB2	2	X'00000002'

MQQMSTA_ * (statut du gestionnaire de files d'attente de format de commande)

Tableau 304. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQMSTA_DEMARRAGE	1	X'00000001'
MQQMSTA_EN COURS D'EXECUTION	2	X'00000002'
MQQMSTA_MISE au repos	3	X'00000003'

MQQMT_ * (Types de gestionnaire de files d'attente de format de commande)

Tableau 305. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQMT_NORMAL	0	X'00000000'
MQQMT_REPOSITORY	1	X'00000001'

MQQO_ * (Options de mise au repos du format de commande)

Tableau 306. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQO_OUI	1	X'00000001'

Tableau 306. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQQO_NON	0	X'00000000'

MQQSGD_ * (Dispositions de groupe de partage de files d'attente)

Tableau 307. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQQSGD_ALL	-1	X'FFFFFFFF'
MQQSGD_Q_DIR	0	X'00000000'
MQQSGD_COPY	1	X'00000001'
MQQSGD_SHARED	2	X'00000002'
GROUPE_MQQSG	3	X'00000003'
MQQSGD_PRIVE	4	X'00000004'
MQQSGD_LIVE	6	X'00000006'

MQQSGS_ * (statut QSG de format de commande)

Tableau 308. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQQSGS_INCONNU	0	X'00000000'
MQQSGS_CREE	1	X'00000001'
MQQSGS_ACTIF	2	X'00000002'
MQQSGS_XX_ENCODE_CASE_ONE inactif	3	X'00000003'
MQQSGS_ECHEC	4	X'00000004'
MQQSGS_PENDING	5	X'00000005'

MQQSIE_ * (Service de file d'attente de format de commande-Evénements d'intervalle)

Tableau 309. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQQSIE_NONE	0	X'00000000'
MQQSIE_HIGH	1	X'00000001'
MQQSIE_OK	2	X'00000002'

MQQSO_ * (options d'ouverture de statut de file d'attente de format de commande pour SET, BROWSE, INPUT)

Tableau 310. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQQSO_NO	0	X'00000000'
MQQSO_OUI	1	X'00000001'
MQQSO_SHARED	1	X'00000001'
MQQSO_EXCLUSIF	2	X'00000002'

MQQSOT_ * (Types d'ouverture de statut de file d'attente de format de commande)

Tableau 311. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQSOT_ALL	1	X'00000001'
MQQSOT_ENTREE	2	X'00000002'
MQQSOT_OUTPUT	3	X'00000003'

MQQSUM_ * (format de commande-Messages de statut de file d'attente non validés)

Tableau 312. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQSUM_OUI	1	X'00000001'
MQQSUM_NO	0	X'00000000'

MQQT_ * (types de file d'attente et types de file d'attente étendue)

Types de file d'attente

Tableau 313. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQT_LOCAL	1	X'00000001'
MQQT_MODELE	2	X'00000002'
MQQT_ALIAS (alias MQ)	3	X'00000003'
MQQT_REMOTE	6	X'00000006'
MQQT_CLUSTER	7	X'00000007'

Types de file d'attente étendue

Tableau 314. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQQT_ALL	1001	X'000003E9'

MQRC_ * (codes anomalie)

Tableau 315. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRC_NONE	0	X'00000000'
MQRC_APPL_FIRST	900	X'00000384'
MQRC_APPL_LAST	999	X'000003E7'
MQRC_ALIAS_BASE_Q_TYPE_ERROR	2001	X'000007D1'
MQRC_ALREADY_CONNECTED	2002	X'000007D2'
MQRC_BACKED_OUT	2003	X'000007D3'
MQRC_BUFFER_ERROR	2004	X'000007D4'
MQRC_BUFFER_LENGTH_ERROR	2005	X'000007D5'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_CHAR_ATTR_LENGTH_ERROR	2006	X'000007D6'
MQRC_CHAR_ATTRS_ERREUR	2007	X'000007D7'
MQRC_CHAR_ATTRS_TOO_SHORT	2008	X'000007D8'
MQRC_CONNECTION_BROKEN	2009	X'000007D9'
MQRC_DATA_LENGTH_ERROR	2010	X'000007DA'
MQRC_DYNAMIC_Q_NAME_ERROR	2011	X'000007DB'
MQRC_ENVIRONMENT_ERROR	2012	X'000007DC'
MQRC_EXPIRY_ERROR	2013	X'000007DD'
MQRC_ERREUR_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_VALEUR_ERREUR	2020	X'000007E4'
MQRC_INT_ATTR_COUNT_ERROR	2021	X'000007E5'
MQRC_INT_ATTR_COUNT_TOO_SMALL	2022	X'000007E6'
MQRC_INT_ATTRS_ARRAY_ERREUR	2023	X'000007E7'
MQRC_SYNCPOINT_LIMIT_REACHED	2024	X'000007E8'
MQRC_MAX_CONNS_LIMIT_ATTEINTE	2025	X'000007E9'
MQRC_MD_ERROR	2026	X'000007EA'
MQRC_MISSING_REPLY_TO_Q	2027	X'000007EB'
MQRC_MSG_TYPE_ERREUR	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'
MQRC_OBJECT_CHANGED	2041	X'000007F9'
MQRC_OBJECT_IN_USE	2042	X'000007FA'
MQRC_OBJECT_TYPE_ERREUR	2043	X'000007FB'
MQRC_ERREUR D'OD_DU	2044	X'000007FC'
MQRC_OPTION_NOT_VALID_FOR_TYPE	2045	X'000007FD'
MQRC_OPTIONS_ERROR	2046	X'000007FE'
MQRC_XX_ENCODE_CASE_ONE err_persistant	2047	X'000007FF'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_PERSISTENT_NOT_ALLOWED	2048	X'00000800'
MQRC_PRIORITY_MAXIMAL_DÉPASSEMENTS	2049	X'00000801'
MQRC_PRIORITY_ERROR	2050	X'00000802'
MQRC_PUT_INHIBÉ	2051	X'00000803'
MQRC_Q_DELETED	2052	X'00000804'
MQRC_Q_FULL	2053	X'00000805'
MQRC_Q_NOT_EMPTY	2055	X'00000807'
MQRC_Q_ESPACE_NON_DISPONIBLE	2056	X'00000808'
MQRC_Q_TYPE_ERREUR	2057	X'00000809'
MQRC_Q_MGR_NAME_ERROR	2058	X'0000080A'
MQRC_Q_MGR_NOT_AVAILABLE	2059	X'0000080B'
MQRC_REPORT_OPTIONS_ERREUR	2061	X'0000080D'
MQRC_SECOND_MARK_NOT_ALLOWED	2062	X'0000080E'
MQRC_SECURITY_ERROR (ERREUR DE SECURITE MQ)	2063	X'0000080F'
MQRC_SELECTOR_COUNT_ERREUR	2065	X'00000811'
MQRC_SELECTOR_LIMIT_EXCEEDED	2066	X'00000812'
MQRC_SELECTOR_ERREUR	2067	X'00000813'
MQRC_SELECTOR_NOT_FOR_TYPE	2068	X'00000814'
MQRC_SIGNAL_OUTSTANDING	2069	X'00000815'
MQRC_SIGNAL_REQUEST_ACCEPTED	2070	X'00000816'
MQRC_STORAGE_NOT_AVAILABLE	2071	X'00000817'
MQRC_SYNCPOINT_NOT_AVAILABLE	2072	X'00000818'
MQRC_TRIGGER_CONTROL_ERROR	2075	X'0000081B'
MQRC_TRIGGER_DEPTH_ERREUR	2076	X'0000081C'
MQRC_TRIGGER_MSG_PRIORITY_ERR	2077	X'0000081D'
MQRC_TRIGGER_TYPE_ERREUR	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_NOM_OBJET	2085	X'00000825'
MQRC_UNKNOWN_OBJET_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'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_CONTEXT_HANDLE_ERREUR	2097	X'00000831'
MQRC_CONTEXT_NOT_AVAILABLE	2098	X'00000832'
MQRC_SIGNAL1_ERROR	2099	X'00000833'
MQRC_OBJECT_ALREADY_EXISTS	2100	X'00000834'
MQRC_OBJECT_DAMAGED	2101	X'00000835'
MQRC_RESOURCE_PROBLEM	2102	X'00000836'
MQRC_ANOTHER_Q_MGR_CONNECTED	2103	X'00000837'
MQRC_UNKNOWN_REPORT_OPTION	2104	X'00000838'
MQRC_STORAGE_CLASS_ERROR	2105	X'00000839'
MQRC_COD_NOT_VALID_FOR_XCF_Q	2106	X'0000083A'
MQRC_XWAIT_ANNULE	2107	X'0000083B'
MQRC_XWAIT_ERREUR	2108	X'0000083C'
MQRC_SUPPRESSED_BY_EXIT	2109	X'0000083D'
MQRC_FORMAT_ERROR	2110	X'0000083E'
MQRC_SOURCE_CCSID_ERROR	2111	X'0000083F'
MQRC_SOURCE_INTEGER_ENC_ERROR	2112	X'00000840'
MQRC_SOURCE_DECIMAL_ENC_ERROR	2113	X'00000841'
MQRC_SOURCE_FLOAT_ENC_ERROR	2114	X'00000842'
MQRC_TARGET_CCSID_ERROR	2115	X'00000843'
MQRC_TARGET_INTEGER_ENC_ERROR	2116	X'00000844'
MQRC_TARGET_DECIMAL_ENC_ERROR	2117	X'00000845'
MQRC_TARGET_FLOAT_ENC_ERROR	2118	X'00000846'
MQRC_NOT_CONVERTED	2119	X'00000847'
MQRC_CONVERTED_MSG_TOO_BIG	2120	X'00000848'
MQRC_TRONQUÉ	2120	X'00000848'
MQRC_NO_EXTERNAL_PARTICIPANTS	2121	X'00000849'
MQRC_PARTICIPANT_NOT_XX_ENCODE_CASE_ONE disponible	2122	X'0000084A'
MQRC_OUTCOME_MIXTE	2123	X'0000084B'
MQRC_OUTCOME_EN attente	2124	X'0000084C'
MQRC_BRIDGE_STARTED	2125	X'0000084D'
MQRC_BRIDGE_STOPPED	2126	X'0000084E'
MQRC_ADAPTER_STOCKAGE_RUPTURE	2127	X'0000084F'
MQRC_UOW_EN_COURS	2128	X'00000850'
MQRC_ADAPTER_CONN_LOAD_ERROR	2129	X'00000851'
MQRC_ADAPTER_SERV_LOAD_ERROR	2130	X'00000852'
MQRC_ADAPTER_DEFS_ERREUR	2131	X'00000853'
MQRC_ADAPTER_DEFS_LOAD_ERROR	2132	X'00000854'
MQRC_ADAPTER_CONV_LOAD_ERROR	2133	X'00000855'
MQRC_BO_ERREUR	2134	X'00000856'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_DH_ERREUR	2135	X'00000857'
MQRC_MULTIPLE_MOTIFS	2136	X'00000858'
MQRC_OPEN_FAILED	2137	X'00000859'
MQRC_ADAPTER_DISC_LOAD_ERROR	2138	X'0000085A'
MQRC_CNO_ERREUR	2139	X'0000085B'
MQRC_CICS_WAIT_FAILED	2140	X'0000085C'
MQRC_DLH_ERREUR	2141	X'0000085D'
MQRC_HEADER_ERREUR	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_ERREUR	2148	X'00000864'
MQRC_PCF_ERREUR	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_Présentateur-ERREUR	2154	X'0000086A'
MQRC_OBJECT_RECORDS_ERROR	2155	X'0000086B'
MQRC_RESPONSE_RECORDS_ERROR	2156	X'0000086C'
MQRC_ASID_MISMATCH	2157	X'0000086D'
MQRC_PMO_RECORD_FLAGS_ERROR	2158	X'0000086E'
MQRC_PUT_MSG_RECORDS_ERROR	2159	X'0000086F'
MQRC_CONN_ID_IN_USE	2160	X'00000870'
MQRC_Q_MGR_QUIESCING	2161	X'00000871'
MQRC_Q_MGR_STOPPING	2162	X'00000872'
MQRC_DUPLICATE_RECOV_COORD	2163	X'00000873'
MQRC_PMO_ERREUR	2173	X'0000087D'
MQRC_API_EXIT_NOT_FOUND	2182	X'00000886'
MQRC_API_EXIT_LOAD_ERROR	2183	X'00000887'
MQRC_REMOTE_Q_NAME_ERROR	2184	X'00000888'
MQRC_INCONSISTENT_PERSISTENCE	2185	X'00000889'
MQRC_GMO_ERROR	2186	X'0000088A'
MQRC_CICS_BRIDGE_RESTRICTION	2187	X'0000088B'
MQRC_STOPPED_BY_CLUSTER_EXIT	2188	X'0000088C'
Erreur de résolution MQRC_CLUSTER_RESOLUTION_ERROR	2189	X'0000088D'
MQRC_CONVERTED_STRING_TOO_BIG	2190	X'0000088E'
MQRC_TMC_ERREUR	2191	X'0000088F'
MQRC_PAGESET_FULL	2192	X'00000890'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_STORAGE_MEDIUM_FULL	2192	X'00000890'
MQRC_PAGESET_ERROR	2193	X'00000891'
MQRC_NAME_NOT_VALID_FOR_TYPE	2194	X'00000892'
MQRC_UNEXPECTED_ERROR	2195	X'00000893'
MQRC_UNKNOWN_XMIT_Q	2196	X'00000894'
MQRC_UNKNOWN_DEF_XMIT_Q	2197	X'00000895'
MQRC_DEF_XMIT_Q_TYPE_ERREUR	2198	X'00000896'
MQRC_DEF_XMIT_Q_USAGE_ERROR	2199	X'00000897'
MQRC_MSG_MARKED_BROWSE_CO_OP	2200	X'00000898'
MQRC_NAME_IN_USE	2201	X'00000899'
MQRC_CONNECTION_QUIESCING	2202	X'0000089A'
MQRC_CONNECTION_STOPPING	2203	X'0000089B'
MQRC_ADAPTER_NOT_AVAILABLE	2204	X'0000089C'
MQRC_MSG_ID_ERROR	2206	X'0000089E'
MQRC_CORREL_ID_ERROR	2207	X'0000089F'
MQRC_FILE_SYSTEM_ERROR	2208	X'000008A0'
MQRC_NO_MSG_LOCKED	2209	X'000008A1'
MQRC_SOAP_DOTNET_ERROR	2210	X'000008A2'
ERREUR MQRC_SOAP_AXIS_ERROR	2211	X'000008A3'
MQRC_SOAP_URL_ERROR	2212	X'000008A4'
MQRC_FILE_NON_AUDITÉE	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_ERREUR	2220	X'000008AC'
MQRC_Q_MGR_ACTIF	2222	X'000008AE'
MQRC_Q_MGR_NOT_ACTIVE	2223	X'000008AF'
MQRC_Q_DEPTH_ELEVE	2224	X'000008B0'
MQRC_Q_DEPTH_LOW	2225	X'000008B1'
MQRC_Q_SERVICE_INTERVAL_HIGH	2226	X'000008B2'
MQRC_Q_SERVICE_INTERVAL_OK	2227	X'000008B3'
MQRC_RFH_HEADER_FIELD_ERROR	2228	X'000008B4'
MQRC_RAS_PROPERTY_ERROR	2229	X'000008B5'
MQRC_UNIT_OF_WORK_NOT_STARTED	2232	X'000008B8'
MQRC_CHANNEL_AUTO_DEF_OK	2233	X'000008B9'
MQRC_CHANNEL_AUTO_ERREUR PAR DEFAUT	2234	X'000008BA'
MQRC_CFH_ERREUR	2235	X'000008BB'
MQRC_CFIL_ERREUR	2236	X'000008BC'
MQRC_CFIN_ERREUR	2237	X'000008BD'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_CFSL_ERREUR	2238	X'000008BE'
MQRC_CFST_ERREUR	2239	X'000008BF'
MQRC_INCOMPLETE_GROUP	2241	X'000008C1'
MQRC_INCOMPLETE_MSG	2242	X'000008C2'
MQRC_INCONSISTENT_CCSIDS	2243	X'000008C3'
MQRC_INCONSISTENT_ENCODINGS	2244	X'000008C4'
MQRC_INCONSISTENT_UOW	2245	X'000008C5'
MQRC_INVALID_MSG_UNDER_CURSOR	2246	X'000008C6'
MQRC_MATCH_OPTIONS_ERROR	2247	X'000008C7'
MQRC_MDE_ERREUR	2248	X'000008C8'
MQRC_MSG_FLAGS_ERREUR	2249	X'000008C9'
MQRC_MSG_SEQ_NUMBER_ERROR	2250	X'000008CA'
ERREUR MQRC_OFFSET_ERROR	2251	X'000008CB'
MQRC_ORIGINAL_LENGTH_ERROR	2252	X'000008CC'
MQRC_SEGMENT_LONGUEUR_ZÉRO	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_ERREUR	2258	X'000008D2'
MQRC_INCONSISTENT_BROWSE	2259	X'000008D3'
MQRC_XQH_ERREUR	2260	X'000008D4'
MQRC_SRC_ENV_ERREUR	2261	X'000008D5'
MQRC_SRC_NAME_ERROR	2262	X'000008D6'
MQRC_DEST_ENV_ERREUR	2263	X'000008D7'
MQRC_DEST_NAME_ERREUR	2264	X'000008D8'
MQRC_TM_ERREUR	2265	X'000008D9'
MQRC_CLUSTER_EXIT_ERREUR	2266	X'000008DA'
MQRC_CLUSTER_EXIT_LOAD_ERROR	2267	X'000008DB'
MQRC_CLUSTER_PUT_INHIBÉ	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_ERREUR	2273	X'000008E1'
MQRC_OPTION_ENVIRONMENT_ERROR	2274	X'000008E2'
MQRC_CD_ERREUR	2277	X'000008E5'
MQRC_CLIENT_CONN_ERROR	2278	X'000008E6'
MQRC_CHANNEL_STOPPED_BY_USER	2279	X'000008E7'
ERREUR MQRC_HCONFIG_ERROR	2280	X'000008E8'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
Erreur de fonction MQRC_FUNCTION_ERROR	2281	X'000008E9'
MQRC_CHANNEL_STARTED	2282	X'000008EA'
MQRC_CHANNEL_STOPPED	2283	X'000008EB'
MQRC_CHANNEL_CONV_ERROR	2284	X'000008EC'
MQRC_SERVICE_NOT_AVAILABLE	2285	X'000008ED'
MQRC_INITIALIZATION_FAILED	2286	X'000008EE'
Echec de la commande MQRC_TERMINATION_FAILED	2287	X'000008EF'
MQRC_UNKNOWN_NOM_QQ	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_ENTITÉ_AUTH	2293	X'000008F5'
MQRC_UNKNOWN_REF_OBJECT	2294	X'000008F6'
MQRC_CHANNEL_ACTIVATED	2295	X'000008F7'
MQRC_CHANNEL_NOT_ACTIVATED	2296	X'000008F8'
MQRC_UOW_ANNULE	2297	X'000008F9'
MQRC_FUNCTION_NOT_SUPPORTED	2298	X'000008FA'
MQRC_SELECTOR_TYPE_ERREUR	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_ERREUR	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'
ERREUR MQRC_STRING_ERROR	2307	X'00000903'
MQRC_ENCODING_NOT_SUPPORTED	2308	X'00000904'
MQRC_SELECTOR_NOT_PRESENT	2309	X'00000905'
MQRC_OUT_SELECTOR_ERREUR	2310	X'00000906'
MQRC_STRING_TRONQUÉ	2311	X'00000907'
MQRC_SELECTOR_WRONG_TYPE	2312	X'00000908'
MQRC_INCONSISTENT_ITEM_TYPE	2313	X'00000909'
ERREUR MQRC_INDEX_ERREUR	2314	X'0000090A'
MQRC_SYSTEM_BAG_NOT_ALTERABLE	2315	X'0000090B'
MQRC_ITEM_COUNT_ERREUR	2316	X'0000090C'
MQRC_FORMAT_NOT_SUPPORTED	2317	X'0000090D'
MQRC_SELECTOR_NOT_SUPPORTED	2318	X'0000090E'
MQRC_ITEM_VALEUR_ERREUR	2319	X'0000090F'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_HBAG_ERREUR	2320	X'00000910'
MQRC_PARAMETER_MISSING	2321	X'00000911'
MQRC_CMD_SERVER_NOT_AVAILABLE	2322	X'00000912'
MQRC_STRING_LENGTH_ERROR	2323	X'00000913'
MQRC_INQUIRY_COMMAND_ERROR	2324	X'00000914'
MQRC_NESTED_BAG_NOT_SUPPORTED	2325	X'00000915'
TYPE 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_WIH	2332	X'0000091C'
MQRC_WIH_ERREUR	2333	X'0000091D'
MQRC_RFH_ERREUR	2334	X'0000091E'
MQRC_RFH_STRING_ERROR	2335	X'0000091F'
MQRC_RFH_ERREUR DE COMMANDE	2336	X'00000920'
MQRC_RFH_PARM_ERREUR	2337	X'00000921'
MQRC_RFH_DUPLICATE_PARM	2338	X'00000922'
MQRC_RFH_PARM_MISSING	2339	X'00000923'
MQRC_CHAR_CONVERSION_ERREUR	2340	X'00000924'
MQRC_UCS2_CONVERSION_ERROR	2341	X'00000925'
MQRC_DB2_NOT_AVAILABLE	2342	X'00000926'
MQRC_OBJECT_NOT_UNIQUE	2343	X'00000927'
MQRC_CONN_TAG_NOT_RELÂCHÉE	2344	X'00000928'
MQRC_CF_NOT_XX_ENCODE_CASE_ONE disponible	2345	X'00000929'
MQRC_CF_STRUC_IN_USE	2346	X'0000092A'
MQRC_CF_STRUC_LIST_HDR_IN_USE	2347	X'0000092B'
Echec de MQRC_CF_STRUC_AUTH_FAILED	2348	X'0000092C'
MQRC_CF_STRUC_ERREUR	2349	X'0000092D'
MQRC_CONN_TAG_NON_UTILISABLE	2350	X'0000092E'
MQRC_GLOBAL_UOW_CONFLICT	2351	X'0000092F'
MQRC_LOCAL_UOW_CONFLICT	2352	X'00000930'
MQRC_HANDLE_IN_USE_FOR_UOW	2353	X'00000931'
MQRC_UOW_ENLISTMENT_ERROR	2354	X'00000932'
MQRC_UOW_MIX_NOT_SUPPORTED	2355	X'00000933'
MQRC_WXP_ERREUR	2356	X'00000934'
MQRC_CURRENT_RECORD_ERROR	2357	X'00000935'
MQRC_NEXT_OFFSET_ERREUR	2358	X'00000936'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_NO_RECORD_DISPONIBLE	2359	X'00000937'
MQRC_OBJET_NIVEAU_INCOMPATIBLE	2360	X'00000938'
MQRC_NEXT_RECORD_ERREUR	2361	X'00000939'
MQRC_BACKOUT_SEUIL_ATTEINT	2362	X'0000093A'
MQRC_MSG_NON_CORRESPONDANCE	2363	X'0000093B'
MQRC_JMS_FORMAT_ERREUR	2364	X'0000093C'
MQRC_SEGMENTS_NON_PRIS en charge	2365	X'0000093D'
MQRC_WRONG_CF_NIVEAU	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_DEFINI	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_ERREUR	2376	X'00000948'
MQRC_EXIT_MOTIF-ERREUR	2377	X'00000949'
MQRC_RESERVED_VALEUR_ERREUR	2378	X'0000094A'
MQRC_NO_DATA_XX_ENCODE_CASE_ONE disponible	2379	X'0000094B'
MQRC_ERREUR DE SCO_DU	2380	X'0000094C'
MQRC_KEY_REPOSITORY_ERROR	2381	X'0000094D'
MQRC_CRYPTO_ERREUR_MATÉRIEL	2382	X'0000094E'
MQRC_AUTH_INFO_REC_COUNT_ERROR	2383	X'0000094F'
MQRC_AUTH_INFO_REC_ERROR	2384	X'00000950'
ERREUR 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_NOM_UTILISATEUR_LENGTH_ERR	2389	X'00000955'
MQRC_LDAP_PASSWORD_ERROR	2390	X'00000956'
MQRC_SSL_ALREADY_INITIALISÉ	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_ERREUR	2395	X'0000095B'
MQRC_SSL_NOT_ALLOWED	2396	X'0000095C'
MQRC_JSSE_ERREUR	2397	X'0000095D'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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_RÉVOQUÉ	2401	X'00000961'
MQRC_SSL_CERT_STORE_ERREUR	2402	X'00000962'
MQRC_CLIENT_EXIT_LOAD_ERROR	2406	X'00000966'
MQRC_CLIENT_EXIT_ERREUR	2407	X'00000967'
MQRC_UOW_COMMITTED	2408	X'00000968'
MQRC_SSL_KEY_RESET_ERROR	2409	X'00000969'
MQRC_UNKNOWN_NOM_COMPOSANT	2410	X'0000096A'
STATUT DU GESTIONNAIRE DE FILES D'ATTENTE	2411	X'0000096B'
MQRC_COMMAND_MQSC	2412	X'0000096C'
MQRC_COMMAND_PCF	2413	X'0000096D'
MQRC_CFIF_ERREUR	2414	X'0000096E'
MQRC_CFSF_ERREUR	2415	X'0000096F'
MQRC_CFGR_ERREUR	2416	X'00000970'
MQRC_MSG_NOT_ALLOWED_IN_GROUP	2417	X'00000971'
MQRC_FILTER_OPERATOR_ERROR	2418	X'00000972'
MQRC_NESTED_SELECTOR_ERREUR	2419	X'00000973'
MQRC_EPH_ERREUR	2420	X'00000974'
MQRC_RFH_FORMAT_ERREUR	2421	X'00000975'
MQRC_CFBF_ERREUR	2422	X'00000976'
MQRC_CLIENT_CHANNEL_CONFLICT	2423	X'00000977'
MQRC_SD_ERREUR	2424	X'00000978'
MQRC_TOPIC_STRING_ERROR	2425	X'00000979'
MQRC_STS_ERREUR	2426	X'0000097A'
MQRC_NO_ABONNEMENT	2428	X'0000097C'
MQRC_SUBSCRIPTION_IN_USE	2429	X'0000097D'
MQRC_STAT_TYPE_ERREUR	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_ERREUR	2435	X'00000983'
MQRC_DURABILITY_NOT_ALLOWED	2436	X'00000984'
MQRC_NO_RETAINED_MSG	2437	X'00000985'
MQRC_SRO_ERREUR	2438	X'00000986'
MQRC_SUB_NAME_ERREUR	2440	X'00000988'
MQRC_OBJECT_STRING_ERROR	2441	X'00000989'
MQRC_PROPERTY_NAME_ERROR	2442	X'0000098A'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_SEGMENTATION_NOT_ALLOWED	2443	X'0000098B'
MQRC_CBD_ERREUR	2444	X'0000098C'
MQRC_CTLO_ERREUR	2445	X'0000098D'
MQRC_NO_CALLBACKS_ACTIVE	2446	X'0000098E'
MQRC_CALLBACK_NOT_REGISTERED	2448	X'00000990'
MQRC_OPTIONS_CHANGED	2457	X'00000999'
MQRC_READ_AHEAD_MSGS	2458	X'0000099A'
MQRC_SELECTOR_SYNTAX_ERREUR	2459	X'0000099B'
ERREUR MQRC_HMSG_ERROR	2460	X'0000099C'
ERREUR MQRC_CMHO_ERROR	2461	X'0000099D'
MQRC_DMHO_ERREUR	2462	X'0000099E'
MQRC_SMPO_ERREUR	2463	X'0000099F'
MQRC_IMPO_ERREUR	2464	X'000009A0'
MQRC_PROPERTY_NAME_TOO_BIG	2465	X'000009A1'
MQRC_PROP_VALEUR_NOT_CONVERTED	2466	X'000009A2'
MQRC_PROP_TYPE_NON_PRIS en charge	2467	X'000009A3'
MQRC_PROPERTY_VALEUR_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_RETENU	2479	X'000009AF'
MQRC_ALIAS_TARGTYPE_CHANGED	2480	X'000009B0'
MQRC_DMPO_ERREUR	2481	X'000009B1'
MQRC_PD_ERREUR	2482	X'000009B2'
MQRC_CALLBACK_TYPE_ERREUR	2483	X'000009B3'
MQRC_CBD_OPTIONS_ERREUR	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'
ERREUR D'OPERATION_MQRC	2488	X'000009B8'
MQRC_BMHO_ERREUR	2489	X'000009B9'
MQRC_UNSUPPORTED_PROPERTY	2490	X'000009BA'
MQRC_PROP_NOM_NON_CONVERTI	2492	X'000009BC'
MQRC_GET_ENABLED	2494	X'000009BE'
MQRC_MODULE_NOT_FOUND	2495	X'000009BF'
MQRC_MODULE_INVALID	2496	X'000009C0'
MQRC_MODULE_ENTRY_NOT_FOUND	2497	X'000009C1'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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_ERREUR	2501	X'000009C5'
MQRC_PUBLICATION_FAILURE	2502	X'000009C6'
MQRC_SUB_INHIBÉ	2503	X'000009C7'
MQRC_SELECTOR_ALWAYS_FALSE	2504	X'000009C8'
MQRC_XEPO_ERREUR	2507	X'000009CB'
MQRC_DURATIONITY_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_GROUPE_SUB	2514	X'000009D2'
MQRC_GROUPING_NOT_ALTERABLE	2515	X'000009D3'
MQRC_SELECTOR_INVALID_FOR_TYPE	2516	X'000009D4'
MQRC_HOBJ_MISE au repos	2517	X'000009D5'
MQRC_HOBJ_MSGS mis au repos	2518	X'000009D6'
MQRC_SELECTION_STRING_ERROR	2519	X'000009D7'
MQRC_RES_OBJECT_STRING_ERROR	2520	X'000009D8'
MQRC_CONNECTION_SUSPENDUE	2521	X'000009D9'
MQRC_INVALID_DESTINATION	2522	X'000009DA'
MQRC_INVALID_ABONNEMENT	2523	X'000009DB'
MQRC_SELECTOR_NOT_ALTERABLE	2524	X'000009DC'
MQRC_RETAINED_MSG_Q_ERREUR	2525	X'000009DD'
MQRC_RETAINED_NOT_LIVRÉ	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_INHIBÉ	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'
ERREUR 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_NOM_CANAL	2540	X'000009EC'
MQRC_LOOPING_PUBLICATION	2541	X'000009ED'

Tableau 315. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRC_ALREADY_JOINT	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_ERREUR	2592	X'00000A20'
MQRC_REOPEN_EXCL_INPUT_ERROR	6100	X'000017D4'
MQRC_REOPEN_INQUIRE_ERROR	6101	X'000017D5'
MQRC_REOPEN_SAVED_CONTEXT_ERR	6102	X'000017D6'
MQRC_REOPEN_TEMPORARY_Q_ERROR	6103	X'000017D7'
MQRC_ATTRIBUT_LOCKED	6104	X'000017D8'
MQRC_CURSOR_NOT_VALID	6105	X'000017D9'
ERREUR_CODAGE_MQRC_ERREUR	6106	X'000017DA'
MQRC_STRUC_ID_ERROR	6107	X'000017DB'
POINT_NULL_MQRC_NULL	6108	X'000017DC'
MQRC_NO_RÉFÉRENCE_CONNEXION	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_INSUFFISENT_BUFFER	6113	X'000017E1'
MQRC_DONNÉES_INSUFFISANTES	6114	X'000017E2'
MQRC_DATA_TRONQUÉ	6115	X'000017E3'
LONGUEUR_ZÉRO_MQRC_	6116	X'000017E4'
LONGUEUR_NÉGATIVE_MQRC	6117	X'000017E5'
DÉCALAGE_NÉGATIVE_MQRC	6118	X'000017E6'
MQRC_INCONSISTENT_FORMAT	6119	X'000017E7'
MQRC_INCONSISTENT_OBJECT_STATE	6120	X'000017E8'
MQRC_CONTEXT_OBJET_NON_VALIDE	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'
VERSION MQRC_WRONG_VERSION	6128	X'000017F0'
ERREUR MQRC_REFERENCE_ERROR	6129	X'000017F1'

MQRCCF_ * (codes anomalie d'en-tête de format de commande)

Tableau 316. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_CFH_TYPE_ERROR	3001	X'00000BB9'
MQRCCF_CFH_LENGTH_ERROR	3002	X'00000BBA'
MQRCCF_CFH_VERSION_ERROR	3003	X'00000BBB'
MQRCCF_CFH_MSG_SEQ_NUMBER_ERR	3004	X'00000BBC'
MQRCCF_CFH_CONTROL_ERROR	3005	X'00000BBD'
MQRCCF_CFH_PARM_COUNT_ERROR	3006	X'00000BBE'
MQRCCF_CFH_COMMAND_ERROR	3007	X'00000BBF'
Echec de la commande MQRCCF_COMMAND_FAILED	3008	X'00000BC0'
MQRCCF_CFIN_LENGTH_ERROR	3009	X'00000BC1'
MQRCCF_CFST_LENGTH_ERROR	3010	X'00000BC2'
MQRCCF_CFST_STRING_LENGTH_ERR	3011	X'00000BC3'
MQRCCF_FORCE_VALEUR_ERREUR	3012	X'00000BC4'
MQRCCF_STRUCTURE_TYPE_ERROR	3013	X'00000BC5'
MQRCCF_CFIN_PARM_ID_ERROR	3014	X'00000BC6'
MQRCCF_CFST_PARM_ID_ERROR	3015	X'00000BC7'
MQRCCF_MSG_LENGTH_ERROR	3016	X'00000BC8'
MQRCCF_CFIN_DUPLICATE_PARM	3017	X'00000BC9'
MQRCCF_CFST_DUPLICATE_PARM	3018	X'00000BCA'
MQRCCF_PARM_COUNT_TOO_SMALL	3019	X'00000BCB'
MQRCCF_COMPTE_PARM_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'
MQRCCF_CFSL_LENGTH_ERROR	3024	X'00000BD0'
MQRCCF_REPLACE_VALEUR_ERREUR	3025	X'00000BD1'
MQRCCF_CFIL_DUPLICATE_VALUE	3026	X'00000BD2'
MQRCCF_CFIL_COUNT_ERROR	3027	X'00000BD3'
MQRCCF_CFIL_LENGTH_ERROR	3028	X'00000BD4'
MQRCCF_QUIESCE_VALEUR_ERREUR	3029	X'00000BD5'
MQRCCF_MODE_VALEUR_ERREUR	3029	X'00000BD5'
MQRCCF_MSG_SEQ_NUMBER_ERROR	3030	X'00000BD6'
MQRCCF_PING_DATA_COUNT_ERROR	3031	X'00000BD7'
MQRCCF_PING_DATA_COMPARE_ERROR	3032	X'00000BD8'
MQRCCF_CFSL_PARM_ID_ERROR	3033	X'00000BD9'
MQRCCF_CHANNEL_TYPE_ERROR	3034	X'00000BDA'
MQRCCF_PARM_SEQUENCE_ERROR	3035	X'00000BDB'
MQRCCF_XMIT_PROTOCOL_TYPE_ERR	3036	X'00000BDC'
MQRCCF_TAILLE_BATCH_ERREUR	3037	X'00000BDD'

Tableau 316. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
ERREUR MQRCCF_DISC_INT_ERROR	3038	X'00000BDE'
MQRCCF_SHORT_RETRY_ERROR	3039	X'00000BDF'
MQRCCF_SHORT_TIMER_ERROR	3040	X'00000BE0'
MQRCCF_LONG_RETRY_ERROR	3041	X'00000BE1'
MQRCCF_LONG_TIMER_ERREUR	3042	X'00000BE2'
MQRCCF_SEQ_NUMBER_WRAP_ERROR	3043	X'00000BE3'
MQRCCF_MAX_MSG_LENGTH_ERROR	3044	X'00000BE4'
MQRCCF_PUT_AUTH_ERROR	3045	X'00000BE5'
MQRCCF_PURGE_VALEUR_ERREUR	3046	X'00000BE6'
MQRCCF_CFIL_PARM_ID_ERROR	3047	X'00000BE7'
MQRCCF_MSG_TRUNCATED	3048	X'00000BE8'
ERREUR MQRCCF_CCSID	3049	X'00000BE9'
MQRCCF_ERREUR_CODAGE	3050	X'00000BEA'
MQRCCF_QUEUES_VALUE_ERROR	3051	X'00000BEB'
MQRCCF_DATA_CONV_VALEUR_ERREUR	3052	X'00000BEC'
MQRCCF_INDOUBT_VALEUR_ERREUR	3053	X'00000BED'
MQRCCF_ESCAPE_TYPE_ERROR	3054	X'00000BEE'
MQRCCF_REPOS_VALEUR_ERREUR	3055	X'00000BEF'
MQRCCF_CHANNEL_TABLE_ERROR	3062	X'00000BF6'
MQRCCF_MCA_TYPE_ERROR	3063	X'00000BF7'
MQRCCF_CHL_INST_TYPE_ERROR	3064	X'00000BF8'
MQRCCF_CHL_STATUS_NOT_FOUND	3065	X'00000BF9'
MQRCCF_CFSL_DUPLICATE_PARM	3066	X'00000BFA'
MQRCCF_CFSL_TOTAL_LENGTH_ERROR	3067	X'00000BFB'
MQRCCF_CFSL_COUNT_ERROR	3068	X'00000BFC'
MQRCCF_CFSL_STRING_LENGTH_ERR	3069	X'00000BFD'
MQRCCF_BROKER_DELETED	3070	X'00000BFE'
MQRCCF_FLOT_ERREUR	3071	X'00000BFF'
MQRCCF_TOPIC_ERREUR	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_NOM_ERREUR	3076	X'00000C04'
MQRCCF_NO_RETAINED_MSG	3077	X'00000C05'
MQRCCF_IDENTITÉ_DOUBLON	3078	X'00000C06'
MQRCCF_INCORRECT_Q	3079	X'00000C07'
MQRCCF_CORREL_ID_ERREUR	3080	X'00000C08'
MQRCCF_NON_AUTORISÉ	3081	X'00000C09'
MQRCCF_FLUX_INCONNU	3082	X'00000C0A'

Tableau 316. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_REG_OPTIONS_ERREUR	3083	X'00000C0B'
MQRCCF_PUB_OPTIONS_ERROR	3084	X'00000C0C'
MQRCCF_COURTIER inconnu	3085	X'00000C0D'
MQRCCF_Q_MGR_CCSID_ERROR	3086	X'00000C0E'
MQRCCF_DEL_OPTIONS_ERROR	3087	X'00000C0F'
MQRCCF_CLUSTER_NAME_CONFLICT	3088	X'00000C10'
MQRCCF_NOM_RÉFÉRENTIEL_CONFLIT	3089	X'00000C11'
MQRCCF_CLUSTER_Q_USAGE_ERROR	3090	X'00000C12'
MQRCCF_ACTION_VALEUR_ERREUR	3091	X'00000C13'
MQRCCF_COMMS_LIBRARY_ERROR	3092	X'00000C14'
MQRCCF_NETBIOS_NAME_ERROR	3093	X'00000C15'
Echec de la commande MQRCCF_BROKER_COMMAND_FAILED	3094	X'00000C16'
MQRCCF_CFST_CONFLICTING_PARM	3095	X'00000C17'
MQRCCF_CHEMIN_NON_VALIDE	3096	X'00000C18'
MQRCCF_PARM_SYNTAX_ERROR	3097	X'00000C19'
MQRCCF_PWD_LENGTH_ERROR	3098	X'00000C1A'
MQRCCF_FILTER_ERREUR	3150	X'00000C4E'
MQRCCF_UTILISATEUR_FAUT_ERREUR	3151	X'00000C4F'
MQRCCF_ABONNEMENT_DOUBLON	3152	X'00000C50'
MQRCCF_SUB_NAME_ERREUR	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_JOINT	3157	X'00000C55'
MQRCCF_OBJET_EN_UTILISATION	3160	X'00000C58'
MQRCCF_NOM_FICHIER_INCONNU	3161	X'00000C59'
MQRCCF_FILE_NON_DISPONIBLE	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_NUMÉRO_PORT_ERREUR	3167	X'00000C5F'
MQRCCF_CHL_SYSTEM_NOT_ACTIVE	3168	X'00000C60'
MQRCCF_ENTITY_NAME_MISSING	3169	X'00000C61'
MQRCCF_NOM_PROFIL_ERROR	3170	X'00000C62'
MQRCCF_AUTH_VALEUR_ERREUR	3171	X'00000C63'
MQRCCF_VALEUR_AUTH_MANQUANTE	3172	X'00000C64'
MQRCCF_TYPE_OBJET_MANQUANT	3173	X'00000C65'
MQRCCF_CONNECTION_ID_ERROR	3174	X'00000C66'

Tableau 316. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_LOG_TYPE_ERROR	3175	X'00000C67'
MQRCCF_PROGRAM_NON_DISPONIBLE	3176	X'00000C68'
MQRCCF_PROGRAM_AUTH_FAILED	3177	X'00000C69'
MQRCCF_NON_FOUND	3200	X'00000C80'
MQRCCF_SECURITY_SWITCH_OFF	3201	X'00000C81'
MQRCCF_SECURITY_REFRESH_FAILED	3202	X'00000C82'
MQRCCF_PARM_CONFLICT	3203	X'00000C83'
MQRCCF_COMMANDE_INHIBÉE	3204	X'00000C84'
MQRCCF_OBJET_BEING_DELETED	3205	X'00000C85'
MQRCCF_STORAGE_CLASS_IN_USE	3207	X'00000C87'
MQRCCF_NOM_OBJET_RESTREINT	3208	X'00000C88'
MQRCCF_OBJET_LIMITE_EXCEEDED	3209	X'00000C89'
MQRCCF_OBJET_OPEN_FORCE	3210	X'00000C8A'
MQRCCF_DISPOSITION_CONFLICT	3211	X'00000C8B'
MQRCCF_Q_MGR_NOT_IN_QSG	3212	X'00000C8C'
MQRCCF_ATTR_VALEUR_FIXE	3213	X'00000C8D'
MQRCCF_NOMELIST_ERREUR	3215	X'00000C8F'
MQRCCF_NO_CHANNEL_INITIATOR	3217	X'00000C91'
MQRCCF_CHANNEL_INITIATOR_ERROR	3218	X'00000C92'
MQRCCF_COMMAND_NIVEAU_CONFLIT	3222	X'00000C96'
MQRCCF_Q_ATTR_CONFLIT	3223	X'00000C97'
MQRCCF_EVENTS_DISABLED	3224	X'00000C98'
MQRCCF_COMMAND_SCOPE_ERROR	3225	X'00000C99'
MQRCCF_COMMAND_REPLY_ERROR	3226	X'00000C9A'
MQRCCF_FONCTION_RESTREINTE	3227	X'00000C9B'
MQRCCF_PARM_MISSING	3228	X'00000C9C'
MQRCCF_PARM_VALEUR_ERREUR	3229	X'00000C9D'
MQRCCF_COMMAND_LENGTH_ERROR	3230	X'00000C9E'
MQRCCF_COMMANDE_ORIGINE_ERREUR	3231	X'00000C9F'
MQRCCF_LISTENER_CONFLICT	3232	X'00000CA0'
MQRCCF_LISTENER_STARTED	3233	X'00000CA1'
MQRCCF_LISTENER_STOPPED	3234	X'00000CA2'
MQRCCF_CHANNEL_ERROR	3235	X'00000CA3'
MQRCCF_CF_STRUC_ERROR	3236	X'00000CA4'
MQRCCF_ID_UTILISATEUR_INCONNU	3237	X'00000CA5'
MQRCCF_ERREUR_INATTENDU	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'

Tableau 316. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
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_ERREUR_OPERATEUR	3246	X'00000CAE'
MQRCCF_CFSF_PARM_ID_ERROR	3247	X'00000CAF'
MQRCCF_TOO_MANY_FILTERS	3248	X'00000CB0'
MQRCCF_LISTENER_RUNNING	3249	X'00000CB1'
MQRCCF_LSTR_STATUS_NOT_FOUND	3250	X'00000CB2'
MQRCCF_SERVICE_RUNNING	3251	X'00000CB3'
MQRCCF_SERV_STATUS_NOT_FOUND	3252	X'00000CB4'
MQRCCF_SERVICE_STOPPED	3253	X'00000CB5'
MQRCCF_CFBS_DUPLICATE_PARM	3254	X'00000CB6'
MQRCCF_CFBS_LENGTH_ERROR	3255	X'00000CB7'
MQRCCF_CFBS_PARM_ID_ERROR	3256	X'00000CB8'
MQRCCF_CFBS_STRING_LENGTH_ERR	3257	X'00000CB9'
MQRCCF_CFGR_LENGTH_ERROR	3258	X'00000CBA'
MQRCCF_CFGR_PARM_COUNT_ERROR	3259	X'00000CBB'
MQRCCF_CONN_NOT_STOPPED	3260	X'00000CBC'
MQRCCF_SERVICE_REQUEST_PENDING	3261	X'00000CBD'
MQRCCF_NO_START_CMD	3262	X'00000CBE'
MQRCCF_NO_STOP_CMD	3263	X'00000CBF'
MQRCCF_CFBF_LENGTH_ERROR	3264	X'00000CC0'
MQRCCF_CFBF_PARM_ID_ERROR	3265	X'00000CC1'
MQRCCF_CFBF_OPERATOR_ERROR	3266	X'00000CC2'
MQRCCF_CFBF_FILTER_VAL_LEN_ERR	3267	X'00000CC3'
MQRCCF_LISTENER_STILL_ACTIF	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'
MQRCCF_SUB_ALREADY_EXISTS	3311	X'00000CEF'
MQRCCF_NOM_OBJET_INCONNU	3312	X'00000CF0'

Tableau 316. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_REMOTE_Q_NAME_ERROR	3313	X'00000CF1'
MQRCCF_DURABILITY_NOT_ALLOWED	3314	X'00000CF2'
MQRCCF_HOBJ_ERREUR	3315	X'00000CF3'
MQRCCF_DEST_NAME_ERROR	3316	X'00000CF4'
MQRCCF_DESTINATION_NON_VALIDE	3317	X'00000CF5'
MQRCCF_PUBSUB_INHIBÉ	3318	X'00000CF6'
MQRCCF_CHLAUTH_TYPE_ERREUR	3326	X'00000CFE'
MQRCCF_CHLAUTH_ACTION_ERROR	3327	X'00000CFF'
MQRCCF_CHLAUTH_USERSRC_ERROR	3335	X'00000D07'
MQRCCF_TYPE_AUTH	3336	X'00000D08'
MQRCCF_CHLAUTH_ALREADY_EXISTS	3337	X'00000D09'
MQRCCF_CHLAUTH_NOT_FOUND	3338	X'00000D0A'
MQRCCF_FAUT_CHLAUTH_ACTION	3339	X'00000D0B'
MQRCCF_FAUT_CHLAUTH_USERSRC	3340	X'00000D0C'
MQRCCF_CHLAUTH_WARN_ERREUR	3341	X'00000D0D'
MQRCCF_CONCORDANCE-erreur-chlorauth_correspondance	3342	X'00000D0E'
MQRCCF_IPADDR_RANGE_CONFLIT	3343	X'00000D0F'
MQRCCF_CHLAUTH_MAX_DEPASSE	3344	X'00000D10'
MQRCCF_IPADDR_ERREUR	3345	X'00000D11'
MQRCCF_IPADDR_RANGE_ERROR	3346	X'00000D12'
MQRCCF_NOM_PROFIL_MANQUANT	3347	X'00000D13'
MQRCCF_CHLAUTH_CLNTUSER_ERROR	3348	X'00000D14'
MQRCCF_CHLAUTH_NOM_ERREUR	3349	X'00000D15'
MQRCCF_SUITE_B_ERREUR	3353	X'00000D19'
MQRCCF_PSCLUS_DISABLED_TOPDEF	3359	X'00000D1F'
MQRCCF_PSCLUS_TOPIC_EXISTS	3360	X'00000D20'
MQRCCF_OBJET_ALREADY_EXISTS	4001	X'00000FA1'
MQRCCF_OBJET_TYPE_ERREUR	4002	X'00000FA2'
MQRCCF_LIKE_OBJECT_MAL_TYPE	4003	X'00000FA3'
MQRCCF_OBS_OPEN	4004	X'00000FA4'
MQRCCF_ATTR_VALEUR_ERREUR	4005	X'00000FA5'
MQRCCF_UNKNOWN_Q_MGR	4006	X'00000FA6'
MQRCCF_Q_TYPE_ERREUR	4007	X'00000FA7'
MQRCCF_OBJECT_NAME_ERROR	4008	X'00000FA8'
MQRCCF_ALLOCATE_FAILED	4009	X'00000FA9'
MQRCCF_HÔTE_NON_DISPONIBLE	4010	X'00000FAA'
MQRCCF_ERREUR de configuration	4011	X'00000FAB'
MQRCCF_CONNECTION_REFUSE	4012	X'00000FAC'
MQRCCF_ENTRY_ERROR	4013	X'00000FAD'

Tableau 316. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
Echec de MQRCCF_SEND_FAILED	4014	X'00000FAE'
MQRCCF_ERREUR_DONNÉES_REÇUES	4015	X'00000FAF'
Echec de MQRCCF_RECEIVE_FAILED	4016	X'00000FB0'
MQRCCF_CONNEXION_FERMÉ	4017	X'00000FB1'
MQRCCF_NO_STORAGE	4018	X'00000FB2'
MQRCCF_NO_COMMS_MANAGER	4019	X'00000FB3'
MQRCCF_LISTENER_NOT_STARTED	4020	X'00000FB4'
Echec de MQRCCF_BIND_FAILED	4024	X'00000FB8'
MQRCCF_CHANNEL_EN attente de validation	4025	X'00000FB9'
MQRCCF_MQCONN_FAILED	4026	X'00000FBA'
MQRCCF_MQOPEN_FAILED	4027	X'00000FBB'
MQRCCF_MQGET_FAILED	4028	X'00000FBC'
Echec de MQRCCF_MQPUT_FAILED	4029	X'00000FBD'
MQRCCF_ERREUR_PING_ERROR	4030	X'00000FBE'
MQRCCF_CHANNEL_IN_USE	4031	X'00000FBF'
MQRCCF_CHANNEL_NOT_FOUND	4032	X'00000FC0'
MQRCCF_UNKNOWN_CANAL_DISTANT	4033	X'00000FC1'
MQRCCF_REMOTE_QM_UNAVAILABLE	4034	X'00000FC2'
MQRCCF_REMOTE_QM_TERMINAISON	4035	X'00000FC3'
MQRCCF_MQINQ_ÉCHEC	4036	X'00000FC4'
MQRCCF_NOT_XMIT_Q	4037	X'00000FC5'
MQRCCF_CHANNEL_DISABLED	4038	X'00000FC6'
MQRCCF_USER_EXIT_NOT_AVAILABLE	4039	X'00000FC7'
Echec de MQRCCF_COMMIT_FAILED	4040	X'00000FC8'
MQRCCF_TYPE_CANAL_INJUSTE	4041	X'00000FC9'
MQRCCF_CHANNEL_ALREADY_EXISTS	4042	X'00000FCA'
MQRCCF_DONNÉES_TOO_LARGE	4043	X'00000FCB'
MQRCCF_CHANNEL_NAME_ERROR	4044	X'00000FCC'
MQRCCF_XMIT_Q_NAME_ERROR	4045	X'00000FCD'
MQRCCF_MCA_NAME_ERROR	4047	X'00000FCF'
MQRCCF_SEND_EXIT_NAME_ERROR	4048	X'00000FD0'
MQRCCF_SEC_EXIT_NAME_ERROR	4049	X'00000FD1'
MQRCCF_MSG_EXIT_NAME_ERROR	4050	X'00000FD2'
MQRCCF_RCV_NOM_EXIT	4051	X'00000FD3'
MQRCCF_XMIT_Q_NAME_MAL_TYPE	4052	X'00000FD4'
MQRCCF_MCA_NOM_TYPE_ERREUR	4053	X'00000FD5'
MQRCCF_DISC_INT_TYPE	4054	X'00000FD6'
MQRCCF_SHORT_RETRY_MAL_TYPE	4055	X'00000FD7'
MQRCCF_SHORT_TIMER_MAL_TYPE	4056	X'00000FD8'

Tableau 316. Valeurs des constantes (suite)

Nom	Valeur décimale	Valeur hexadécimale
MQRCCF_LONG_RETRY_TYPE	4057	X'00000FD9'
MQRCCF_LONG_TIMER_TYPE	4058	X'00000FDA'
MQRCCF_PUT_AUTH_TYPE_ERREUR	4059	X'00000FDB'
MQRCCF_KEEP_ALIVE_INT_ERROR	4060	X'00000FDC'
MQRCCF_NOM_CONN_MANQUANT	4061	X'00000FDD'
MQRCCF_CONN_NAME_ERROR	4062	X'00000FDE'
MQRCCF_MQSET_FAILED	4063	X'00000FDF'
MQRCCF_CHANNEL_NOT_ACTIVE	4064	X'00000FE0'
MQRCCF_TERMINÉ_PAR_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_ERREUR	4073	X'00000FE9'
MQRCCF_MR_INTERVAL_TYPE_ERREUR	4074	X'00000FEA'
MQRCCF_NPM_SPEED_ERROR	4075	X'00000FEB'
MQRCCF_NPM_SPEED_INJUSTEMENT _type	4076	X'00000FEC'
ERREUR MQRCCF_HB_INTERVAL_ERROR	4077	X'00000FED'
MQRCCF_HB_INTERVAL_TYPE_ERREUR	4078	X'00000FEE'
MQRCCF_CHAD_ERREUR	4079	X'00000FEF'
MQRCCF_CHAD_TYPE_ERREUR	4080	X'00000FF0'
ERREUR MQRCCF_CHAD_EVENT_ERROR	4081	X'00000FF1'
MQRCCF_CHAD_EVENT_MAL_TYPE	4082	X'00000FF2'
MQRCCF_CHAD_EXIT_ERREUR	4083	X'00000FF3'
MQRCCF_CHAD_EXIT_TYPE_ERREUR	4084	X'00000FF4'
MQRCCF_SUPPRESSED_BY_EXIT	4085	X'00000FF5'
MQRCCF_BATCH_INT_ERREUR	4086	X'00000FF6'
MQRCCF_BATCH_INT_TYPE	4087	X'00000FF7'
MQRCCF_NET_PRIORITY_ERROR	4088	X'00000FF8'
MQRCCF_NET_PRIORITÉ_TYPE_ERREUR	4089	X'00000FF9'
MQRCCF_CHANNEL_CLOSED	4090	X'00000FFA'
MQRCCF_Q_STATUS_NOT_FOUND	4091	X'00000FFB'
MQRCCF_SSL_CIPHER_SPEC_ERROR	4092	X'00000FFC'
MQRCCF_SSL_PEER_NAME_ERROR	4093	X'00000FFD'
MQRCCF_SSL_CLIENT_AUTH_ERROR	4094	X'00000FFE'
MQRCCF_RETAINED_NOT_SUPPORTED	4095	X'00000FFF'

MQRCN_ * (Constantes de reconnexion client)

Tableau 317. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRCN_NO	0	X'00000000'
MQRCN_YES	1	X'00000001'
MQRCN_Q_MGR	2	X'00000002'
MQRCN_DISABLED	3	X'00000003'

MQRCVTIME_ * (types de délai d'attente de réception)

Tableau 318. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRCVTIME_MULTIPLIER	0	X'00000000'
MQRCVTIME_ADD	1	X'00000001'
MQRCVTIME_EQUAL	2	X'00000002'

MQREADA_ * (valeurs de lecture anticipée)

Tableau 319. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQREADA_NO	0	X'00000000'
MQREADA_YES	1	X'00000001'
MQREADA_DISABLED	2	X'00000002'
MQREADA_INHIBÉ	3	X'00000003'
MQREADA_BACKLOG	4	X'00000004'

MQRECORDING_ * (options d'enregistrement)

Tableau 320. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRECORDING_DISABLED	0	X'00000000'
MQRECORDING_Q	1	X'00000001'
MSG_ENREGISTREMENT_MQG	2	X'00000002'

MQREGO_ * (Options d'enregistrement de publication / abonnement)

Tableau 321. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQREGO_AUCUN	0	X'00000000'
MQREGO_CORREL_ID_AS_IDENTITY	1	X'00000001'
MQREGO_ANONYME	2	X'00000002'
MQREGO_LOCAL	4	X'00000004'
MQREGO_DIRECT_REQUESTS	8	X'00000008'
MQREGO_NEW_PUBLICATIONS_ONLY	16	X'00000010'
MQREGO_PUBLISH_ON_REQUEST_ONLY	32	X'00000020'

Tableau 321. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQREGO_DEREGISTER_ALL	64	X'00000040'
MQREGO_INCLUDE_STREAM_NAME	128	X'00000080'
MQREGO_INFORM_SI_RETENU	256	X'00000100'
MQREGO_DUPLICATES_OK	512	X'00000200'
MQREGO_NON_PERSISTENT	1024	X'00000400'
MQREGO_PERSISTENT	2048	X'00000800'
MQREGO_PERSISTENT_AS_PUBLISH	4096	X'00001000'
MQREGO_XX_ENCODE_CASE_ONE persistENT_AS_q	8192	X'00002000'
NOM_ADD_MQREGO_	16384	X'00004000'
MQREGO_NO_ALTÉRATION	32768	X'00008000'
MQREGO_RÉPONSE_COMPLET_RÉPONSE	65536	X'00010000'
MQREGO_JOIN_SHARED	131072	X'00020000'
MQREGO_JOIN_EXCLUSIVE	262144	X'00040000'
MQREGO_LEAVE_ONLY	524288	X'00080000'
ID_UTILISATEUR_MQREGO_VARIABLE_	1048576	X'00100000'
MQREGO_VERROUILLÉ	2097152	X'00200000'

MQRFH_ * (règles et mise en forme de la structure d'en-tête et des indicateurs)

Règles et structure d'en-tête de formatage

Tableau 322. Structures de constantes	
Nom	**Structure**
ID MQRFH_STRUC_ID	"RFH¬"
MQRFH_STRUC_ID_ARRAY	'R','F','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 323. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'

Règles et indicateurs d'en-tête de formatage

Tableau 324. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQRFH_AUCUN	0	X'00000000'
MQRFH_NO_FLAGS	0	X'00000000'

MQRFH2_* (balise Options de publication / abonnement RFH2 Balises de dossier de niveau supérieur)

Tableau 325. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRFH2_NAME_VALUE_VERSION	1	X'00000001'

MQRFH2_* (noms de balise d'options de publication / abonnement)

Tableau 326. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRFH2_PUBSUB_CMD_FOLDER	"psc"	
MQRFH2_PUBSUB_RESP_FOLDER	"pscr"	
MQRFH2_MSG_CONTENT_FOLDER	"mcd"	
MQRFH2_USER_FOLDER	"usr"	

MQRFH2_* (noms de balise XML des options de publication / abonnement)

Tableau 327. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_ * (Longueur renvoyée)

Tableau 328. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRL_UNDEFINED	-1	X'FFFFFFFF'

MQRMH_ * (Structure d'en-tête de message de référence)

Tableau 329. Structures de constantes

Nom	Structure
ID_STRUC_MQRMH_	"RMH¬"
MQRMH_STRUC_ID_ARRAY	'R','M','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 330. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRMH_VERSION_1	1	X'00000001'
MQRMH_VERSION_CURRENT_VERSION	1	X'00000001'

MQRMHF_ * (indicateurs d'en-tête de message de référence)

Tableau 331. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRMHF_LAST	1	X'00000001'
MQRMHF_NOT_LAST	0	X'00000000'

MQRO_ * (options de rapport)

Tableau 332. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRO_EXCEPTION	16777216	X'01000000'
MQRO_EXCEPTION_WITH_DATA	50331648	X'03000000'
MQRO_EXCEPTION_WITH_FULL_DATA	117440512	X'07000000'
MQRO_EXPIRATION	2097152	X'00200000'
MQRO_EXPIRATION_WITH_DATA	6291456	X'00600000'
MQRO_EXPIRATION_WITH_FULL_DATA	14680064	X'00E00000'
MQRO_COA	256	X'00000100'
MQRO_COA_WITH_DATA	768	X'00000300'
MQRO_COA_WITH_FULL_DATA	1792	X'00000700'
MQRO_COD	2048	X'00000800'
MQRO_COD_WITH_DATA	6144	X'00001800'
MQRO_COD_WITH_FULL_DATA	14336	X'00003800'
MQRO_PAN	1	X'00000001'
MQRO_NAN	2	X'00000002'
ACTIVITE MQRO	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_ET_EXPIRATION	16384	X'00004000'
MQRO_AUCUN	0	X'00000000'

MQRO_ * (Masques d'options de rapport)

Tableau 333. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRO_REJECT_UNSUP_MASK	270270464	X'101C0000'
MQRO_ACCEPT_UNSUP_MASK	-270532353	X'EFE000FF'
MQRO_ACCEPT_UNSUP_IF_XMIT_MASK	261888	X'0003FF00'

MQROUTE_ * (Trace-route)

Nombre maximal d'activités de routage de trace (MQIACF_MAX_ACTIVITÉS)

Tableau 334. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_UNLIMITED_ACTIVITÉS	0	X'00000000'

Détails du routage de trace (MQIACF_ROUTE_DETAIL)

Tableau 335. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_DETAIL_LOW	2	X'00000002'
MQROUTE_DETAIL_MEDIUM	8	X'00000008'
MQROUTE_DETAIL_ELEVE	32	X'00000020'

Acheminement de la route de trace (MQIACF_ROUTE_FORWARDING)

Tableau 336. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_ACHEMINEMENT_TOUT	256	X'00000100'
MQROUTE_WARD_IF_SUPPORTED	512	X'00000200'
MQROUTE_TRANSFER_REJ_UNSUP_MASK	-65536	X'FFFF0000'

Distribution de routage de trace (MQIACF_ROUTE_DELIVERY)

Tableau 337. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_DELIVER_YES	4096	X'00001000'
MQROUTE_NO	8192	X'00002000'
MQROUTE_LIVRE_REJ_UNSUP_MASK	-65536	X'FFFF0000'

Accumulation de route de trace (MQIACF_ROUTE_ACCUMULATION)

Tableau 338. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQROUTE_ACCUMULATE_NONE	65539	X'00010003'
MQROUTE_ACCUMULATE_EN_MSG	65540	X'00010004'
MQROUTE_ACCUMULATE_ET_REPONSE	65541	X'00010005'

MQRP_ * (options de remplacement de format de commande)

Tableau 339. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRP_OUI	1	X'00000001'
MQRP_NON	0	X'00000000'

MQRQ_ * (Qualificateurs de raison de format de commande)

Tableau 340. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRQ_CONN_NOT_AUTHORIZED	1	X'00000001'
MQRQ_OPEN_NOT_AUTHORIZED	2	X'00000002'
MQRQ_CLOSE_NOT_AUTHORIZED	3	X'00000003'
MQRQ_CMD_NOT_AUTHORIZED	4	X'00000004'
MQRQ_Q_MGR_STOPPING	5	X'00000005'
MQRQ_Q_MGR_QUIESCING	6	X'00000006'
MQRQ_CHANNEL_STOPPED_OK	7	X'00000007'
MQRQ_CHANNEL_STOPPED_ERROR	8	X'00000008'
MQRQ_CHANNEL_STOPPED_RETRY	9	X'00000009'
MQRQ_CHANNEL_STOPPED_DISABLED	10	X'0000000A'
MQRQ_BRIDGE_STOPPED_OK	11	X'0000000B'
MQRQ_BRIDGE_STOPPED_ERROR	12	X'0000000C'
MQRQ_SSL_HANDSHAKE_ERREUR	13	X'0000000D'
MQRQ_SSL_CIPHER_SPEC_ERROR	14	X'0000000E'
MQRQ_SSL_CLIENT_AUTH_ERROR	15	X'0000000F'
MQRQ_SSL_PEER_NAME_ERREUR	16	X'00000010'
MQRQ_SUB_NOT_AUTHORIZED	17	X'00000011'
MQRQ_SUB_DEST_NOT_AUTHORIZED	18	X'00000012'
MQRQ_SSL_UNKNOWN_RÉVOCATION	19	X'00000013'

MQRT_ * (Types d'actualisation de format de commande)

Tableau 341. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
CONFIGURATION_MQR	1	X'00000001'
EXPIRATION_MQRT_EXPIRATION	2	X'00000002'
MQRT_NSPROC	3	X'00000003'
MQRT_PROXYSUB	4	X'00000004'

MQRU_ * (Demande uniquement)

Tableau 342. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQRU_PUBLISH_ON_REQUEST	1	X'00000001'
MQRU_PUBLISH_ALL	2	X'00000002'

MQSCA_ * (authentification de client SSL)

Tableau 343. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSCA_REQUIS	0	X'00000000'

Tableau 343. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSCA_XX_ENCODE_CASE_ONE facultatif	1	X'00000001'

MQSCO_ * (options de configuration SSL)

Structure des options de configuration SSL

Tableau 344. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQSCO_	"SCO¬"
MQSCO_STRUC_ID_ARRAY	'S','C','O','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 345. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSCO_VERSION_1	1	X'00000001'
MQSCO_VERSION_2	2	X'00000002'
MQSCO_VERSION_3	3	X'00000003'
MQSCO_VERSION_4	4	X'00000004'
VERSION MQSCO_CURRENT_VERSION	4	X'00000004'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Options de configuration SSL Nombre de réinitialisations de clé

Tableau 346. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSCO_RESET_COUNT_DEFAULT	0	X'00000000'

Portée des définitions de file d'attente de format de commande

Tableau 347. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSCO_Q_DIR	1	X'00000001'
MQSCO_CELL	2	X'00000002'

MQSCOPE_ * (portée de publication)

Tableau 348. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSCOPE_ALL	0	X'00000000'
MQSCOPE_AS_PARENT	1	X'00000001'
MQSCOPE_QMGR	4	X'00000004'

MQSCYC_ * (cas de sécurité)

Tableau 349. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSCYC_SUPÉRIEURE	0	X'00000000'
MQSCYC_MIXTE	1	X'00000001'

MQSD_ * (structure de descripteur d'objet)

Tableau 350. Noms et structures des constantes

Nom	Structure
ID_STRUCD_MQS	"SD¬¬"
MQSD_STRUC_ID_ARRAY	'S','D','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 351. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSD_VERSION_1	1	X'00000001'
MQSD_CURRENT_VERSION	1	X'00000001'

MQSECITEM_ * (Eléments de sécurité de format de commande)

Tableau 352. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_ * (Commutateurs de sécurité de format de commande et états de commutation)

Commutateurs de sécurité de format de commande

Tableau 353. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSECSW_PROCESSUS	1	X'00000001'
MQSECSW_NAMELIST	2	X'00000002'

Tableau 353. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_SOUS-SYSTEME	10	X'0000000A'
RESSOURCES MQSECSW_COMMAND_RESSOURCES	11	X'0000000B'
MQSECSW_Q_DIR	15	X'0000000F'
MQSECSW_QSG	16	X'00000010'

Etats des commutateurs de sécurité de format de commande

Tableau 354. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'
ERREUR MQSECSW_OFF_ERROR	25	X'00000019'
MQSECSW_ON_SUBSTITUÉ	26	X'0000001A'

MQSECTYPE_ * (Types de sécurité de format de commande)

Tableau 355. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSECTYPE_AUTHSERV	1	X'00000001'
MQSECTYPE_SSL	2	X'00000002'
MQSECTYPE_CLASSES	3	X'00000003'

MQSEG_ * (Segmentation)

Tableau 356. Noms et valeurs de constante	
Nom	**Valeur**
MQSEG_INHIBÉ	'¬'
MQSEG_AUTORISÉ_	'A'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQSEL_ * (Valeurs de sélecteur spéciales)

Tableau 357. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSEL_ANY_SELECTOR	-30001	X'FFFF8ACF'

Tableau 357. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSEL_ANY_USER_SELECTOR	-30002	X'FFFF8ACE'
MQSEL_ANY_SYSTEM_SELECTOR	-30003	X'FFFF8ACD'
MQSEL_ALL_SELECTEURS	-30001	X'FFFF8ACF'
MQSEL_ALL_USER_SELECTEURS	-30002	X'FFFF8ACE'
MQSEL_ALL_SYSTEM_SELECTEURS	-30003	X'FFFF8ACD'

MQSELTYPE_ * (types de sélecteur)

Tableau 358. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSELTYPE_NONE	0	X'00000000'
MQSELTYPE_STANDARD	1	X'00000001'
MQSELTYPE_ETENDU	2	X'00000002'

MQSID_ * (identificateur de sécurité)

Tableau 359. Noms et valeurs de constante	
Nom	**Valeur**
MQSID_AUCUN	X'00...00' (40 valeurs nulles)
MQSID_NON_ARRAY	'\0','\0',... (40 valeurs nulles)

MQSIDT_ * (types d'identificateur de sécurité)

Tableau 360. Noms et valeurs de constante	
Nom	**Valeur hexadécimale**
MQSIDT_AUCUN	X'00'
ID_SÉCURITÉ_NT_MQSIDT_ID	X'01'
MQSIDT_ID_SÉCURITÉ_WAS	X'02'

MQSMPO_ * (Définition des options et de la structure des propriétés de message)

Définir la structure des options de propriété de message

Tableau 361. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQSMPO_	"SMPO"
MQSMPO_STRUC_ID_ARRAY	'S','M','P','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 362. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSMPO_VERSION_1	1	X'00000001'
MQSMPO_CURRENT_VERSION	1	X'00000001'

Définir les options de propriété de message

Tableau 363. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSMPO_SET_FIRST	0	X'00000000'
MQSMPO_SET_PROP_UNDER_CURSOR	1	X'00000001'
MQSMPO_SET_PROP_AFTER_CURSOR	2	X'00000002'
MQSMPO_XX_ENCODE_CASE_ONE annexes	4	X'00000004'
MQSMPO_SET_PROP_BEFORE_CURSOR	8	X'00000008'
MQSMPO_NONE	0	X'00000000'

MQSO_ * (options d'abonnement)

Tableau 364. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSO_AUCUN	0	X'00000000'
MQSO_NON_DURABLE	0	X'00000000'
MQSO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQSO_ALTER	1	X'00000001'
MQSO_CREER	2	X'00000002'
MQSO_RESUME	4	X'00000004'
MQSO_DURABLE	8	X'00000008'
SOUS-GROUPE_MQSO	16	X'00000010'
MQSO_GERE	32	X'00000020'
MQSO_SET_IDENTITY_CONTEXT	64	X'00000040'
ID_UTILISATEUR_FIXE_MQSO_FIXE	256	X'00000100'
ID utilisateur MQSO_ANY_USERID	512	X'00000200'
MQSO_PUBLICATIONS_ON_REQUEST	2048	X'00000800'
MQSO_NEW_PUBLICATIONS_ONLY	4096	X'00001000'
MQSO_FAIL_IF_QUIESCING	8192	X'00002000'
MQSO_ALTERNATE_USER_AUTHORITY	262144	X'00040000'
MQSO_WILDCARD_CHAR	1048576	X'00100000'
MQSO_WILDCARD_TOPIC	2097152	X'00200000'
ID_CORREL_SET_MQSO	4194304	X'00400000'
MQSO_SCOPE_QMGR	67108864	X'04000000'
MQSO_NO_READ_AHEAD	134217728	X'08000000'
MQSO_READ_AHEAD	268435456	X'10000000'

MQSP_ * (disponibilité du point de synchronisation)

Tableau 365. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSP_XX_ENCODE_CASE_ONE disponible	1	X'00000001'
MQSP_NON DISPONIBLE	0	X'00000000'

MQSQQM_ * (nom du gestionnaire de files d'attente partagées)

Tableau 366. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSQQM_USE	0	X'00000000'
MQSQQM_IGNORE	1	X'00000001'

MQSR_ * (Action)

Tableau 367. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
PUBLICATION MQSR_ACTION_PUBLICATION	1	X'00000001'

MQSRO_ * (structure des options de demande d'abonnement)

Tableau 368. Structures de constantes

Nom	Structure
ID_STRUC_MQS	"SRO¬"
MQSRO_STRUC_ID_ARRAY	'S','R','O','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 369. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSRO_VERSION_1	1	X'00000001'
VERSION_MQSRO_CURRENT_VERSION	1	X'00000001'
MQSRO_AUCUN	0	X'00000000'
MQSRO_FAIL_IF_QUIESCING	8192	X'00002000'

MQSS_ * (statut du segment)

Tableau 370. Noms et structures des constantes

Nom	Structure
MQSS_NON_SEGMENT	'¬'
SEGMENT_MQSS_SEGMENT	'S'
SEGMENT_LAST_MQSS_	'L'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQSSL_ * (exigences SSL FIPS)

Tableau 371. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSSL_FIPS_NO	0	X'00000000'
MQSSL_FIPS_YES	1	X'00000001'

MQSTAT_ * (options de statistiques)

MQSTAT_TYPE_ASYNC_ERROR	0	X'00000000'

MQSTAT_TYPE_RECONNEXION	0	X'00000000'
MQSTAT_TYPE_RECONNECTION_ERROR	0	X'00000000'

MQSTS_ * (Structure de la structure de génération de rapports de statut)

Tableau 372. Structures de constantes

Nom	Structure
ID_STRUCTE_MQST	"STAT"
MQSTS_STRUC_ID_ARRAY	'S','T','A','T'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 373. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSTS_VERSION_1	1	X'00000001'
MQSTS_CURRENT_VERSION	1	X'00000001'

MQSUB_ * (Abonnements durables)

Abonnements durables

Tableau 374. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSUB_DURABLE_AS_PARENT	0	X'00000000'
MQSUB_DURABLE_ALLOWED	1	X'00000001'
MQSUB_DURABLE_INHIBÉE	2	X'00000002'

Abonnements durables

Tableau 375. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSUB_DURABLE_ALL	-1	X'FFFFFFFF'
MQSUB_DURABLE_YES	1	X'00000001'
MQSUB_DURABLE_NO	2	X'00000002'

MQSUBTYPE_ * (Types d'abonnement au format de commande)

Tableau 376. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSUBTYPE_API	1	X'00000001'
MQSUBTYPE_ADMIN	2	X'00000002'
MQSUBTYPE_PROXY	3	X'00000003'
MQSUBTYPE_ALL	-1	X'FFFFFFFF'
MQSUBTYPE_UTILISATEUR	-2	X'FFFFFFFE'

MQSUS_ * (statut d'interruption du format de commande)

Tableau 377. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSUS_OUI	1	X'00000001'
MQSUS_NO	0	X'00000000'

MQSVC_ * (Service)

Types de service

Tableau 378. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSVC_TYPE_COMMANDE	0	X'00000000'
SERVEUR_TYPE_MQ	1	X'00000001'

Contrôles de service

Tableau 379. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSVC_CONTROL_Q_MGR	0	X'00000000'
MQSVC_CONTROL_Q_MGR_START	1	X'00000001'
MQSVC_CONTROL_MANUAL	2	X'00000002'

Statut du service

Tableau 380. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSVC_STATUS_STOPPED	0	X'00000000'
MQSVC_STATUS_DEMARRAGE	1	X'00000001'
MQSVC_STATUS_RUNNING	2	X'00000002'
MQSVC_STATUS_ARRÊT	3	X'00000003'
MQSVC_STATUS_NOUVELLE tentative	4	X'00000004'

MQSYNCPOINT_ * (valeurs de point de synchronisation de format de commande pour la migration de publication / d'abonnement)

Tableau 381. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSYNCPOINT_OUI	0	X'00000000'
MQSYNCPOINT_IFPER	1	X'00000001'

MQSYSP_ * (valeurs de paramètre système de format de commande)

Tableau 382. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQSYSP_NO	0	X'00000000'

Tableau 382. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQSYSP_OUI	1	X'00000001'
MQSYSP_ETENDU	2	X'00000002'
MQSYSP_TYPE_INITIAL	10	X'0000000A'
MQSYSP_TYPE_SET	11	X'0000000B'
MQSYSP_TYPE_LOG_COPY	12	X'0000000C'
ETAT_LOG_TYPE_MQSYSP_	13	X'0000000D'
MQSYSP_TYPE_ARCHIVE_TAPE	14	X'0000000E'
MQSYSP_ALLOC_BLK	20	X'00000014'
MQSYSP_ALLOC_TRK	21	X'00000015'
MQSYSP_ALLOC_CYL	22	X'00000016'
MQSYSP_STATUS_BUSY	30	X'0000001E'
MQSYSP_STATUS_PREMOUNT	31	X'0000001F'
MQSYSP_STATUS_XX_ENCODE_CASE_ONE disponible	32	X'00000020'
MQSYSP_STATUS_INCONNU	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_ * (attributs de rubrique)

Caractères génériques

Tableau 383. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
BLOC_MQTA_BLOC	1	X'00000001'
MQTA_PASSTHRU	2	X'00000002'

Abonnements autorisés

Tableau 384. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQTA_SUB_AS_PARENT	0	X'00000000'
MQTA_SUB_INHIBÉ	1	X'00000001'
MQTA_SOUS-AUTORISÉ_AUTORISÉ	2	X'00000002'

Sous-propagation du proxy

Tableau 385. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQTA_PROXY_SUB_FORCE	1	X'00000001'
MQTA_PROXY_SUB_FIRSTUSE	2	X'00000002'

Publications autorisées

Tableau 386. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQTA_PUB_AS_PARENT	0	X'00000000'
MQTA_PUB_INHIBÉ	1	X'00000001'
MQTA_PUB_AUTORISÉ	2	X'00000002'

MQTC_ * (contrôles de déclencheur)

Tableau 387. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQTC_OFF	0	X'00000000'
MQTC_ON	1	X'00000001'

MQTCPKEEP_ * (signal de présence TCP)

Tableau 388. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQTCPKEEP_NO	0	X'00000000'
MQTCPKEEP_OUI	1	X'00000001'

MQTCPSTACK_ * (types de pile TCP)

Tableau 389. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQTCPSTACK_SINGLE	0	X'00000000'
MQTCPSTACK_MULTIPLE	1	X'00000001'

MQTIME_ * (Unités de temps de format de commande)

Tableau 390. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQTIME_UNIT_MINS	0	X'00000000'
MQTIME_UNIT_SECS	1	X'00000001'

MQTM_ * (structure de message de déclenchement)

Tableau 391. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQTM	"TM¬¬"
MQTM_STRUC_ID_ARRAY	'T','M','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 392. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQTM_VERSION_1	1	X'00000001'
MQTM_CURRENT_VERSION	1	X'00000001'

MQTMC_ * (structure de format de caractère de message de déclencheur)

Tableau 393. Structures de constantes

Nom	Structure
ID MQTMC_STRUC_ID	"TMC¬"
MQTMC_STRUC_ID_ARRAY	'T','M','C','¬'
MQTMC_VERSION_1	"¬¬¬1"
MQTMC_VERSION_2	"¬¬¬2"
MQTMC_CURRENT_VERSION	"¬¬¬2"
MQTMC_VERSION_1_ARRAY	'¬','¬','¬','1'
MQTMC_VERSION_2_ARRAY	'¬','¬','¬','2'
MQTMC_CURRENT_VERSION_ARRAY	'¬','¬','¬','2'

Remarque : Le symbole ¬ représente un caractère blanc unique.

MQTOPT_ * (Type de rubrique)

Tableau 394. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTOPT_LOCAL	0	X'00000000'
MQTOPT_CLUSTER	1	X'00000001'
MQTOPT_ALL	2	X'00000002'

MQTRAXSTR_ * (Démarrage automatique de la trace de l'initialisateur de canal)

Tableau 395. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTRAXSTR_NO	0	X'00000000'
MQTRAXSTR_YES	1	X'00000001'

MQTSCOPE_ * (portée de l'abonnement)

Tableau 396. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTSCOPE_QMGR	1	X'00000001'
MQTSCOPE_ALL	2	X'00000002'

MQTT_ * (types de déclencheur)

Tableau 397. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTT_AUCUN	0	X'00000000'
MQTT_PREMIER	1	X'00000001'
MQTT_EVERY	2	X'00000002'
MQTT_DEPTH	3	X'00000003'

MQTYPE_ * (types de données de propriété)

Tableau 398. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQTYPE_AS_SET	0	X'00000000'
MQTYPE_NULL	2	X'00000002'
MQTYPE_BOOLÉEN	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_CHAINE	1024	X'00000400'

MQUA_ * (Sélecteurs d'attribut utilisateur de publication / abonnement)

Tableau 399. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQUA_FIRST	65536	X'00010000'
MQUA_LAST	999999999	X'3B9AC9FF'

MQUIDSUPP_ * (Prise en charge de l'ID utilisateur au format de commande)

Tableau 400. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQUIDSUPP_NO	0	X'00000000'
MQUIDSUPP_OUI	1	X'00000001'

MQUNDELIVERED_ * (valeurs de format de commande non distribuées pour la migration de publication / abonnement)

Tableau 401. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
Mqundelivered_normal	0	X'00000000'
MQUNDELIVERED_SAFE	1	X'00000001'
Carte MQUNDELIVERED_DISCARD	2	X'00000002'
MQUNDELIVERED_KEEP	3	X'00000003'

MQUOWST_ * (Etats de l'unité de travail du format de commande)

Tableau 402. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQUOWST_AUCUN	0	X'00000000'

Tableau 402. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQUOWST_ACTIF	1	X'00000001'
MQUOWST_PREPARE	2	X'00000002'
MQUOWST_NON résolu	3	X'00000003'

MQUOWT_ * (Types d'unité de travail de format de commande)

Tableau 403. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQUOWT_Q_MGR	0	X'00000000'
MQUOWT_CICS	1	X'00000001'
RRS MQUOWT_RRS	2	X'00000002'
MQUOWT_IMS	3	X'00000003'
MQUOWT_XA	4	X'00000004'

MQUS_ * (utilisations de file d'attente)

Tableau 404. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQUS_NORMAL	0	X'00000000'
TRANSMIS_MQUS_TRANSMISSION	1	X'00000001'

MQUSAGE_ * (valeurs d'utilisation de l'ensemble de pages de format de commande et valeurs d'utilisation de l'ensemble de données)

Valeurs d'utilisation de l'ensemble de pages de format de commande

Tableau 405. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQUSAGE_PS_XX_ENCODE_CASE_ONE disponible	0	X'00000000'
MQUSAGE_PS_DEFINI	1	X'00000001'
MQUSAGE_PS_OFFLINE	2	X'00000002'
MQUSAGE_PS_NOT_DEFINI	3	X'00000003'
MQUSAGE_EXPAND_USER	1	X'00000001'
MQUSAGE_EXPAND_SYSTEM	2	X'00000002'
MQUSAGE_EXPAND_NONE	3	X'00000003'

Valeurs d'utilisation du fichier de format de commande

Tableau 406. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_ * (longueur de la valeur)

Tableau 407. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQVL_NULL_TERMINATED	-1	X'FFFFFFFF'
MQVL_EMPTY_CHAINE	0	X'00000000'

MQVU_ * (ID utilisateur de variable)

Tableau 408. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQVU_UTILISATEUR_FIXED_USER	1	X'00000001'
Utilisateur MQVU_ANY_USER	2	X'00000002'

MQWDR_ * (structure d'enregistrement de destination d'exit de charge de travail de cluster)

Tableau 409. Structures de constantes

Nom	Structure
ID_STRUC_MQWDR_	"WDR¬"
MQWDR_STRUC_ID_ARRAY	'W','D','R','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 410. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQWDR_VERSION_1	1	X'00000001'
MQWDR_VERSION_2	2	X'00000002'
MQWDR_VERSION_CURRENT_	2	X'00000002'
MQWDR_LENGTH_1	124	X'0000007C'
MQWDR_LENGTH_2	136	X'00000088'
MQWDR_LONGUEUR_EN_COURS	136	X'00000088'

MQWI_ * (intervalle d'attente)

Tableau 411. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQWI_ILLIMITÉ	-1	X'FFFFFFFF'

MQWIH_ * (structure d'en-tête d'informations de charge de travail et indicateurs)

Structure d'en-tête des informations de charge de travail

Tableau 412. Structures de constantes

Nom	Structure
ID_STRUC_MQWIH	"WIH¬"
MQWIH_STRUC_ID_ARRAY	'W','I','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 413. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQWIH_VERSION_1	1	X'00000001'
MQWIH_CURRENT_VERSION	1	X'00000001'
MQWIH_LENGTH_1	120	X'00000078'
MQWIH_LONGUEUR_EN_COURS	120	X'00000078'

Indicateurs d'en-tête d'informations de charge de travail

Tableau 414. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQWIH_AUCUN	0	X'00000000'

MQWQR_ * (structure d'enregistrement de file d'attente d'exit de charge de travail de cluster)

Tableau 415. Structures de constantes	
Nom	**Structure**
MQWQR_STRUC_ID	"WQR¬"
MQWQR_STRUC_ID_ARRAY	'W','Q','R','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 416. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_LONGUEUR_EN_COURS	212	X'000000D4'

MQWS_ * (Schéma de caractère générique)

Tableau 417. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQWS_VALEUR PAR DEFAUT	0	X'00000000'
MQWS_CHAR	1	X'00000001'
MQWS_TOPIC	2	X'00000002'

MQWXP_ * (structure des paramètres d'exit de charge de travail de cluster)

MQWXP_ * (structure des paramètres d'exit de charge de travail de cluster)

Tableau 418. Structures de constantes

Nom	Structure
MQWXP_STRUC_ID	"WXP¬"
MQWXP_STRUC_ID_ARRAY	'W','X','P','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 419. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQWXP_VERSION_1	1	X'00000001'
MQWXP_VERSION_2	2	X'00000002'
MQWXP_VERSION_3	3	X'00000003'
MQWXP_VERSION_4	4	X'00000004'
MQWXP_CURRENT_VERSION	4	X'00000004'

MQWXP_ * (indicateurs de charge de travail de cluster)

Tableau 420. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQWXP_PUT_BY_CLUSTER_CHL	2	X'00000002'

Référence associée
Zones de MQWXP-Structure des paramètres d'exit de charge de travail de cluster

MQXACT_ * (types d'appelant d'API)

Tableau 421. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQXACT_EXTERNAL	1	X'00000001'
MQXACT_INTERNAL	2	X'00000002'

MQXC_ * (commandes exit)

Tableau 422. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
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_ * (réponses d'exit)

Tableau 423. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQXCC_OK	0	X'00000000'
MQXCC_SUPPRESS_FUNCTION	-1	X'FFFFFFFF'
MQXCC_LISTE_FONCTION	-2	X'FFFFFFFE'
MQXCC_SEND_AND_REQUEST_SEC_MSG	-3	X'FFFFFFFD'
MQXCC_SEND_SEC_MSG	-4	X'FFFFFFFC'
MQXCC_SUPPRESS_EXIT	-5	X'FFFFFFFB'
MQXCC_CANAL_FERMÉ	-6	X'FFFFFFFA'
Accusé de réception MQXCC_REQUEST_ACK	-7	X'FFFFFFF9'
MQXCC_ECHEC	-8	X'FFFFFFF8'

MQXDR_ * (réponse d'exit)

Tableau 424. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQXDR_OK	0	X'00000000'
Echec de MQXDR_CONVERSION_FAILED	1	X'00000001'

MQXE_ * (Environnements)

Tableau 425. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQXE_AUTRES	0	X'00000000'
MQXE_MCA	1	X'00000001'
MQXE_MCA_SVRCONN	2	X'00000002'
SERVEUR_COMMAND_MQ	3	X'00000003'
MQXE_MQSC	4	X'00000004'

MQXEPO_ * (structure des options de point d'entrée de registre et options de sortie)

Structure des options de point d'entrée de registre

Tableau 426. Structures de constantes

Nom	Structure
MQXEPO_STRUC_ID	"XEPO"
MQXEPO_STRUC_ID_ARRAY	'X','E','P','O'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 427. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQXEPO_VERSION_1	1	X'00000001'
MQXEPO_VERSION_CURRENT_	1	X'00000001'

Options de sortie

Tableau 428. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQXEPO_NONE	0	X'00000000'

MQXF_ * (identificateurs de fonction d'API)

Tableau 429. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQXF_INIT	1	X'00000001'
MQXF_TERME	2	X'00000002'
MQXF_CONN	3	X'00000003'
MQXF_CONNX	4	X'00000004'
MQXF_DISQUE	5	X'00000005'
MQXF_OPEN	6	X'00000006'
MQXF_FERMETURE	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_DÉBUT	14	X'0000000E'
MQXF_CMIT	15	X'0000000F'
MQXF_RETOUR	16	X'00000010'
MQXF_STAT	18	X'00000012'
MQXF_CB	19	X'00000013'
MQXF_CTL	20	X'00000014'
MQXF_RAPPEL	21	X'00000015'
MQXF_SUB	22	X'00000016'
MQXF_SUBRQ	23	X'00000017'
MQXF_XACLOSE	24	X'00000018'
MQXF_XACOMMIT	25	X'00000019'
MQXF_XATERMINE	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'

Tableau 429. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQXF_AXUNREG	35	X'00000023'

MQXP_ * (structure des paramètres d'exit de croisement d'API)

Tableau 430. Structures de constantes	
Nom	**Structure**
ID_STRUCT_MQXP	"XP¬¬"
MQXP_STRUC_ID_ARRAY	'X','P','¬','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 431. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQXP_VERSION_1	1	X'00000001'

MQXPDA_ * (Zone d'identification des incidents)

Tableau 432. Noms et valeurs de constante	
Nom	**Valeur**
MQXPDA_NONE	X'00...00' (48 valeurs nulles)
MQXPDA_NONE_ARRAY	'\0','\0',... (48 valeurs nulles)

MQXPT_ * (types de transport)

Tableau 433. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQXPT_ALL	-1	X'FFFFFFFF'
MQXPT_LOCAL	0	X'00000000'
MQXPT_LU62	1	X'00000001'
MQXPT_TCP	2	X'00000002'
MQXPT_NETBIOS	3	X'00000003'
MQXPT_SPX	4	X'00000004'
MQXPT_DECNET	5	X'00000005'
MQXPT_UDP	6	X'00000006'

MQXQH_ * (structure d'en-tête de file d'attente de transmission)

Tableau 434. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQXQH_	"XQH¬"
MQXQH_TABLE_ID_STRUCTURE	'X','Q','H','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 435. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQXQH_VERSION_1	1	X'00000001'
MQXQH_CURRENT_VERSION	1	X'00000001'

MQXR_ * (Raisons de l'exit)

Tableau 436. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQXR_AVANT	1	X'00000001'
MQXR_APRES	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'
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_* (réponse d'exit 2)

Tableau 437. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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'

Tableau 437. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQXR2_STATIC_CACHE	0	X'00000000'
MQXR2_DYNAMIC_CACHE	32	X'00000020'

MQXT_ * (identificateurs d'exit)

Tableau 438. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
EXIT 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'
EXIT 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_ * (valeur de la zone utilisateur d'exit)

Tableau 439. Noms et valeurs de constante	
Nom	**Valeur**
MQXUA_NONE	X'00...00' (16 valeurs nulles)
MQXUA_NON_ARRAY	'\0','\0',... (16 valeurs nulles)

MQXWD_ * (structure de descripteur d'attente d'exit)

Tableau 440. Structures de constantes	
Nom	**Structure**
ID_STRUCD_MQXWD_	"XWD¬"
MQXWD_STRUC_ID_ARRAY	'X','W','D','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 441. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQXWD_VERSION_1	1	X'00000001'

MQZAC_ * (structure de contexte d'application)

Tableau 442. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQZAC	"ZAC¬"
MQZAC_STRUC_ID_ARRAY	'Z','A','C','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 443. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZAC_VERSION_1	1	X'00000001'
MQZAC_CURRENT_VERSION	1	X'00000001'

MQZAD_ * (Structure de données de droits d'accès)

Tableau 444. Structures de constantes

Nom	Structure
ID_STRUC_MQZ	"ZAD¬"
MQZAD_STRUC_ID_ARRAY	'Z','A','D','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 445. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZAD_VERSION_1	1	X'00000001'
MQZAD_VERSION_2	2	X'00000002'
MQZAD_CURRENT_VERSION	2	X'00000002'

MQZAET_ * (types d'entité de services installables)

Tableau 446. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZAET_AUCUN	0	X'00000000'
MQZAET_PRINCIPAL	1	X'00000001'
GROUPE MQZAET_GROUP	2	X'00000002'
MQZAET_INCONNU	3	X'00000003'

MQZAO_ * (Autorisations des services installables)

Tableau 447. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZAO_CONNECT	1	X'00000001'
MQZAO_PARCOURIR	2	X'00000002'
MQZAO_ENTREE	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_PUBLICATION	2048	X'00000800'

Tableau 447. Valeurs des constantes (suite)		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQZAO_ABONNEMENT	4096	X'00001000'
MQZAO_RESUME	8192	X'00002000'
MQZAO_ALL_MQI	16383	X'00003FFF'
MQZAO_CREER	65536	X'00010000'
MQZAO_DELETE	131072	X'00020000'
MQZAO_DISPLAY	262144	X'00040000'
MODIFICATION MQZAO_DE	524288	X'00080000'
MQZAO_CLEAR	1048576	X'00100000'
CONTROLE MQZ	2097152	X'00200000'
MQZAO_CONTRÔLE_ÉTENDU	4194304	X'00400000'
MQZAO_AUTORISATION	8388608	X'00800000'
MQZAO_ALL_ADMIN	16646144	X'00FE0000'
MQZAO_ALL	16662527	X'00FE3FFF'
MQZAO_RETRAIT	16777216	X'01000000'
MQZAO_AUCUN	0	X'00000000'

MQZAS_ * (version de l'interface de service des services installables)

Tableau 448. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
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_ * (types d'authentification)

Tableau 449. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQZAT_INITIAL_CONTEXT	0	X'00000000'
MQZAT_CHANGE_CONTEXT	1	X'00000001'

MQZCI_ * (indicateur de continuation des services installables)

Tableau 450. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQZCI_DEFAULT	0	X'00000000'
MQZCI_CONTINUER	0	X'00000000'
MQZCI_ARRET	1	X'00000001'

MQZED_ * (structure de données d'entité)

Tableau 451. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQZ	"ZED¬"
MQZED_STRUC_ID_ARRAY	'Z','E','D','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 452. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQZED_VERSION_1	1	X'00000001'
MQZED_VERSION_2	2	X'00000002'
VERSION_MQZED_CURRENT_VERSION	2	X'00000002'

MQZFP_ * (structure des paramètres disponibles)

Tableau 453. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQZFP	"ZFP¬"
MQZFP_STRUC_ID_ARRAY	'Z','F','P','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 454. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQZFP_VERSION_1	1	X'00000001'
VERSION_MQZFP_CURRENT_VERSION	1	X'00000001'

MQZIC_ * (structure de contexte d'identité)

Tableau 455. Structures de constantes	
Nom	**Structure**
ID_STRUC_MQZ	"ZIC¬"
MQZIC_STRUC_ID_ARRAY	'Z','I','C','¬'

Remarque : Le symbole ¬ représente un caractère blanc unique.

Tableau 456. Valeurs des constantes		
Nom	**Valeur décimale**	**Valeur hexadécimale**
MQZIC_VERSION_1	1	X'00000001'
MQZIC_CURRENT_VERSION	1	X'00000001'

MQZID_ * (ID de fonction pour les services)

ID de fonction communs à tous les services

Tableau 457. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZID_INIT	0	X'00000000'
MQZID_TERM	1	X'00000001'

ID de fonction pour le service de droits d'accès

Tableau 458. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZID_INIT_AUTHORITY	0	X'00000000'
MQZID_TERM_AUTHORITY	1	X'00000001'
MQZID_CHECK_AUTHORITY	2	X'00000002'
MQZID_COPY_ALL_AUTHORITY	3	X'00000003'
MQZID_DELETE_AUTHORITY	4	X'00000004'
MQZID_SET_AUTHORITY	5	X'00000005'
MQZID_GET_AUTHORITY	6	X'00000006'
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'
Utilisateur_FREE_MQZ	11	X'0000000B'
MQZID_INQUIRE	12	X'0000000C'
MQZID_CHECK_PRIVILEGED	13	X'0000000D'

ID de fonction pour le service Name

Tableau 459. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
NOM_INIT_MQZID	0	X'00000000'
NOM_MQZID_TERM	1	X'00000001'
MQZID_LOOKUP_NAME	2	X'00000002'
MQZID_INSERT_NAME	3	X'00000003'
NOM_DELETE_MQZ	4	X'00000004'

ID de fonction pour le service Userid

Tableau 460. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
ID_INIT_ID_MQZID	0	X'00000000'
ID utilisateur MQZID_TERM_	1	X'00000001'
MQZID_FIND_USERID	2	X'00000002'

MQZIO_ * (Options d'initialisation des services installables)

Tableau 461. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZIO_PRIMAIRE	0	X'00000000'
MQZIO_SECONDAIRE	1	X'00000001'

MQZNS_ * (Version de l'interface de service de nom)

Tableau 462. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZNS_VERSION_1	1	X'00000001'

MQZSE_ * (Indicateur de début-énumération des services installables)

Tableau 463. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
DEMARRAGE MQZ	1	X'00000001'
MQZSE_CONTINUE	0	X'00000000'

MQZSL_ * (indicateur de sélecteur de services installables)

Tableau 464. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZSL_NOT_RETURNED	0	X'00000000'
MQZSL_RENVOYE	1	X'00000001'

MQZTO_ * (Options de résiliation des services installables)

Tableau 465. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZTO_PRIMAIRE	0	X'00000000'
MQZTO_SECONDAIRE	1	X'00000001'

MQZUS_ * (version de l'interface de service d'ID utilisateur)

Tableau 466. Valeurs des constantes

Nom	Valeur décimale	Valeur hexadécimale
MQZUS_VERSION_1	1	X'00000001'

Types de données utilisés dans l'interface MQI

Informations sur les types de données pouvant être utilisés dans MQI. Descriptions, zones et déclarations de langue pour les langues appropriées avec chaque type de données.

Présentation des types de données utilisés dans l'interface MQI

Cette section présente les types de données utilisés dans l'interface MQI et vous explique comment les utiliser dans les langages de programmation pris en charge.

Types de données élémentaires

Cette section contient des informations sur les types de données utilisés dans l'interface MQI (ou dans les fonctions d'exit). Ils sont décrits en détail, suivis d'exemples montrant comment déclarer les types de données élémentaires dans les langages de programmation pris en charge dans les rubriques suivantes.

Les types de données utilisés dans l'interface MQI (ou dans les fonctions d'exit) sont les suivants:

- Types de données élémentaires, ou
- Agrégats de types de données élémentaires (tableaux ou structures)

Les types de données élémentaires suivants sont utilisés dans l'interface MQI (ou dans les fonctions d'exit):

Nom du type de données élémentaires	**Type de données**	**Description**
MQBOOL	Boolean	Le type de données MQBOOL représente une valeur booléenne. La valeur 0 représente false. Toute autre valeur représente true. Un MQBOOL doit être aligné comme pour le type de données MQLONG.

Nom du type de données élémentaires	Type de données	Description
MQBYTE	Octet	Le type de données MQBYTE représente un octet de données unique. Aucune interprétation particulière n'est placée sur l'octet ; elle est traitée comme une chaîne de bits, et non comme un nombre ou un caractère binaire. Aucun alignement spécial n'est requis. Lorsque des données MQBYTE sont envoyées entre des gestionnaires de files d'attente qui utilisent des jeux de caractères ou des codages différents, les données MQBYTE ne sont *pas* converties de quelque manière que ce soit. Les zones *MsgId* et *CorrelId* de la structure MQMD sont similaires à ceci. Un tableau de MQBYTE est parfois utilisé pour représenter une zone de la mémoire principale qui n'est pas connue du gestionnaire de files d'attente. Par exemple, la zone peut contenir des données de message d'application ou une structure. L'alignement des limites de cette zone doit être compatible avec la nature des données qu'elle contient. Dans le langage de programmation C, tout type de données peut être utilisé pour les paramètres de fonction affichés sous forme de tableaux de MQBYTE. En effet, ces paramètres sont toujours transmis par adresse, et en C, le paramètre de fonction est déclaré comme un pointeur vers vide.

Nom du type de données élémentaires	Type de données	Description
MQBYTEn	Chaîne de *n* octets	Chaque type de données MQBYTEn représente une chaîne de *n* octets, où *n* peut prendre l'une des valeurs suivantes: 8, 16, 24, 32, 40 ou 128. Chaque octet est décrit par le type de données MQBYTE. Aucun alignement spécial n'est requis. Si les données de la chaîne d'octets sont plus courtes que la longueur définie de la chaîne, les données doivent être complétées par des valeurs nulles pour remplir la chaîne. Lorsque le gestionnaire de files d'attente renvoie des chaînes d'octets à l'application (par exemple, sur l'appel MQGET), le gestionnaire de files d'attente remplit avec des valeurs nulles la longueur définie de la chaîne. Les constantes nommées sont disponibles pour définir les longueurs des zones de chaîne d'octets. Ces éléments sont répertoriés dans le «Constantes», à la page 50

Nom du type de données élémentaires	Type de données	Description
MQCHAR	Caractère	Le type de données MQCHAR représente un caractère à un octet ou un octet d'un caractère à deux octets ou à plusieurs octets. Aucun alignement spécial n'est requis. Lorsque des données MQCHAR sont envoyées entre des gestionnaires de files d'attente qui utilisent des jeux de caractères ou des codages différents, les données MQCHAR doivent généralement être converties pour que les données soient interprétées correctement. Le gestionnaire de files d'attente effectue cette opération automatiquement pour les données MQCHAR dans la structure MQMD. La conversion des données MQCHAR dans les données de message d'application est contrôlée par l'option MQGMO_CONVERT spécifiée dans l'appel MQGET ; voir la description de cette option dans «Options MQGMO-Get-message», à la page 345 pour plus de détails.

Nom du type de données élémentaires	Type de données	Description
MQCHARn	Chaîne de *n* caractères	Chaque type de données MQCHARn représente une chaîne de *n* caractères, où *n* peut prendre l'une des valeurs suivantes: 4, 8, 12, 20, 28, 32, 48, 64, 128 ou 256. Chaque caractère est décrit par le type de données MQCHAR. Aucun alignement spécial n'est requis. Si les données de la chaîne sont plus courtes que la longueur définie de la chaîne, les données doivent être complétées par des blancs pour remplir la chaîne. Dans certains cas, un caractère nul peut être utilisé pour terminer la chaîne prématurément, au lieu d'être rempli avec des blancs ; le caractère nul et les caractères qui le suivent sont traités comme des blancs, jusqu'à la longueur définie de la chaîne. Les emplacements où une valeur nulle peut être utilisée sont identifiés dans les descriptions d'appel et de type de données. Lorsque le gestionnaire de files d'attente renvoie des chaînes de caractères à l'application (par exemple, sur l'appel MQGET), le gestionnaire de files d'attente remplit toujours avec des blancs la longueur définie de la chaîne ; le gestionnaire de files d'attente n'utilise pas le caractère null pour délimiter la chaîne. Des constantes nommées sont disponibles pour définir les longueurs des zones de chaîne de caractères et sont répertoriées dans «Constantes», à la page 50.

Nom du type de données élémentaires	Type de données	Description
MQFLOAT32	Nombre à virgule flottante 32 bits	Le type de données MQFLOAT32 est un nombre à virgule flottante 32 bits représenté à l'aide du format à virgule flottante IEEE standard. Un MQFLOAT32 doit être aligné sur une limite de 4 octets. L'utilisation de MQFLOAT32 dans C sous z/OS nécessite l'utilisation de l'indicateur de compilateur FLOAT (IEEE). L'utilisation de MQFLOAT32 en COBOL est limitée aux compilateurs qui prennent en charge les nombres à virgule flottante au format IEEE. Cela peut nécessiter l'utilisation de l'indicateur de compilateur FLOAT (NATIVE).
MQFLOAT64	Nombre à virgule flottante 64 bits	Le type de données MQFLOAT64 est un nombre à virgule flottante 64 bits représenté à l'aide du format à virgule flottante IEEE standard. Un MQFLOAT64 doit être aligné sur une limite de 8 octets. L'utilisation de MQFLOAT64 dans C sous z/OS nécessite l'utilisation de l'indicateur de compilateur FLOAT (IEEE). L'utilisation de MQFLOAT64 en COBOL est limitée aux compilateurs qui prennent en charge les nombres à virgule flottante au format IEEE. Cela peut nécessiter l'utilisation de l'indicateur de compilateur FLOAT (NATIVE).

Nom du type de données élémentaires	Type de données	Description
MQHCONFIG	Descripteur de configuration	Le type de données MQHCONFIG représente un descripteur de configuration, c'est-à-dire le composant configuré pour un service installable particulier. Un descripteur de configuration doit être aligné sur sa limite naturelle. Les applications ne doivent pas s'appuyer sur le format des données stockées dans ce descripteur. Si elle est valide, sa valeur est destinée à être utilisable dans d'autres appels MQI, mais n'est pas destinée à avoir de signification en dehors de cet objectif.
MQHCONN	Descripteur de connexion	Le type de données MQHCONN représente un descripteur de connexion, c'est-à-dire la connexion à un gestionnaire de files d'attente particulier. Un descripteur de connexion doit être aligné sur une limite de 4 octets. Les applications ne doivent pas s'appuyer sur le format des données stockées dans ce descripteur. Si elle est valide, sa valeur est destinée à être utilisable dans d'autres appels MQI, mais n'est pas destinée à avoir de signification en dehors de cet objectif.
MQHMSG	descripteur de message	Le type de données MQHMSG représente un descripteur de message qui donne accès à un message. Un descripteur de message doit être aligné sur une limite de 8 octets. Les applications ne doivent pas s'appuyer sur le format des données stockées dans ce descripteur. Si elle est valide, sa valeur est destinée à être utilisable dans d'autres appels MQI, mais n'est pas destinée à avoir de signification en dehors de cet objectif.

Nom du type de données élémentaires	Type de données	Description
MQHOBJ	Descripteur d'objet	Le type de données MQHOBJ représente un descripteur d'objet qui donne accès à un objet. Un descripteur d'objet doit être aligné sur une limite de 4 octets. Les applications ne doivent pas s'appuyer sur le format des données stockées dans ce descripteur. Si elle est valide, sa valeur est destinée à être utilisable dans d'autres appels MQI, mais n'est pas destinée à avoir de signification en dehors de cet objectif.
MQINT8	Entier signé 8 bits	Le type de données MQINT8 est un entier signé de 8 bits qui peut prendre n'importe quelle valeur comprise entre -128 et +127, sauf indication contraire du contexte.
MQINT16	Entier signé 16 bits	Le type de données MQINT16 est un entier signé 16 bits qui peut prendre n'importe quelle valeur comprise entre -32 768 et +32 767, sauf indication contraire du contexte. Un MQINT16 doit être aligné sur une limite de 2 octets.
MQINT32	Entier signé 32 bits	Le type de données MQINT32 est un entier binaire signé 32 bits qui peut prendre n'importe quelle valeur comprise entre -2 147 483 648 et + 2 147 483 647, sauf restriction par le contexte. Voir la définition de MQLONG.
MQINT64	Entier signé sur 64 bits	Le type de données MQINT64 est un entier signé de 64 bits qui peut prendre n'importe quelle valeur comprise entre -9 223 372 036 854 775 808 et + 9 223 372 036 854 775 807, sauf restriction par le contexte. Pour COBOL, la plage valide est limitée à -999 999 999 999 999 999 à +999 999 999 999 999 999. Un entier 64 bits doit être aligné sur une limite de 8 octets.

Nom du type de données élémentaires	Type de données	Description
MQLONG	Entier signé 32 bits	Le type de données MQLONG est un entier binaire signé 32 bits qui peut prendre n'importe quelle valeur comprise entre -2 147 483 648 et + 2 147 483 647, sauf restriction par le contexte. Pour COBOL, la plage valide est comprise entre -999 999 999 et +999 999 999. Un MQLONG doit être aligné sur une limite de 4 octets.
MQPID	Identificateur de processus	Identificateur de processus WebSphere MQ . Il s'agit du même identificateur utilisé dans les vidages MQ trace et FFST™ , mais il peut être différent de l'identificateur de processus du système d'exploitation.
MQPTR	Pointeur	Le type de données MQPTR est l'adresse des données de n'importe quel type. Un pointeur doit être aligné sur sa limite naturelle ; il s'agit d'une limite de 16 octets sur IBM i et d'une limite de 8 octets sur d'autres plateformes. Certains langages de programmation prennent en charge les pointeurs typés ; l'interface MQI les utilise également dans quelques cas (par exemple, PMQCHAR et PMQLONG dans le langage de programmation C).
MQTID	Identificateur d'unité d'exécution	Identificateur de l'unité d'exécution WebSphere MQ . Il s'agit du même identificateur utilisé dans les vidages MQ trace et FFST™ , mais il peut être différent de l'identificateur d'unité d'exécution du système d'exploitation.

Nom du type de données élémentaires	Type de données	Description
MQUINT8	Entier non signé 8 bits	Le type de données MQUINT8 est un entier non signé de 8 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +255, sauf indication contraire du contexte.
MQUINT16	Entier non signé 16 bits	Le type de données MQUINT16 est un entier non signé de 16 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +65 535, sauf restriction par le contexte. Une MQUINT16 doit être alignée sur une limite de 2 octets.
MQUINT32	entier non marqué 32-bit	Le type de données MQUINT32 est un entier binaire non signé 32 bits. Voir la définition de MQULONG.
MQUINT64	Entier non signé sur 64 bits	Le type de données MQINT64 est un entier non signé de 64 bits qui peut prendre n'importe quelle valeur comprise entre 0 et +18 446 744 073 709 551 615, sauf restriction par le contexte. Pour COBOL, la plage valide est comprise entre 0 et +999 999 999 999 999 999. Un entier 64 bits doit être aligné sur une limite de 8 octets.
MQULONG	entier non marqué 32-bit	Le type de données MQULONG est un entier binaire non signé de 32 bits qui peut prendre n'importe quelle valeur comprise entre 0 et + 4 294 967 294, sauf restriction par le contexte. Pour COBOL, la plage valide est comprise entre 0 et +999 999 999. Une valeur MQULONG doit être alignée sur une limite de 4 octets.
PMQACH	Pointeur	Pointeur vers une structure de données de type MQACH
PMQAIR	Pointeur	Pointeur vers une structure de données de type MQAIR
PMQAXC	Pointeur	Pointeur vers une structure de données de type MQAXC

Nom du type de données élémentaires	Type de données	Description
PMQAXP	Pointeur	Pointeur vers une structure de données de type MQAXP
PMQBMHO	Pointeur	Pointeur vers une structure de données de type MQBMHO
PMQBO	Pointeur	Pointeur vers une structure de données de type MQBO
PMQBOOL	Pointeur	Pointeur vers des données de type MQBOOL
PMQBYTE	Pointeur	Pointeur vers des données de type MQBYTE
PMQBYTEn	Pointeur	Pointeur vers des données de type MQBYTEn, où n peut être 8, 16, 24, 32, 40, 128
PMQCBC	Pointeur	Pointeur vers une structure de données de type MQCBC
PMQCBD	Pointeur	Pointeur vers une structure de données de type MQCBD
PMQCHAR	Pointeur	Pointeur vers des données de type MQCHAR
PMQCHARN	Pointeur	Pointeur vers un type de données de MQCHARN, où n peut être 4, 8, 12, 20, 28, 32, 48, 64, 128, 256, 264
PMQCHARV	Pointeur	Pointeur vers une structure de données de type MQCHARV
PMQCIH	Pointeur	Pointeur vers une structure de données de type MQCIH
PMQCMHO	Pointeur	Pointeur vers une structure de données de type MQCMHO
PMQCNO	Pointeur	Pointeur vers une structure de données de type MQCNO
PMQCSP	Pointeur	Pointeur vers une structure de données de type MQCSP
PMQCTLO	Pointeur	Pointeur vers une structure de données de type MQCTLO
PMQDH	Pointeur	Pointeur vers une structure de données de type MQDH
PMQDHO	Pointeur	Pointeur vers une structure de données de type MQDHO
PMQDLH	Pointeur	Pointeur vers une structure de données de type MQDLH
PMQDMHO	Pointeur	Pointeur vers une structure de données de type MQDMHO

Nom du type de données élémentaires	Type de données	Description
PMQDMPO	Pointeur	Pointeur vers une structure de données de type MQDMPO
PMQEPH	Pointeur	Pointeur vers une structure de données de type MQEPH
PMQFLOAT32	Pointeur	Pointeur vers une structure de données de type MQFLOAT32
PMQFLOAT64	Pointeur	Pointeur vers une structure de données de type MQFLOAT64
PMQFUNC	Pointeur	Pointeur vers une fonction
PMQGMO	Pointeur	Pointeur vers une structure de données de type MQGMO
PMQHCONFIG	Pointeur	Pointeur vers des données de type MQHCONFIG
PMQHCONN	Pointeur	Pointeur vers des données de type MQHCONN
PMQHMSG	Pointeur	Pointeur vers des données de type MQHMSG
PMQHOBJ	Pointeur	Pointeur vers des données de type MQHOBJ
PMQIIH	Pointeur	Pointeur vers une structure de données de type MQIIH
PMQIMPO	Pointeur	Pointeur vers une structure de données de type MQIMPO
PMQINT8	Pointeur	Pointeur vers des données de type MQINT8
PMQINT16	Pointeur	Pointeur vers des données de type MQINT16
PMQINT32	Pointeur	Pointeur vers des données de type MQINT32
PMQINT64	Pointeur	Pointeur vers des données de type MQINT64
PMQLONG	Pointeur	Pointeur vers des données de type MQLONG
PMQMD	Pointeur	Pointeur vers la structure de type MQMD
PMQMDE	Pointeur	Pointeur vers une structure de données de type MQMDE
PMQMD1	Pointeur	Pointeur vers une structure de données de type MQMD1
PMQMD2	Pointeur	Pointeur vers une structure de données de type MQMD2

Nom du type de données élémentaires	Type de données	Description
PMQMHBO	Pointeur	Pointeur vers une structure de données de type MQMHBO
PMQOD	Pointeur	Pointeur vers une structure de données de type MQOD
PMQOR	Pointeur	Pointeur vers une structure de données de type MQOR
PMQPD	Pointeur	Pointeur vers une structure de données de type MQPD
PMQPID	Pointeur	Pointeur vers un identificateur de processus
PMQMD	Pointeur	Pointeur vers une structure de données de type MQMD
PMQPMO	Pointeur	Pointeur vers une structure de données de type MQPMO
PMQPTR	Pointeur	Pointeur vers des données de type MQPTR
PMQRFH	Pointeur	Pointeur vers une structure de données de type MQRFH
PMQRFH2	Pointeur	Pointeur vers une structure de données de type MQRFH2
PMQRMH	Pointeur	Pointeur vers une structure de données de type MQRMH
PMQRR	Pointeur	Pointeur vers une structure de données de type MQRR
PMQSCO	Pointeur	Pointeur vers une structure de données de type MQSCO
PMQSD	Pointeur	Pointeur vers une structure de données de type MQSD
PMQSMPO	Pointeur	Pointeur vers une structure de données de type MQSMPO
PMQSRO	Pointeur	Pointeur vers une structure de données de type MQSRO
PMSSTS	Pointeur	Pointeur vers une structure de données de type MQSTS
PMQTID	Pointeur	Pointeur vers un ID d'unité d'exécution
PMQTM	Pointeur	Pointeur vers une structure de données de type MQTM
PMQTMC2	Pointeur	Pointeur vers une structure de données de type MQTMC2
PMQUINT8	Pointeur	Pointeur vers un type de données MQUINT8

Nom du type de données élémentaires	Type de données	Description
PMQUINT16	Pointeur	Pointeur vers un type de données de MQUINT16
PMQUINT32	Pointeur	Pointeur vers un type de données de MQUINT32
PMQUINT64	Pointeur	Pointeur vers un type de données de MQUINT64
PPMQULONG	Pointeur	Pointeur vers un type de données de MQULONG
ID PMQVOID	Pointeur	
PMQWIH	Pointeur	Pointeur vers une structure de données de type MQWIH
PMQXQH	Pointeur	Pointeur vers une structure de données de type MQXQH

Déclarations C

Type de données	Représentation
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];`

Type de données	Représentation
MQCHAR32	`typedef MQCHAR MQCHAR32[32];`
MQCHAR48	`typedef MQCHAR MQCHAR48[48];`
MQCHAR64	`typedef MQCHAR MQCHAR64[64];`
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	Sur les systèmes UNIX 64 bits:
	`typedef long;`
	Sous AIX, Solaris et HP-UX32 bits:
	`typedef int64_t;`
	Sous IBM i, Linuxet z/OS:
	`typedef long long;`
	Sous Windows:
	`typedef _int64;`

MQLONG	Sous IBM i :
	`typedef long MQLONG;`
	autres plateformes:
	`if defined(MQ_64_BIT)`
	`  typedef int MQLONG;`
	`else`
	`  typedef long MQLONG;`

Type de données	Représentation
MQPID	`typedef MQLONG MQPID;`
MQPTR	`typedef void MQPOINTER MQPTR;`
MQTID	`typedef MQLONG MQTID;`
MQUINT8	`typedef unsigned char MQUINT8;`
MQUINT16	`typedef unsigned short MQUINT16;`

MQUINT64	Sur les systèmes UNIX 64 bits:
	`typedef unsigned long;`
	Sous AIX, Solaris et HP-UX32 bits:
	`typedef uint64_t;`
	Sous IBM i, Linuxet z/OS:
	`typedef unsigned long long;`
	Sous Windows:
	`typedef unsigned _int64;`

MQULONG	Sous IBM i :
	`typedef unsigned long MQULONG;`
	autres plateformes:
	`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];`

Type de données	Représentation
PMQBYTE40	`typedef MQBYTE40[40] MQPOINTER PMQBYTE40[40];`
PMQBYTE128	`typedef MQBYTE128[128] MQPOINTER PMQBYTE128[128];`
PMQCHAR	`typedef MQCHAR MQPOINTER PMQCHAR;`
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;`

Type de données	Représentation
PMQHCONFIG	`typedef MQHCONFIG MQPOINTER PMQHCONFIG;`
PMQHCONN	`typedef MQHCONN MQPOINTER PMQHCONN;`
PMQHOBJ	`typedef MQHOBJ MQPOINTER PMQHOBJ;`
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;`
PPMQULONG	`typedef MQULONG MQPOINTER PMQULONG;`

Type de données	Représentation
ID PMQVOID	`typedef void MQPOINTER PMQVOID;`
PMQWIH	`typedef MQWIH MQPOINTER PMQWIH;`
PMQXQH	`typedef MQXQH MQPOINTER PMQXQH;`
PPMQBO	`typedef PMQBO MQPOINTER PPMQBO;`
PPMQBYTE (octet)	`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;`
Où `defined(MQ_64_BIT)` correspond à une plateforme 64 bits.	

Pour une description de la variable de macro MQPOINTER, voir «types de données», à la page 245 .

Déclarations COBOL

Type de données	Représentation
MQBOOL	`PIC S9(9) BINARY`
MQBYTE	`PIC X`

Type de données	Représentation
MQBYTE8	`PIC X(8)`
MQBYTE16	`PIC X(16)`
MQBYTE24	`PIC X(24)`
MQBYTE32	`PIC X(32)`
MQBYTE40	`PIC X(40)`
MQCHAR	`PIC X`
MQCHAR4	`PIC X(4)`
MQCHAR8	`PIC X(8)`
MQCHAR12	`PIC X(12)`
MQCHAR20	`PIC X(20)`
MQCHAR28	`PIC X(28)`
MQCHAR32	`PIC X(32)`
MQCHAR48	`PIC X(48)`
MQCHAR64	`PIC X(64)`
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`

Type de données	Représentation
MQINT16	`PIC S9(4) BINARY`
MQINT64	`PIC S9(18) BINARY`
MQLONG	`PIC S9(9) BINARY`
MQPTR	`POINTER`
MQUINT8	`PIC 9(2) BINARY`
MQUINT16	`PIC 9(4) BINARY`
MQUINT64	`PIC 9(18) BINARY`
MQULONG	`PIC 9(9) BINARY`

Déclarations PL/I
PL/I est pris en charge sur z/OS.

Type de données	Représentation
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)`

Type de données	Représentation
MQCHAR20	`char(20)`
MQCHAR28	`char(28)`
MQCHAR32	`char(32)`
MQCHAR48	`char(48)`
MQCHAR64	`char(64)`
MQCHAR128	`char(128)`
MQCHAR256	`char(256)`
MQFLOAT32	`binary float(21) ieee`
MQFLOAT64	`binary float(52) ieee`
MQHCONN	`fixed bin(31)`
MQHOBJ	`fixed bin(31)`
MQINT8	`fixed bin(7)`
MQINT16	`fixed bin(15)`
MQINT64	`fixed bin(63)`
MQLONG	`fixed bin(31)`
MQPTR	`pointer`
MQUINT8	`fixed bin(8)`
MQUINT16	`fixed bin(16)`
MQUINT64	`fixed bin(64)`
MQULONG	`fixed bin(32)`

Déclarations assembleur System/390
L'assembleur System/390 est pris en charge sur z/OS uniquement.

Type de données	Représentation
MQBOOL	`DS F`
MQBYTE	`DS XL1`
MQBYTE8	`DS XL8`
MQBYTE16	`DS XL16`
MQBYTE24	`DS XL24`
MQBYTE32	`DS XL32`
MQBYTE40	`DS XL40`
MQCHAR	`DS CL1`
MQCHAR4	`DS CL4`
MQCHAR8	`DS CL8`
MQCHAR12	`DS CL12`
MQCHAR20	`DS CL20`
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`

Type de données	Représentation
MQHOBJ	`DS F`
MQINT8	`DS XL1`
MQINT16	`DS H`
MQINT64	`DS D`
MQLONG	`DS F`
MQPTR	`DS F`
MQUINT8	`DS XL1`
MQUINT16	`DS H`
MQUINT64	`DS D`
MQULONG	`DS F`

Types de données de structure-Introduction

Cette section présente les types de données de structure utilisés dans l'interface MQI. Les types de données de structure eux-mêmes sont décrits dans les sections suivantes.

Récapitulatif

Les tableaux suivants récapitulent les types de données de structure utilisés dans l'interface MQI.

Tableau 467. Types de données de structure utilisés sur les appels MQI (ou les fonctions d'exit):

Structure	Description	Appels utilisés
MQACH	En-tête de chaîne d'exit d'API	
MQAIR	Enregistrement d'informations d'authentification	MQCONNX
MQAXC	Contexte d'exit d'API	
MQAXP	Paramètre d'exit API	
MQBMHO	Options de traitement de la mémoire tampon vers le message	MQBUFMH
MQBO	Options de début	MQBEGIN
MQCBD	Descripteur de rappel	MQCB
MQCBO	Options de création de sac	Sac mqCreate
MQCHARV	Chaîne de longueur variable	MQINQMP
MQCNO	Options de connexion	MQCONNX

Tableau 467. Types de données de structure utilisés sur les appels MQI (ou les fonctions d'exit): (suite)		
Structure	**Description**	**Appels utilisés**
MQCSP	Paramètres de sécurité	MQCONNX
MQCTLO	Options de rappel	MQCTL
MQDMPO	Options de suppression de propriété de message	MQDLTMP
MQGMO	Options de message Get	MQGet
MQIMPO	Options de propriété de message d'interrogation	MQINQMP
MQMD	Descripteur de message	MQBUFMH, MQMHBUF, MQCB, MQGET, MQPUT, MQPUT1
MQMHBO	Options de traitement des messages vers la mémoire tampon	MQMHBUF
MQOD	Descripteur d'objet	MQOPEN, MQPUT1
MQOR	Enregistrement d'objet	MQOPEN, MQPUT1
MQPD	Descripteur de propriété	MQSETMP
MQPMO	Options de message Put	MQPUT, MQPUT1
MQPMR	Enregistrement d'insertion de message	MQPUT, MQPUT1
MQRR	Enregistrement de réponse	MQOPEN, MQPUT, MQPUT1
MQSCO	Options de configuration SSL	MQCONNX
MQSD	Descripteur d'abonnement	MQSUB
MQSMPO	Définir l'option de propriété de message	MQSETMP
MQSRO	Options de demande d'abonnement	MQSUBRQ
MQSTS	Structure de génération de rapports de statut	MQSTAT

Tableau 468. Types de données de structure utilisés dans les données de message:	
Structure	**Description**
MQCIH	En-tête d'informations CICS
MQCFH	En-tête PCF
MQEPH	En-tête PCF imbriqué
MQDH	En-tête de distribution
MQDLH	En-tête de rebut (message non distribué)
MQIIH	En-tête d'informations IMS
MQMDE	Extension du descripteur de message
MQRFH	Règles et en-tête de formatage

Tableau 468. Types de données de structure utilisés dans les données de message: (suite)	
Structure	**Description**
MQRFH2	Règles et en-tête de formatage 2
MQRMH	En-tête de message de référence
MQTM	Message de déclenchement
MQTMC2	Message de déclenchement (format de caractère 2)
MQWIH	En-tête d'informations de travail
MQXQH	En-tête de file d'attente de transmission

Remarque : La structure MQDXP (paramètre d'exit de conversion de données) est décrite dans «Exit de conversion de données», à la page 898, avec les appels de conversion de données associés.

Règles pour les types de données de structure

Les langages de programmation varient dans leur niveau de prise en charge des structures, et certaines règles et conventions sont adoptées pour mapper les structures MQI de manière cohérente dans chaque langage de programmation:

1. Les structures doivent être alignées sur leurs limites naturelles.
   - La plupart des structures MQI nécessitent un alignement sur 4 octets.
   - Sous IBM i, les structures contenant des pointeurs requièrent un alignement sur 16 octets ; il s'agit de MQCNO, MQOD, MQPMO.
2. Chaque zone d'une structure doit être alignée sur sa limite naturelle.
   - Les zones dont les types de données sont équivalents à MQLONG doivent être alignées sur des limites de 4 octets.
   - Les zones dont les types de données sont équivalents à MQPTR doivent être alignées sur des limites de 16 octets sur IBM iet sur des limites de 4 octets dans d'autres environnements.
   - Les autres zones sont alignées sur des limites de 1 octet.
3. La longueur d'une structure doit être un multiple de son alignement de limite.
   - La plupart des structures MQI ont des longueurs multiples de 4 octets.
   - Sous IBM i, les structures contenant des pointeurs ont des longueurs multiples de 16 octets.
4. Si nécessaire, des octets ou des zones de remplissage doivent être ajoutés pour garantir le respect des règles ci-dessus.

Conventions utilisées dans les descriptions

La description de chaque type de données de structure comprend:

- Un aperçu de l'objectif et de l'utilisation de la structure
- Description des zones de la structure, sous une forme indépendante du langage de programmation
- Exemples de déclaration de la structure dans chacun des langages de programmation pris en charge

La description de chaque type de données de structure contient les sections suivantes:

Nom de structure
Nom de la structure, suivi d'un récapitulatif des zones de la structure.

Présentation
Brève description de l'objet et de l'utilisation de la structure.

Zones
Description des zones. Pour chaque champ, le nom du champ est suivi de son type de données élémentaires entre parenthèses (). Dans le texte, les noms de zone sont affichés en italique ; par exemple, *Version*.

Il contient également une description de l'objectif de la zone, ainsi qu'une liste de toutes les valeurs que la zone peut prendre. Les noms des constantes sont affichés en majuscules ; par exemple, MQGMO_STRUC_ID. Un ensemble de constantes ayant le même préfixe est affiché à l'aide du caractère *, par exemple: MQIA_ *.

Dans les descriptions des zones, les termes suivants sont utilisés:

entrée
Vous fournissez des informations dans la zone lorsque vous effectuez un appel.

sortie
Le gestionnaire de files d'attente renvoie des informations dans la zone lorsque l'appel se termine ou échoue.

entrée-sortie
Vous fournissez des informations dans la zone lorsque vous effectuez un appel et le gestionnaire de files d'attente modifie les informations lorsque l'appel se termine ou échoue.

Valeurs initiales
Tableau présentant les valeurs initiales de chaque zone dans les fichiers de définition de données fournis avec l'interface MQI.

Déclaration C
Déclaration typique de la structure en C.

Déclaration COBOL
Déclaration standard de la structure en COBOL.

Déclaration PL/I
Déclaration typique de la structure en PL/I.

Déclaration assembleur System/390
Déclaration standard de la structure dans le langage assembleur System/390 .

Déclaration Visual Basic
Déclaration typique de la structure dans Visual Basic.

Programmation C
Cette section contient des informations pour vous aider à utiliser l'interface MQI du langage de programmation C.

Fichiers d'en-tête
Des fichiers d'en-tête sont fournis pour vous aider à écrire des programmes d'application C qui utilisent l'interface MQI.

Ces fichiers d'en-tête sont récapitulés dans Tableau 469, à la page 244.

Tableau 469. Fichiers d'en-tête C

Fichier	Contenu
CMQC	Prototypes de fonction, types de données et constantes nommées pour l'interface MQI principale
CMQXC	Prototypes de fonction, types de données et constantes nommées pour l'exit de conversion de données
CMQEC	Prototypes de fonction, types de données et constantes nommées pour l'interface MQI principale, l'exit de conversion de données et la structure des points d'entrée d'interface (CMQEC inclut CMQXC et CMQC).

Pour améliorer la portabilité des applications, codez le nom du fichier d'en-tête en minuscules sur la directive de préprocesseur `#include` :

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

Fonctions
Il n'est pas nécessaire de spécifier tous les paramètres transmis par l'adresse chaque fois que vous appelez une fonction.

- Transmettez les paramètres *d'entrée uniquement* et de type MQHCONN, MQHOBJ ou MQLONG par valeur.
- Transmettez tous les autres paramètres par adresse.

Lorsqu'un paramètre particulier n'est pas requis, utilisez un pointeur null comme paramètre sur l'appel de fonction, à la place de l'adresse des données de paramètre. Les paramètres pour lesquels cela est possible sont identifiés dans les descriptions d'appel.

Aucun paramètre n'est renvoyé comme valeur de la fonction ; dans la terminologie C, cela signifie que toutes les fonctions renvoient `void`.

Les attributs de la fonction sont définis par la variable de macro MQENTRY ; la valeur de cette variable de macro dépend de l'environnement.

Paramètres avec type de données non défini

Le paramètre *`Buffer`* des fonctions MQGET, MQPUT et MQPUT1 a un type de données non défini. Ce paramètre est utilisé pour envoyer et recevoir les données de message de l'application.

Les paramètres de ce type sont présentés dans les exemples C sous la forme de tableaux de MQBYTE. Vous pouvez déclarer les paramètres de cette manière, mais il est généralement plus pratique de les déclarer comme la structure particulière qui décrit la présentation des données dans le message. Déclarez le paramètre de fonction réel en tant que pointeur vers vide et indiquez l'adresse de n'importe quel type de données en tant que paramètre lors de l'appel de fonction.

types de données

Définissez tous les types de données à l'aide de l'instruction C `typedef` . Pour chaque type de données, définissez également le type de données de pointeur correspondant. Le nom du type de données de pointeur est le nom du type de données élémentaire ou de structure précédé de la lettre P pour désigner un pointeur. Définissez les attributs du pointeur à l'aide de la variable de macro MQPOINTER ; la valeur de cette variable de macro dépend de l'environnement. L'exemple suivant montre comment déclarer des types de données de pointeur:

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

Manipulation des chaînes binaires
Déclarez des chaînes de données binaires comme l'un des types de données MQBYTEn.

Chaque fois que vous copiez, comparez ou définissez des zones de ce type, utilisez les fonctions C `memcpy`, `memcmp`ou `memset`; par exemple:

```
#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));
```

N'utilisez pas les fonctions de chaîne `strcpy`, `strcmp`, `strncpy`ou `strncmp`, car elles ne fonctionnent pas correctement pour les données déclarées avec les types de données MQBYTEn.

Manipulation des chaînes de caractères
Lorsque le gestionnaire de files d'attente renvoie des données de type caractère à l'application, il remplit toujours les données de type caractère avec des blancs à la longueur définie de la zone ; le gestionnaire de files d'attente *ne renvoie pas* de chaînes à terminaison nulle.

Par conséquent, lors de la copie, de la comparaison ou de la concaténation de ces chaînes, utilisez les fonctions de chaîne `strncpy`, `strncmp`ou `strncat`.

N'utilisez pas les fonctions de chaîne qui nécessitent que la chaîne se termine par une valeur nulle (`strcpy`, `strcmp`, `strcat`). N'utilisez pas non plus la fonction `strlen` pour déterminer la longueur de la chaîne ; utilisez plutôt la fonction `sizeof` pour déterminer la longueur de la zone.

Valeurs initiales pour les structures
Les fichiers d'en-tête définissent diverses variables de macro que vous pouvez utiliser pour fournir des valeurs initiales pour les structures MQ lorsque vous déclarez des instances de ces structures.

Ces variables macro ont des noms de la forme MQxxx_DEFAULT, où MQxxx représente le nom de la structure. Ils sont utilisés de la manière suivante:

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

Pour certaines zones de caractères (par exemple, les zones *StrucId* qui apparaissent dans la plupart des structures ou la zone *Format* qui apparaît dans MQMD), l'interface MQI définit des valeurs particulières qui sont valides. Pour chacune des valeurs valides, *deux* variables macro sont fournies:

- Une variable macro définit la valeur sous la forme d'une chaîne avec une longueur, en excluant les correspondances nulles implicites, exactement la longueur définie de la zone. Par exemple, pour la zone *Format* dans MQMD, la variable de macro suivante est fournie (¬ représente un caractère blanc):

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

  Utilisez ce formulaire avec les fonctions `memcpy` et `memcmp` .
- L'autre variable de macro définit la valeur comme un tableau de caractères ; le nom de cette variable de macro est le nom de la forme de chaîne suffixé avec _ARRAY. Exemple :

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

  Utilisez ce formulaire pour initialiser la zone lorsque vous déclarez une instance de la structure avec des valeurs différentes de celles fournies par la variable macro MQMD_DEFAULT. (Ce n'est pas toujours nécessaire ; dans certains environnements, vous pouvez utiliser la forme de chaîne de la valeur dans les deux situations. Toutefois, vous pouvez utiliser le formulaire de tableau pour les déclarations, car cela est requis pour la compatibilité avec le langage de programmation C + +.)

Valeurs initiales pour les structures dynamiques

Lorsqu'un nombre variable d'instances d'une structure est requis, les instances sont généralement créées dans la mémoire principale obtenue dynamiquement à l'aide des fonctions `calloc` ou `malloc` . Pour initialiser les zones dans de telles structures, tenez compte de la technique suivante:

1. Déclarez une instance de la structure à l'aide de la variable de macro MQxxx_DEFAULT appropriée pour initialiser la structure. Cette instance devient le modèle pour les autres instances:

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

Les mots clés `static` ou `auto` peuvent être codés sur la déclaration afin de donner à l'instance de modèle une durée de vie statique ou dynamique, selon les besoins.

2. Utilisez les fonctions `calloc` ou `malloc` pour obtenir du stockage pour une instance dynamique de la structure:

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

3. Utilisez la fonction `memcpy` pour copier l'instance de modèle dans l'instance dynamique:

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

Utiliser à partir de C++

Pour le langage de programmation C + +, les fichiers d'en-tête contiennent les instructions supplémentaires suivantes qui sont incluses uniquement lorsque vous utilisez un compilateur C + +:

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

/* rest of header file */

#ifdef __cplusplus
  }
#endif
```

Conventions de notation
Ces informations montrent comment appeler les fonctions et déclarer des paramètres.

Dans certains cas, les paramètres sont des tableaux dont la taille n'est pas fixe. Pour ceux-ci, un n minuscule est utilisé pour représenter une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez le n par la valeur numérique requise.

Programmation COBOL

Cette section contient des informations pour vous aider à utiliser l'interface MQI du langage de programmation COBOL.

Copier des fichiers
Divers fichiers COPY sont fournis pour vous aider à écrire des programmes d'application COBOL qui utilisent l'interface MQI. Il existe deux fichiers contenant des constantes nommées et deux fichiers pour chacune des structures.

Chaque structure est fournie sous deux formes: une forme avec des valeurs initiales, et une forme sans:

- Utilisez les structures avec les valeurs initiales de la SECTION WORKING-STORAGE d'un programme COBOL ; elles sont contenues dans des fichiers COPY dont les noms sont suffixés par la lettre V (pour les valeurs).
- Utilisez les structures sans valeurs initiales dans le `LINKAGE SECTION` d'un programme COBOL ; elles sont contenues dans des fichiers COPY dont les noms sont suffixés par la lettre L (pour Linkage).

Les fichiers COPY sont récapitulés dans Tableau 470, à la page 247. Tous les fichiers répertoriés ne sont pas disponibles dans tous les environnements.

Tableau 470. Fichiers COBOL COPY

Fichier (avec les valeurs initiales)	**Fichier (sans valeurs initiales)**	**Contenu**
CMQAIRV	CMQAIRL	Enregistrement d'informations d'authentification

Tableau 470. Fichiers COBOL COPY (suite)		
Fichier (avec les valeurs initiales)	**Fichier (sans valeurs initiales)**	**Contenu**
CMQBOV	CMQBOL	Structure des options de début
CMQCIHV	CMQCIHL	Structure d'en-tête d'informations CICS
CMQCNOV	CMQCNOL	Structure des options de connexion
CMQDHV	CMQDHL	Structure d'en-tête de distribution
CMQDLHV	CMQDLHL	Structure d'en-tête de rebut
CMQDXPV	CMQDXPL	Structure des paramètres d'exit de conversion de données
CMQGMOV	CMQGMOL	Obtenir la structure des options de message
CMQIIHV	CMQIIHL	Structure d'en-tête d'informations IMS
CMQMDV	CMQMDL	Structure de descripteur de message
CMQMDEV	CMQMDEL	Structure d'extension du descripteur de message
CMQMD1V	CMQMD1L	Structure de descripteur de message version 1
CMQODV	CMQODL	Structure de descripteur d'objet
CMQORV	CMQORL	Structure d'enregistrement d'objet
CMQPMOV	CMQPMOL	Structure des options de message d'insertion
CMQRFHV	CMQRFHL	Règles et structure d'en-tête de formatage
CMQRFH2V	CMQRFH2L	Règles et formatage de la structure d'en-tête version 2
CMQRMHV	CMQRMHL	Structure d'en-tête de message de référence
CMQRRV	CMQRRL	Structure d'enregistrement de réponse
CMQSCOV	CMQSCOL	Options de configuration SSL
CMQTMV	CMQTML	Structure des messages de déclenchement
CMQTMCV	CMQTMCL	Structure du message de déclenchement (format de caractères)
CMQTMC2V	CMQTMC2L	Structure du message de déclenchement (format de caractère) version 2
CMQWIHV	CMQWIHL	Structure d'en-tête des informations de travail
CMQXQHV	CMQXQHL	Structure d'en-tête de file d'attente de transmission
CMQV	-	Constantes nommées pour l'interface MQI principale
CMQXV	-	Constantes nommées pour l'exit de conversion de données
CMQMD2V	CMQMD2L	Structure de descripteur de message version 2

structures

Dans le fichier COPY, chaque déclaration de structure commence par un élément level-10 ; cela vous permet de déclarer plusieurs instances de la structure en codant la déclaration level-01 , puis en utilisant l'instruction COPY pour copier le reste de la déclaration de structure. Pour faire référence à l'instance appropriée, utilisez le mot clé IN :

```
* 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.
```

Alignez les structures sur les limites appropriées. Si vous utilisez l'instruction COPY pour inclure une structure à la suite d'un élément qui n'est pas l'élément level-01 , assurez-vous que la structure commence au décalage approprié par rapport au début de l'élément level-01 . La plupart des structures MQI nécessitent un alignement à 4 octets ; les exceptions à cela sont MQCNO, MQOD et MQPMO, qui requièrent un alignement à 16 octets sur IBM i.

Dans cette section, les noms des zones des structures sont affichés sans préfixe. En COBOL, les noms de zone sont précédés du nom de la structure suivi d'un trait d'union. Toutefois, si le nom de la structure se termine par un chiffre, indiquant que la structure est une deuxième version ou une version ultérieure de la structure d'origine, le chiffre numérique est omis du préfixe. Les noms de zone en COBOL sont affichés en majuscules (bien que des minuscules ou une casse mixte puissent être utilisées si nécessaire). Par exemple, la zone *MsgType* décrite dans «MQMD-Descripteur de message», à la page 394 devient MQMD-MSGTYPE en COBOL.

Les structures de suffixe V sont déclarées avec des valeurs initiales pour toutes les zones ; vous devez définir uniquement les zones dans lesquelles vous souhaitez une valeur différente de la valeur initiale fournie.

Pointeurs
Certaines structures doivent traiter des données facultatives qui peuvent être discontiguës à la structure, telles que les enregistrements MQOR et MQRR traités par la structure MQOD.

Pour traiter ces données facultatives, les structures contiennent des zones qui sont déclarées avec le type de données de pointeur. Cependant, COBOL ne prend pas en charge le type de données de pointeur dans tous les environnements. Pour cette raison, les données facultatives peuvent également être traitées à l'aide de zones qui contiennent le décalage des données à partir du début de la structure.

Si vous souhaitez porter une application entre des environnements, déterminez si le type de données de pointeur est disponible dans tous les environnements prévus. Si ce n'est pas le cas, l'application doit traiter les données facultatives à l'aide des zones de décalage au lieu des zones de pointeur.

Dans les environnements où les pointeurs ne sont pas pris en charge, déclarez les zones de pointeur en tant que chaînes d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null. Ne modifiez pas cette valeur initiale si vous utilisez les zones de décalage.

Constante nommée

Dans cette section, les noms des constantes sont affichés avec le caractère de soulignement (_) dans le nom. En COBOL, utilisez le trait d'union (-) à la place du trait de soulignement.

Les constantes qui ont des valeurs de chaîne de caractères utilisent le guillemet simple comme délimiteur de chaîne ('). Dans certains environnements, vous devrez peut-être spécifier une option de compilateur appropriée pour que le compilateur accepte le guillemet simple comme délimiteur de chaîne à la place du guillemet double.

Les constantes nommées sont déclarées dans les fichiers COPY en tant qu'éléments level-10 . Pour utiliser les constantes, déclarez explicitement l'élément level-01 , puis utilisez l'instruction COPY pour copier les déclarations des constantes:

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

La méthode précédente permet aux constantes d'occuper la mémoire du programme même si elles ne sont pas référencées. Si vous incluez les constantes dans de nombreux programmes distincts au sein

d'une même unité d'exécution, il existe plusieurs copies des constantes, ce qui consomme inutilement de la mémoire principale. Evitez cet effet en utilisant l'une des techniques suivantes:

- Ajoutez la clause GLOBAL à la déclaration level-01 :

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

  Cela entraîne l'allocation de mémoire pour un seul ensemble de constantes au sein de l'unité d'exécution. Toutefois, les constantes peuvent être référencées par n'importe quel programme au sein de l'unité d'exécution, et pas seulement par le programme qui contient la déclaration level-01 .

  Remarque : La clause GLOBAL n'est pas prise en charge dans tous les environnements.
- Copiez manuellement dans chaque programme uniquement les constantes référencées par ce programme. N'utilisez pas l'instruction COPY pour copier toutes les constantes dans le programme.

Conventions de notation

Les dernières rubriques de cette section montrent comment appeler les appels et déclarer les paramètres. Dans certains cas, les paramètres sont des tables ou des chaînes de caractères dont la taille n'est pas fixe. Pour ceux-ci, un n minuscule est utilisé pour représenter une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez le n par la valeur numérique requise.

Programmation en assembleur System/390

Cette section contient des informations pour vous aider à utiliser l'interface MQI du langage de programmation System/390 Assembler.

Macros

Différentes macros sont fournies pour vous aider à écrire des programmes d'application assembleur qui utilisent l'interface MQI.

Il existe deux macros pour les constantes nommées et une macro pour chacune des structures. Ces fichiers sont récapitulés dans Tableau 471, à la page 250.

Tableau 471. Macros assembleur

Fichier	**Contenu**
CMQA	Constantes nommées (égales) pour l'interface MQI principale
CMQCIHA	Structure d'en-tête d'informations CICS
CMQCNOA	Structure des options de connexion
CMQDLHA	Structure d'en-tête de rebut
CMQDXPA	Structure des paramètres d'exit de conversion de données
CMQGMOA	Obtenir la structure des options de message
CMQIIHA	Structure d'en-tête d'informations IMS
CMQMDA	Structure de descripteur de message
CMQMDEA	Structure d'extension du descripteur de message
CMQODA	Structure de descripteur d'objet
CMQPMOA	Structure des options de message d'insertion
CMQRFHA	Règles et structure d'en-tête de formatage
CMQRFH2A	Règles et formatage de la structure d'en-tête version 2
CMQRMHA	Structure d'en-tête de message de référence

Tableau 471. Macros assembleur (suite)	
Fichier	**Contenu**
Intervalle CMQTMA	Structure des messages de déclenchement
CMQTMC2A	Structure du message de déclenchement (format de caractère) version 2
CMQVERA	Contrôle de version de structure
CMQWIHA	Structure d'en-tête des informations de travail
CMQXA	Constantes nommées pour l'exit de conversion de données
CMQXPA	Structure des paramètres d'exit de croisement d'API
CMQXQHA	Structure d'en-tête de file d'attente de transmission

structures
Les structures sont générées par des macros qui ont différents paramètres pour contrôler l'action de la macro. Ces paramètres sont décrits dans les sections suivantes.

De temps à autre, de nouvelles versions des structures MQ sont introduites. Les zones supplémentaires d'une nouvelle version peuvent faire en sorte qu'une structure de moins de 256 octets devienne supérieure à 256 octets. Pour cette raison, écrivez des instructions d'assembleur destinées à copier une structure MQ ou à définir une structure MQ sur des valeurs nulles, afin de fonctionner correctement avec des structures pouvant dépasser 256 octets. Vous pouvez également utiliser le paramètre de macro DCLVER ou la macro CMQVERA avec le paramètre VERSION pour déclarer une version spécifique de la structure.

Spécification du nom de la structure
Pour déclarer plusieurs instances d'une structure, la macro préfixe le nom de chaque zone de la structure avec une chaîne spécifique à l'utilisateur et un trait de soulignement.

La chaîne utilisée est le libellé spécifié lors de l'appel de la macro. Si aucun libellé n'est spécifié, le nom de la structure est utilisé pour construire le préfixe:

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

Les déclarations de structure affichées dans cette section utilisent le préfixe par défaut.

Spécification de la forme de la structure

Les déclarations de structure peuvent être générées par la macro dans l'une des deux formes, contrôlées par le paramètre DSECT :

DSECT = OUI
Une instruction DSECT assembleur est utilisée pour démarrer une nouvelle section de données ; la définition de structure suit immédiatement l'instruction DSECT . Le libellé de l'appel de macro est utilisé comme nom de la section de données ; si aucun libellé n'est spécifié, le nom de la structure est utilisé.

DSECT = NON
Les instructions DC de l'assembleur permettent de définir la structure à la position en cours dans la routine. Les zones sont initialisées avec des valeurs, qui peuvent être spécifiées en codant les paramètres appropriés lors de l'appel de la macro. Les zones pour lesquelles aucune valeur n'est spécifiée dans l'appel de macro sont initialisées avec des valeurs par défaut.

La valeur indiquée doit être en majuscules. Si le paramètre DSECT n'est pas spécifié, DSECT=NO est supposé.

Contrôle de la version de la structure
Par défaut, les macros déclarent toujours la version la plus récente de chaque structure.

Bien que vous puissiez utiliser le paramètre de macro VERSION pour spécifier une valeur pour la zone *Version* dans la structure, ce paramètre définit la valeur initiale de la zone *Version* et ne contrôle pas la version de la structure réellement déclarée. Pour contrôler la version de la structure déclarée, utilisez le paramètre DCLVER :

DCLVER=EN COURS
La version déclarée est la version en cours (la plus récente).

DCLVER=SPECIFIE
La version déclarée est la version spécifiée par le paramètre VERSION . Si vous omettez le paramètre VERSION , la version par défaut est la version 1.

Si vous spécifiez le paramètre VERSION , la valeur doit être une constante numérique auto-définie ou la constante nommée pour la version requise (par exemple, MQCNO_VERSION_3). Si vous spécifiez une autre valeur, la structure est déclarée comme si DCLVER=CURRENT avait été spécifié, même si la valeur de VERSION est résolue en une valeur valide.

La valeur indiquée doit être en majuscules. Si vous omettez le paramètre DCLVER , la valeur utilisée est extraite de la variable macro globale MQDCLVER . Vous pouvez définir cette variable à l'aide de la macro CMQVERA.

Déclaration d'une structure imbriquée dans une autre

Pour déclarer une structure en tant que composant d'une autre structure, utilisez le paramètre NESTED :

NESTED=YES
La déclaration de structure est imbriquée dans une autre.

NESTED=NO
La déclaration de structure n'est pas imbriquée dans une autre.

La valeur indiquée doit être en majuscules. Si vous omettez le paramètre NESTED , NESTED=NO est supposé.

Spécification de valeurs initiales pour les champs
Indiquez la valeur à utiliser pour initialiser une zone dans une structure en codant le nom de cette zone (sans le préfixe) en tant que paramètre dans l'appel de macro, accompagné de la valeur requise.

Par exemple, pour déclarer une structure de descripteur de message avec la zone *MsgType* initialisée avec MQMT_REQUEST et la zone *ReplyToQ* initialisée avec la chaîne "MY_REPLY_TO_QUEUE", utilisez ce qui suit:

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

Si vous spécifiez une constante nommée (égale) comme valeur lors de l'appel de la macro, utilisez la macro CMQA pour définir la constante nommée. Ne placez pas les valeurs de chaîne de caractères entre apostrophes.

Contrôle de la liste

Contrôlez l'apparence de la déclaration de structure dans la liste de l'assembleur à l'aide du paramètre LIST :

LISTE = OUI
La déclaration de structure apparaît dans la liste de l'assembleur.

LISTE = NON
La déclaration de structure n'apparaît pas dans la liste de l'assembleur.

La valeur indiquée doit être en majuscules. Si vous omettez le paramètre LIST , LIST=NO est supposé.

Macro MQVERA

Cette macro permet de définir la valeur par défaut à utiliser pour le paramètre DCLVER dans les macros de structure. La valeur spécifiée par CMQVERA est utilisée par la macro de structure uniquement si vous

omettez le paramètre DCLVER dans l'appel de la macro de structure. La valeur par défaut est définie en codant la macro CMQVERA avec le paramètre DCLVER :

DCLVER=EN COURS
La version par défaut est définie sur la version en cours (la plus récente).

DCLVER=SPECIFIE
La version par défaut correspond à la version spécifiée par le paramètre VERSION .

Vous devez spécifier le paramètre DCLVER et la valeur doit être en majuscules. La valeur définie par CMQVERA reste la valeur par défaut jusqu'au prochain appel de CMQVERA ou jusqu'à la fin de l'assemblage. Si vous omettez CMQVERA, la valeur par défaut est DCLVER=CURRENT.

Conventions de notation

Les sections suivantes montrent comment appeler les appels et déclarer les paramètres. Dans certains cas, les paramètres sont des tableaux ou des chaînes de caractères dont la taille n'est pas fixe et pour lesquels un n minuscule est utilisé pour représenter une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez le n par la valeur numérique requise.

MQAIR-Enregistrement des informations d'authentification

La structure MQAIR représente l'enregistrement des informations d'authentification.

Le tableau suivant récapitule les zones de la structure.

Tableau 472. Zones dans MQAIR

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
AuthInfoType	Type des informations d'authentification	AuthInfoType
AuthInfoConnName	Nom de connexion du serveur CRL LDAP	AuthInfoConnName
LDAPUserNamePtr	Adresse du nom d'utilisateur LDAP	LDAPUserNamePtr
LDAPUserNameDécalage	Délalage du nom d'utilisateur LDAP à partir du début de MQSCO	DécalageLDAPUserName
LDAPUserNameLongueur	Longueur du nom d'utilisateur LDAP	LDAPUserNameLongueur
LDAPPassword	Mot de passe d'accès au serveur LDAP	LDAPPassword
Remarque : Les zones restantes sont ignorées si *Version* est inférieure à MQAIR_VERSION_2.		
OCSPResponderURL	URL à laquelle le répondeur OCSP peut être contacté	OCSPResponderURL

Présentation de MQAIR

La structure MQAIR permet à une application s'exécutant en tant que client WebSphere MQ MQI de spécifier des informations sur un authentificateur à utiliser pour la connexion client. La structure est un paramètre d'entrée dans l'appel MQCONNX.

Disponibilité: clients AIX, HP-UX, Solaris, Linux et Windows .

Jeu de caractères et codage: les données de MQAIR doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente **CodedCharSetId** et MQENC_NATIVE.

Zones pour MQAIR

La structure MQAIR contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

AuthInfoConnName (MQCHAR264)
Il s'agit du nom d'hôte ou de l'adresse réseau d'un hôte sur lequel le serveur LDAP s'exécute. Il peut être suivi d'un numéro de port facultatif, entre parenthèses. Le numéro de port par défaut est 389.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie MQRC_AUTH_INFO_CONN_NAME_ERROR.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ_AUTH_INFO_CONN_NAME_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.

Type AuthInfo(MQLONG)
Il s'agit du type d'informations d'authentification contenues dans l'enregistrement.

La valeur peut être l'un des deux paramètres suivants:

MQAIT_CRL_LDAP
Vérification de la révocation de certificat à l'aide du serveur LDAP.

MQAIT_OCSP
Vérification de la révocation de certificat à l'aide d'OCSP.

Si la valeur n'est pas valide, l'appel échoue avec le code anomalie MQRC_AUTH_INFO_TYPE_ERROR.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQAIT_CRL_LDAP.

LDAPPassword (MQCHAR32)
Il s'agit du mot de passe requis pour accéder au serveur CRL LDAP. Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs.

Si le serveur LDAP ne requiert pas de mot de passe ou si vous omettez le nom d'utilisateur LDAP, *LDAPPassword* doit être null ou vide. Si vous omettez le nom d'utilisateur LDAP et que *LDAPPassword* n'est pas null ou vide, l'appel échoue avec le code anomalie MQRC_LDAP_PASSWORD_ERROR.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ_LDAP_PASSWORD_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.

LDAPUserNameLongueur (MQLONG)

Il s'agit de la longueur en octets du nom d'utilisateur LDAP indiqué par la zone *LDAPUserNamePtr* ou *LDAPUserNameOffset* . La valeur doit être comprise entre zéro et MQ_DISTINGUISHED_NAME_LENGTH. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie MQRC_LDAP_USER_NAME_LENGTH_ERR.

Si le serveur LDAP impliqué ne requiert pas de nom d'utilisateur, définissez cette zone sur zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

LDAPUserNameDécalage (MQLONG)
Il s'agit du décalage en octets du nom d'utilisateur LDAP à partir du début de la structure MQAIR.

Le décalage peut être positif ou négatif. La zone est ignorée si *LDAPUserNameLength* a la valeur zéro.

Vous pouvez utiliser *LDAPUserNamePtr* ou *LDAPUserNameOffset* pour spécifier le nom d'utilisateur LDAP, mais pas les deux ; voir la description de la zone *LDAPUserNamePtr* pour plus de détails.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

LDAPUserNamePtr (PMQCHAR)
Il s'agit du nom d'utilisateur LDAP.

Il se compose du nom distinctif de l'utilisateur qui tente d'accéder au serveur CRL LDAP. Si la valeur est inférieure à la longueur spécifiée par *LDAPUserNameLength*, terminez la valeur par un caractère null

ou remplissez la valeur avec des blancs pour la longueur *LDAPUserNameLength*. La zone est ignorée si *LDAPUserNameLength* a la valeur zéro.

Vous pouvez fournir le nom d'utilisateur LDAP de l'une des deux manières suivantes:

- A l'aide de la zone de pointeur *LDAPUserNamePtr*

  Dans ce cas, l'application peut déclarer une chaîne distincte de la structure MQAIR et définir *LDAPUserNamePtr* sur l'adresse de la chaîne.

  Envisagez d'utiliser *LDAPUserNamePtr* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).
- En utilisant la zone de décalage *LDAPUserNameOffset*

  Dans ce cas, l'application doit déclarer une structure composée contenant la structure MQSCO suivie du tableau d'enregistrements MQAIR suivi des chaînes de nom d'utilisateur LDAP et définir *LDAPUserNameOffset* sur le décalage de la chaîne de nom appropriée à partir du début de la structure MQAIR. Vérifiez que cette valeur est correcte et qu'elle peut être prise en compte dans un MQLONG (le langage de programmation le plus restrictif est COBOL, pour lequel la plage valide est comprise entre -999 999 999 et +999 999 999).

  Envisagez d'utiliser *LDAPUserNameOffset* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui peut ne pas être portable dans des environnements différents (par exemple, le langage de programmation COBOL).

Quelle que soit la technique choisie, utilisez uniquement *LDAPUserNamePtr* et *LDAPUserNameOffset*; l'appel échoue avec le code anomalie MQRC_LDAP_USER_NAME_ERROR si les deux sont différents de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

Remarque : Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

OCSPResponderURL (MQCHAR256)
Pour une structure MQAIR qui représente les détails de connexion d'un répondeur OCSP, cette zone contient l'URL à laquelle le répondeur peut être contacté.

La valeur de cette zone est une URL HTTP. Cette zone est prioritaire sur une URL dans une extension de certificat AIA (AuthorityInfoAccess).

La valeur est ignorée sauf si les deux instructions suivantes sont vraies:

- La structure MQAIR est à la version 2 ou ultérieure (la zone Version est définie sur MQAIR_VERSION_2 ou une version ultérieure).
- La zone AuthInfoType est définie sur MQAIT_OCSP.

Si la zone ne contient pas d'URL HTTP au format correct (et n'est pas ignorée), l'appel MQCONNX échoue avec le code anomalie MQRC_OCSP_URL_ERROR.

Cette zone est sensible à la casse. Il doit commencer par la chaîne http:// en minuscules. Le reste de l'URL peut être sensible à la casse, en fonction de l'implémentation du serveur OCSP.

Cette zone n'est pas soumise à la conversion de données.

StrucId (MQCHAR4)

La valeur doit être:

ID_STRUC_MQAIR_
Identificateur de l'enregistrement d'informations d'authentification.

Pour le langage de programmation C, la constante MQAIR_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQAIR_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQAIR_STRUC_ID.

Version (MQLONG)
Numéro de version de la structure MQAIR.

Il doit s'agir de l'une des valeurs suivantes :

MQAIR_VERSION_1
Enregistrement des informations d'authentification Version-1 .

MQAIR_VERSION_2
Enregistrement des informations d'authentification Version-2 .

La constante suivante indique le numéro de version de la version en cours:

MQAIR_CURRENT_VERSION
Version actuelle de l'enregistrement des informations d'authentification.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQAIR_VERSION_1.

Valeurs initiales et déclarations de langue pour MQAIR

Tableau 473. Valeurs initiales des zones dans MQAIR

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQAIR_	'AIR¬'
Version	MQAIR_VERSION_1	1
AuthInfoType	MQAIT_CRL_LDAP	1
AuthInfoConnName	Aucun	Chaîne nulle ou blancs
LDAPUserNamePtr	Aucun	Pointeur null ou octets null
LDAPUserNameDécalage	Aucun	0
LDAPUserNameLongueur	Aucun	0
LDAPPassword	Aucun	Chaîne nulle ou blancs
OCSPResponderURL	Aucun	Chaîne nulle ou blancs

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQAIR_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

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

Déclaration C

```
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 */

};
```

Déclaration COBOL

```
**   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).
```

Déclaration Visual Basic

```
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-Options de traitement de la mémoire tampon vers le message

Le tableau suivant récapitule les zones de la structure. Structure MQBMHO-options de la mémoire tampon vers le descripteur de message

Tableau 474. Zones dans MQBMHO

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options contrôlant l'action de MQBMHO	Options

Présentation de MQBMHO

Disponibilité: tous. Structure des options de gestion de la mémoire tampon vers les messages-présentation

Objectif: La structure MQBMHO permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont générés à partir des mémoires tampon. La structure est un paramètre d'entrée dans l'appel MQBUFMH.

Jeu de caractères et codage: les données de MQBMHO doivent être dans le jeu de caractères de l'application et le codage de l'application (MQENC_NATIVE).

Zones pour MQBMHO

Structure des options de la mémoire tampon vers le descripteur de message-zones

La structure MQBMHO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

Options (MQLONG)
Structure de la mémoire tampon vers le descripteur de message-Zone Options

La valeur peut être :

MQBMHO_DELETE_PROPERTIES
Les propriétés ajoutées au descripteur de message sont supprimées de la mémoire tampon. Si l'appel échoue, aucune propriété n'est supprimée.

Options par défaut: si vous n'avez pas besoin de l'option décrite, utilisez l'option suivante:

MQBMHO_AUCUN
Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQBMHO_DELETE_PROPERTIES.

StrucId (MQCHAR4)
Mémoire tampon de la structure de descripteur de message-Zone StrucId

Il s'agit de l'identificateur de structure. La valeur doit être:

ID_STRUC_MQBMHO_
Identificateur de la mémoire tampon pour la structure de descripteur de message.

Pour le langage de programmation C, la constante MQBMHO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQBMHO_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQBMHO_STRUC_ID.

Version (MQLONG)
Structure de la mémoire tampon vers le descripteur de message-Zone Version

Il s'agit du numéro de version de la structure. La valeur doit être:

MQBMHO_VERSION_1
Numéro de version de la mémoire tampon pour la structure de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

MQBMHO_VERSION
Version en cours de la structure de mémoire tampon à descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQBMHO_VERSION_1.

Valeurs initiales et déclarations de langue pour MQBMHO

Structure de la mémoire tampon vers le descripteur de message-Valeurs initiales

Tableau 475. Valeurs initiales des zones dans MQBMHO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQBMHO_	'BMHO'
Version	MQBMHO_VERSION_1	1

Tableau 475. Valeurs initiales des zones dans MQBMHO (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
Options	MQBMHO_AUCUN	0

Remarques :

1. Dans le langage de programmation C, la variable macroMQBMHO_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQBMHO MyBMHO = {MQBMHO_DEFAULT};
```

Déclaration C
Mémoire tampon pour la structure de descripteur de message-Déclaration de langage C

```
typedef struct tagMQBMHO MQBMHO;
struct tagMQBMHO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQBUFMH */
};
```

Déclaration COBOL
Mémoire tampon pour la structure de descripteur de message-Déclaration en langage COBOL

```
**   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.
```

Déclaration PL/I
Mémoire tampon de la structure de descripteur de message-Déclaration de langage PL/I

```
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 */
```

Déclaration High Level Assembler
Structure de la mémoire tampon vers le descripteur de message-Déclaration du langage assembleur

```
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-Options de début

Le tableau suivant récapitule les zones de la structure.

Tableau 476. Zones dans MQBO		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options qui contrôlent l'action de MQBEGIN	Options

Présentation de MQBO

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows; non disponible pour les clients WebSphere MQ MQI.

Objectif: La structure MQBO permet à l'application de spécifier des options relatives à la création d'une unité d'oeuvre. La structure est un paramètre d'entrée-sortie dans l'appel MQBEGIN.

Jeu de caractères et codage: Les données dans MQBO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Zones pour MQBO

La structure MQBO contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

Options (MQLONG)
Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQBO_NONE.

La valeur doit être:

MQBO_AUCUN
Aucune option spécifiée.

StrucId (MQCHAR4)
Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQBO_STRUC_ID.

La valeur doit être:

ID_STRUC_MQBO_
Identificateur de la structure des options de début.

Pour le langage de programmation C, la constante MQBO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQBO_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Version (MQLONG)
Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQBO_VERSION_1.

La valeur doit être:

MQBO_VERSION_1
Numéro de version de la structure des options de début.

La constante suivante indique le numéro de version de la version en cours:

MQBO_CURRENT_VERSION
Version actuelle de la structure des options de début.

Valeurs initiales et déclarations de langue pour MQBO

Tableau 477. Valeurs initiales des zones dans MQBO pour MQBO		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
StrucId	ID_STRUC_MQBO_	'BO¬¬'

Tableau 477. Valeurs initiales des zones dans MQBO pour MQBO (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
Version	MQBO_VERSION_1	1
Options	MQBO_AUCUN	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQBO_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQBO MyBO = {MQBO_DEFAULT};
```

Déclaration C

```
typedef struct tagMQBO MQBO;
struct tagMQBO {
  MQCHAR4  StrucId;  /* Structure identifier */
  MQLONG   Version;  /* Structure version number */
  MQLONG   Options;  /* Options that control the action of MQBEGIN */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration Visual Basic

```
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-Contexte de rappel

Le tableau suivant récapitule les zones de la structure. Structure décrivant la routine de rappel.

Tableau 478. Zones dans MQCBC		
Zone	**Description**	**Topic**
StrucID	Identificateur de structure	StrucID
Version	Numéro de version de structure	Version
CallType	Pourquoi la fonction a été appelée	CallType
Hobj	Descripteur d'objet	HOBJ
CallbackArea	Zone de la fonction de rappel à utiliser	CallbackArea
ConnectionArea	Zone de la fonction de rappel à utiliser	ConnectionArea
CompCode	Code de fin d'exécution	CompCode
Reason	Code raison	Motif
State	Indication de l'état du consommateur actuel	Etat
DataLength	LONGUEUR DE MESSAGE	DataLength
BufferLength	Longueur de la mémoire tampon de message en octets	BufferLength
Flags	Indicateurs généraux	Indicateurs
Remarque : La zone restante est ignorée si la version est inférieure à MQCBC_VERSION_2		
ReconnectDelay	Nombre de millisecondes avant la tentative de reconnexion	ReconnectDelay

Présentation de MQCBC

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OS, plus les clients WebSphere MQ MQI connectés à ces systèmes.

Objectif: La structure MQCBC est utilisée pour spécifier les informations de contexte qui sont transmises à une fonction de rappel.

La structure est un paramètre d'entrée-sortie dans l'appel à une routine de consommateur de message.

Version: la version actuelle de MQCBC est MQCBC_VERSION_2.

Jeu de caractères et codage: les données de MQCBC doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure sera dans le jeu de caractères et le codage du client.

Zones pour MQCBC

Liste alphabétique des zones de la structure MQCBC.

La structure MQCBC contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

BufferLength (MQLONG)
Cette zone indique la longueur, en octets, de la mémoire tampon de messages qui a été transmise à cette fonction.

La mémoire tampon peut être supérieure à la valeur MaxMsgLength définie pour le consommateur et à la valeur ReturnedLength dans le MQGMO.

La longueur réelle des messages est indiquée dans la zone DataLength .

L'application peut utiliser la totalité de la mémoire tampon à ses propres fins pendant la durée de la fonction de rappel.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'exceptions.

CallbackArea (MQPTR)
Cette zone est disponible pour la fonction de rappel à utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone «CallbackArea (MQPTR)», à la page 270 de la structure MQCBD, qui est un paramètre de l'appel MQCB utilisé pour définir la fonction de rappel.

Les modifications apportées à *CallbackArea* sont conservées dans les appels de la fonction de rappel pour un *HObj*. Cette zone n'est pas partagée avec les fonctions de rappel pour les autres descripteurs.

Il s'agit d'un champ d'entrée/sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

CallType (MQLONG)
Zone contenant des informations sur la raison pour laquelle cette fonction a été appelée ; les éléments suivants sont définis.

Types d'appel de distribution de message: ces types d'appel contiennent des informations sur un message. Les paramètres *DataLength* et *BufferLength* sont valides pour ces types d'appel.

MQCBCT_MSG_REMOVED
La fonction de consommateur de message a été appelée avec un message qui a été supprimé de façon destructive de l'identificateur d'objet.

Si la valeur de *CompCode* est MQCC_WARNING, la valeur de la zone *Reason* est MQRC_TRUNCATED_MSG_ACCEPTED ou l'un des codes indiquant un problème de conversion de données.

MQCBCT_MSG_NOT_REMOVED
La fonction de consommateur de message a été appelée avec un message qui n'a pas encore été supprimé de façon destructive de l'identificateur d'objet. Le message peut être supprimé de façon destructive de l'identificateur d'objet à l'aide de *MsgToken*.

Le message n'a peut-être pas été supprimé pour les raisons suivantes:

- Les options MQGMO ont demandé une opération de navigation, MQGMO_BROWSE_ *
- Le message est plus grand que la mémoire tampon disponible et les options MQGMO ne spécifient pas MQGMO_ACCEPT_TRUNCATED_MSG

Si la valeur de *CompCode* est MQCC_WARNING, la valeur de la zone *Reason* est MQRC_TRUNCATED_MSG_FAILED ou l'un des codes indiquant un problème de conversion de données.

Types d'appel de contrôle de rappel: ces types d'appel contiennent des informations sur le contrôle du rappel et ne contiennent pas de détails sur un message. Ces types d'appel sont demandés à l'aide des options de la structure MQCBD.

Les paramètres *DataLength* et *BufferLength* ne sont pas valides pour ces types d'appel.

APPEL MQCBCT_REGISTER_CALL
Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer une configuration initiale.

La fonction de rappel est appelée immédiatement après l'enregistrement du rappel, c'est-à-dire lors du retour d'un appel MQCB à l'aide d'une valeur pour la zone *Operation* de MQOP_REGISTER.

Ce type d'appel est utilisé à la fois pour les consommateurs de messages et les gestionnaires d'événements.

S'il est demandé, il s'agit du premier appel de la fonction de rappel.

La valeur de la zone *Reason* est MQRC_NONE.

Appel MQCBCT_START_CALL

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer une configuration lors de son démarrage, par exemple, en réinstanciant des ressources qui ont été nettoyées lors de son arrêt précédent.

La fonction de rappel est appelée lorsque la connexion est démarrée à l'aide de MQOP_START ou de MQOP_START_WAIT.

Si une fonction de rappel est enregistrée dans une autre fonction de rappel, ce type d'appel est appelé lorsque le rappel est renvoyé.

Ce type d'appel est utilisé uniquement pour les consommateurs de message.

La valeur de la zone *Reason* est MQRC_NONE.

MQCBCT_STOP_CALL

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer un nettoyage lorsqu'elle est arrêtée pendant un certain temps, par exemple, en nettoyant des ressources supplémentaires qui ont été acquises lors de la consommation de messages.

La fonction de rappel est appelée lorsqu'un appel MQCTL est émis à l'aide d'une valeur pour la zone *Operation* de MQOP_STOP.

Ce type d'appel est utilisé uniquement pour les consommateurs de message.

La valeur de la zone *Reason* est définie pour indiquer la raison de l'arrêt.

MQCBCT_DEREGISTER_CALL

Le but de ce type d'appel est de permettre à la fonction de rappel d'effectuer un nettoyage final à la fin du processus de consommation. La fonction de rappel est appelée lorsque:

- La fonction de rappel est désenregistrée à l'aide d'un appel MQCB avec MQOP_REGISTER_.
- La file d'attente est fermée, ce qui entraîne un désenregistrement implicite. Dans cette instance, la fonction de rappel est transmise à MQHO_UNUSABLE_HOBJ en tant que descripteur d'objet.
- L'appel MQDISC se termine-provoquant une fermeture implicite et, par conséquent, un désenregistrement. Dans ce cas, la connexion n'est pas immédiatement déconnectée et toute transaction en cours n'est pas encore validée.

Si l'une de ces actions est effectuée dans la fonction de rappel elle-même, l'action est appelée une fois que le rappel est renvoyé.

Ce type d'appel est utilisé à la fois pour les consommateurs de messages et les gestionnaires d'événements.

S'il est demandé, il s'agit du dernier appel de la fonction de rappel.

La valeur de la zone *Reason* est définie pour indiquer la raison de l'arrêt.

MQCBCT_EVENT_CALL

Fonction du gestionnaire d'événements

La fonction de gestionnaire d'événements a été appelée sans message lorsque le gestionnaire de files d'attente ou la connexion s'arrête ou se met au repos.

Cet appel peut être utilisé pour effectuer l'action appropriée pour toutes les fonctions de rappel.

Fonction de consommateur de message

La fonction de consommateur de message a été appelée sans message lorsqu'une erreur (*CompCode* = MQCC_FAILED) spécifique au descripteur d'objet a été détectée ; par exemple, *Reason* code = MQRC_GET_INHIBÉ.

La valeur de la zone *Reason* est définie pour indiquer la raison de l'appel.

MQCBCT_MC_EVENT_CALL

La fonction de gestionnaire d'événements a été appelée pour les événements de multidiffusion. Le gestionnaire d'événements est envoyé WebSphere MQ Evénements de multidiffusion au lieu d'événements'normaux' WebSphere MQ .

Pour plus d'informations sur MQCBCT_MC_EVENT_CALL, voir Génération de rapports sur les exceptions de multidiffusion .

CompCode (MQLONG)
Cette zone indique le code achèvement. Indique s'il y a eu des problèmes lors de la consommation du message.

La valeur est l'une des suivantes :

MQCC_OK
Achèvement réussi

MQCC_WARNING
Avertissement (achèvement partiel)

MQCC_FAILED
Echec de l'appel

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQCC_OK.

ConnectionArea (MQPTR)
Cette zone est disponible pour la fonction de rappel à utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone «ConnectionArea (MQPTR)», à la page 318 de la structure MQCTLO, qui est un paramètre de l'appel MQCTL utilisé pour contrôler la fonction de rappel.

Toutes les modifications apportées à cette zone par les fonctions de rappel sont conservées dans les appels de la fonction de rappel. Cette zone peut être utilisée pour transmettre des informations qui doivent être partagées par toutes les fonctions de rappel. Contrairement à *CallbackArea* , cette zone est commune à tous les rappels d'un descripteur de connexion.

Il s'agit d'une zone d'entrée et de sortie. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

DataLength (MQLONG)
Il s'agit de la longueur en octets des données d'application du message. Si la valeur est zéro, cela signifie que le message ne contient pas de données d'application.

La zone DataLength contient la longueur du message, mais pas nécessairement la longueur des données de message transmises au consommateur. Il se peut que le message ait été tronqué. Utilisez la zone ReturnedLength dans le MQGMO pour déterminer la quantité de données qui a été réellement transmise au consommateur.

Si le code anomalie indique que le message a été tronqué, vous pouvez utiliser la zone DataLength pour déterminer la taille réelle du message. Cela vous permet de déterminer la taille de la mémoire tampon requise pour accueillir les données de message, puis d'émettre un appel MQCB pour mettre à jour MaxMsgLength avec une valeur appropriée.

Si l'option MQGMO_CONVERT est spécifiée, le message converti peut être supérieur à la valeur renvoyée pour DataLength. Dans de tels cas, l'application doit probablement émettre un appel MQCB pour mettre à jour MaxMsgLength pour qu'il soit supérieur à la valeur renvoyée par le gestionnaire de files d'attente pour DataLength.

Pour éviter les problèmes de troncature de message, spécifiez la longueur MaxMsgsous la forme MQCBD_FULL_MSG_LENGTH. Ainsi, le gestionnaire de files d'attente alloue une mémoire tampon pour la longueur complète des messages après la conversion des données. Sachez toutefois que même si cette option est spécifiée, il est possible que l'espace de stockage disponible soit insuffisant pour

traiter correctement la demande. Les applications doivent toujours vérifier le code anomalie renvoyé. Par exemple, s'il n'est pas possible d'allouer suffisamment de mémoire pour convertir le message, les messages sont renvoyés à l'application non convertie.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

Indicateurs (MQLONG)
Indicateurs contenant des informations sur ce consommateur.

L'option suivante est définie:

MQCBCF_READA_BUFFER_EMPTY
Cet indicateur peut être renvoyé si un appel MQCLOSE précédent utilisant l'option MQCO_QUIESCE a échoué avec le code anomalie MQRC_READ_AHEAD_MSGS.

Ce code indique que le dernier message de lecture anticipée est renvoyé et que la mémoire tampon est à présent vide. Si l'application émet un autre appel MQCLOSE à l'aide de l'option MQCO_QUIESCE), elle aboutit.

Notez qu'il n'est pas garanti qu'une application reçoit un message avec cet indicateur défini, car il peut encore y avoir des messages dans la mémoire tampon de lecture anticipée qui ne correspondent pas aux critères de sélection en cours. Dans cette instance, la fonction de consommateur est appelée avec le code anomalie MQRC_HOBJ_QUIESCED.

Si la mémoire tampon de lecture anticipée est complètement vide, le consommateur est appelé avec l'indicateur MQCBCF_READA_BUFFER_EMPTY et le code anomalie MQRC_HOBJ_QUIESCED_NO_MSGS.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

Hobj (MQHOBJ)
Il s'agit du descripteur d'objet pour les appels au consommateur de message.

Pour un gestionnaire d'événements, cette valeur est MQHO_NONE

L'application peut utiliser ce descripteur et le jeton de message dans le bloc Options d'extraction de message pour extraire le message si un message n'a pas été supprimé de la file d'attente.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQHO_UNUSABLE_HOBJ

Motif (MQLONG)
Il s'agit du code raison qualifiant le *CompCode*.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQRC_NONE.

Etat (MQLONG)
Indication de l'état du consommateur en cours. Cette zone est la plus importante pour une application lorsqu'un code raison différent de zéro est transmis à la fonction de consommateur.

Vous pouvez utiliser cette zone pour simplifier la programmation d'application car vous n'avez pas besoin de coder le comportement de chaque code anomalie.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQCS_NONE

Etat	Action du gestionnaire de files d'attente	Valeur de la constante
MQCS_NONE Ce code anomalie représente un appel normal sans informations supplémentaires sur la raison	Non ; il s'agit du fonctionnement normal.	0
MQCS_SUSPENDED_TEMPORARY Ces codes anomalie représentent des conditions temporaires.	La routine de rappel est appelée pour signaler la condition, puis suspendue. Au bout d'un certain temps, le système peut tenter à nouveau l'opération, ce qui peut entraîner une nouvelle levée de la même condition.	1
MQCS_SUSPENDED_USER_ACTION Ces codes anomalie représentent les conditions dans lesquelles le rappel doit intervenir pour résoudre la condition.	Le consommateur est suspendu et la routine de rappel est appelée pour signaler la condition. La routine de rappel doit résoudre la condition si possible et soit RESUME, soit fermer la connexion.	2
MQCS_SUSPENDED Ces codes anomalie représentent des échecs qui empêchent d'autres rappels de message.	Le gestionnaire de files d'attente interrompt automatiquement la fonction de rappel. Si la fonction de rappel est reprise, il est probable qu'elle reçoive à nouveau le même code anomalie.	3
MQCS_STOPPED Ces codes anomalie représentent la fin de la consommation des messages.	Distribué au gestionnaire d'exceptions et aux rappels qui ont spécifié MQCBDO_STOP_CALL. Aucun autre message ne peut être consommé.	4

StrucId (MQCHAR4)
La valeur de cette zone est l'identificateur de structure.

La valeur doit être:

MQCBC_STRUC_ID
Identificateur de la structure de contexte de rappel.

Pour le langage de programmation C, la constante MQCBC_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQCBC_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCBC_STRUC_ID.

Version (MQLONG)
La valeur de cette zone est le numéro de version de la structure.

La valeur doit être:

MQCBC_VERSION_1
Structure de contexte de rappel Version-1 .

La constante suivante indique le numéro de version de la version en cours:

MQCBC_CURRENT_VERSION
Version actuelle de la structure de contexte de rappel.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCBC_VERSION_1.

La fonction de rappel est toujours transmise à la dernière version de la structure.

ReconnectDelay (MQLONG)
ReconnectDelay indique la durée pendant laquelle le gestionnaire de files d'attente attend avant de tenter de se reconnecter. La zone peut être modifiée par un gestionnaire d'événements pour modifier complètement le délai ou arrêter la reconnexion.

Utilisez la zone `ReconnectDelay` uniquement si la valeur de la zone `Reason` dans le contexte de rappel est `MQRC_RECONNECTING`.

Lors de l'entrée dans le gestionnaire d'événements, la valeur de `ReconnectDelay` correspond au nombre de millisecondes pendant lesquelles le gestionnaire de files d'attente va attendre avant d'effectuer une tentative de reconnexion. Tableau 479, à la page 268 répertorie les valeurs que vous pouvez définir pour modifier le comportement du gestionnaire de files d'attente en cas de retour du gestionnaire d'événements.

Tableau 479. Valeurs de `ReconnectDelay`

Nom	Valeur	Description
`MQRD_NO_RECONNECT`	`-1`	Ne faites plus de tentatives de reconnexion. Une erreur est renvoyée à l'application.
`MQRD_NO_DELAY`	`0`	Essayez de vous reconnecter immédiatement.
`Milliseconds`	`>0`	Attendez ce nombre de millisecondes avant de retenter la connexion.

Valeurs initiales et déclarations de langue pour MQCBC
Structure de contexte de rappel-valeurs initiales

Il n'existe aucune valeur initiale pour la structure **MQCBC** . La structure est transmise en tant que paramètre à une routine de rappel. Le gestionnaire de files d'attente initialise la structure ; les applications ne l'initialisent jamais.

Déclaration C
Structure de contexte de rappel-Déclaration de langage C

```
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 */
```

Déclaration COBOL

```
** 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 **
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
    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-Descripteur de rappel

Le tableau suivant récapitule les zones de la structure. Structure spécifiant la fonction de rappel.

Tableau 480. Zones dans MQCBD

Zone	Description	Topic
StrucID	Identificateur de structure	StrucID

Tableau 480. Zones dans MQCBD (suite)		
Zone	**Description**	**Topic**
Version	Numéro de version de structure	Version
CallbackType	Type de fonction de rappel	CallbackType
Options	Options contrôlant la consommation des messages	Options
Callback Area	Zone de la fonction de rappel à utiliser	CallbackArea
CallbackFunction	Indique si la fonction est appelée en tant qu'appel d'API	CallbackFunction
CallbackName	Indique si la fonction est appelée en tant que programme lié dynamiquement	CallbackName
MaxMsgLength	Longueur du message le plus long pouvant être lu	Longueur maximale des messages

Présentation de MQCBD

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OSet WebSphere MQ clients MQI connectés à ces systèmes.

Objectif: La structure MQCBD est utilisée pour spécifier une fonction de rappel et les options contrôlant son utilisation par le gestionnaire de files d'attente.

La structure est un paramètre d'entrée dans l'appel MQCB.

Version: la version actuelle de MQCBD est MQCBD_VERSION_1.

Jeu de caractères et codage: les données dans MQCBD doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Zones pour MQCBD

Liste alphabétique des zones de la structure MQCBD.

La structure MQCBD contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

CallbackArea (MQPTR)
Structure de descripteur de rappel-Zone CallbackArea

Il s'agit d'une zone disponible que la fonction de rappel peut utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à partir de la zone «CallbackArea (MQPTR)», à la page 263 de la structure MQCBC, qui est un paramètre de la déclaration de la fonction de rappel.

La valeur est utilisée uniquement sur un *Operation* ayant une valeur MQOP_REGISTER, sans rappel actuellement défini, elle ne remplace pas une définition précédente.

Il s'agit d'une zone d'entrée et de sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

CallbackFunction (MQPTR)
Structure de descripteur de rappel-Zone CallbackFunction

La fonction de rappel est appelée en tant qu'appel de fonction.

Utilisez cette zone pour spécifier un pointeur vers la fonction de rappel.

Vous *devez* spécifier *CallbackFunction* ou *CallbackName*. Si vous spécifiez les deux, le code anomalie MQRC_CALLBACK_ROUTINE_ERROR est renvoyé.

Si ni *CallbackName* ni *CallbackFunction* n'est défini, l'appel échoue avec le code anomalie MQRC_CALLBACK_ROUTINE_ERROR.

Cette option n'est pas prise en charge dans l'environnement suivant: Langages de programmation et compilateurs qui ne prennent pas en charge les références de pointeur de fonction. Dans de telles situations, l'appel échoue avec le code anomalie MQRC_CALLBACK_ROUTINE_ERROR.

Sous z/OS , la fonction doit être appelée avec les conventions de liaison du système d'exploitation. Par exemple, dans le langage de programmation C, spécifiez:

```
#pragma linkage(MQCB_FUNCTION,OS)
```

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

Remarque : Lors de l'utilisation de CICS avec WebSphere MQ V7.0.1, la consommation asynchrone est prise en charge si:

- Apar PK66866 est appliqué à CICS TS 3.2
- Apar PK89844 est appliqué à CICS TS 4.1

CallbackName (MQCHAR128)
Structure de descripteur de rappel-Zone CallbackName

La fonction de rappel est appelée en tant que programme lié dynamiquement.

Vous *devez* spécifier *CallbackFunction* ou *CallbackName*. Si vous spécifiez les deux, le code anomalie MQRC_CALLBACK_ROUTINE_ERROR est renvoyé.

Si *CallbackName* ou *CallbackFunction* n'est pas défini, l'appel échoue avec le code anomalie MQRC_CALLBACK_ROUTINE_ERROR.

Le module est chargé lorsque la première routine de rappel à utiliser est enregistrée et déchargé lorsque la dernière routine de rappel à utiliser désenregistre.

Sauf indication contraire dans le texte suivant, le nom est cadré à gauche dans la zone, sans espaces imbriqués ; le nom lui-même est complété par des blancs à la longueur de la zone. Dans les descriptions qui suivent, les crochets ([ ]) indiquent des informations facultatives:

IBM i
Le nom de rappel peut être l'un des formats suivants:

- Programme de bibliothèque "/"
- Bibliothèque "/" ServiceProgram "("FunctionName")"

Par exemple, `MyLibrary/MyProgram(MyFunction)`.

Le nom de la bibliothèque peut être *LIBL. Les noms de bibliothèque et de programme sont limités à un maximum de 10 caractères.

Systèmes UNIX
Le nom de rappel est le nom d'un module ou d'une bibliothèque chargeable dynamiquement, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire:

```
[path]library(function)
```

Si le chemin n'est pas spécifié, le chemin de recherche système est utilisé.

Le nom est limité à un maximum de 128 caractères.

Windows
Le nom de rappel est le nom d'une bibliothèque de liens dynamiques, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire et d'une unité:

```
[d:][path]library(function)
```

Si l'unité et le chemin ne sont pas spécifiés, le chemin de recherche système est utilisé.

Le nom est limité à un maximum de 128 caractères.

z/OS
Le nom de rappel est le nom d'un module de chargement qui est valide pour la spécification sur le paramètre EP de la macro LINK ou LOAD.

Le nom est limité à un maximum de 8 caractères.

z/OS CICS
Le nom de rappel est le nom d'un module de chargement valide pour la spécification sur le paramètre PROGRAM de la macro de la commande EXEC CICS LINK.

Le nom est limité à un maximum de 8 caractères.

Le programme peut être défini comme programme distant à l'aide de l'option REMOTESYTEM de la définition PROGRAM installée ou par le programme de routage dynamique.

La région CICS distante doit être connectée à WebSphere MQ si le programme doit utiliser des appels API WebSphere MQ . Notez toutefois que la zone «Hobj (MQHOBJ)», à la page 266 de la structure MQCBC n'est pas valide sur un système distant.

Si un incident se produit lors de la tentative de chargement de *CallbackName*, l'un des codes d'erreur suivants est renvoyé à l'application:

- MQRC_MODULE_NOT_FOUND
- MQRC_MODULE_INVALID
- MQRC_MODULE_ENTRY_NOT_FOUND

Un message est également consigné dans le journal des erreurs contenant le nom du module pour lequel le chargement a été tenté et le code anomalie du système d'exploitation.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est une chaîne nulle ou des blancs.

CallbackType (MQLONG)
Structure de descripteur de rappel-Zone CallbackType

Il s'agit du type de la fonction de rappel. Les valeurs suivantes sont possibles :

MQCBT_MESSAGE_CONSUMER
Définit ce rappel en tant que fonction de consommateur de message.

Une fonction de rappel de consommateur de message est appelée lorsqu'un message répondant aux critères de sélection spécifiés est disponible sur un descripteur d'objet et que la connexion est démarrée.

MQCBT_EVENT_HANDLER
Définit ce rappel comme la routine d'événement asynchrone ; il n'est pas piloté pour consommer des messages pour un descripteur.

Hobj n'est pas requis sur l'appel MQCB définissant le gestionnaire d'événements et est ignoré s'il est spécifié.

Le gestionnaire d'événements est appelé pour les conditions qui affectent l'ensemble de l'environnement du consommateur de messages. La fonction de consommateur est appelée sans message lorsqu'un événement, par exemple l'arrêt ou la mise au repos d'un gestionnaire de files d'attente ou d'une connexion, se produit. Elle n'est pas appelée pour les conditions spécifiques à un seul consommateur de message, par exemple, MQRC_GET_INHIBÉ.

Les événements sont distribués à l'application, que la connexion soit démarrée ou arrêtée, sauf dans les environnements suivants:

- Environnement CICS on z/OS
- applications sans unités d'exécution

Si l'appelant ne transmet pas l'une de ces valeurs, l'appel échoue avec le code *Reason* MQRC_CALLBACK_TYPE_ERROR

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCBT_MESSAGE_CONSUMER.

Longueur MaxMsg(MQLONG)
Il s'agit de la longueur en octets du message le plus long qui peut être lu à partir du descripteur et transmis à la routine de rappel. Structure de descripteur de rappel- MaxMsgZone Longueur

Si un message a une longueur plus longue, la routine de rappel reçoit *MaxMsgLength* octets du message et le code anomalie suivant:

- MQRC_TRUNCATED_MSG_FAILED ou
- MQRC_TRUNCATED_MSG_ACCEPTED si vous avez spécifié MQGMO_ACCEPT_TRUNCATED_MSG.

La longueur réelle des messages est indiquée dans la zone «DataLength (MQLONG)», à la page 265 de la structure MQCBC.

La valeur spéciale suivante est définie:

MQCBD_LONGUEUR_MSG_COMPLET
La longueur de la mémoire tampon est ajustée par le système pour renvoyer les messages sans troncature.

Si la mémoire disponible est insuffisante pour allouer une mémoire tampon pour recevoir le message, le système appelle la fonction de rappel avec un code anomalie MQRC_STORAGE_NOT_AVAILABLE.

Si, par exemple, vous demandez une conversion de données et que la mémoire disponible est insuffisante pour convertir les données de message, le message non converti est transmis à la fonction de rappel.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *MaxMsgLength* est MQCBD_FULL_MSG_LENGTH.

Options (MQLONG)
Structure de descripteur de rappel-Zone Options

Tout ou partie des éléments suivants peuvent être spécifiés. Si plusieurs options sont requises, les valeurs peuvent être:

- Ajouté ensemble (ne pas ajouter la même constante plus d'une fois), ou
- combinées par le biais de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations par bit).

MQCBDO_ECHEC_IF_QUIESCING
L'appel MQCB échoue si le gestionnaire de files d'attente est à l'état de mise au repos.

Sous z/OS, cette option force également l'échec de l'appel MQCB si la connexion (pour une application CICS ou IMS) est à l'état de mise au repos.

Spécifiez MQGMO_FAIL_IF_QUIESCING, dans les options MQGMO transmises à l'appel MQCB, pour envoyer une notification aux consommateurs de message lorsqu'ils sont mis au repos.

Options de contrôle: Les options suivantes contrôlent si la fonction de rappel est appelée, sans message, lorsque l'état du consommateur change:

APPEL MQCBDO_REGISTER_CALL
La fonction de rappel est appelée avec le type d'appel MQCBCT_REGISTER_CALL.

Appel MQCBDO_START_CALL
La fonction de rappel est appelée avec le type d'appel MQCBCT_START_CALL.

Appel MQCBDO_STOP_CALL
La fonction de rappel est appelée avec le type d'appel MQCBCT_STOP_CALL.

MQCBDO_DEREGISTER_CALL
La fonction de rappel est appelée avec le type d'appel MQCBCT_DEREGISTER_CALL.

MQCBDO_EVENT_CALL
La fonction de rappel est appelée avec le type d'appel MQCBCT_EVENT_CALL.

MQCBDO_MC_EVENT_CALL
La fonction de rappel est appelée avec le type d'appel MQCBCT__MC_EVENT_CALL.

Pour plus de détails sur ces types d'appel, voir «CallType (MQLONG)», à la page 263 .

Option par défaut: si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

MQCBDO_AUCUN
Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

MQCBDO_NONE est défini pour faciliter la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *Options* est MQCBDO_NONE.

StrucId (MQCHAR4)
Structure de descripteur de rappel-Zone StrucId

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID MQCBD_STRUC_ID
Identificateur de la structure de descripteur de rappel.

Pour le langage de programmation C, la constante MQCBD_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQCBD_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCBD_STRUC_ID.

Version (MQLONG)
Structure de descripteur de rappel-Zone Version

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQCBD_VERSION_1
Structure de descripteur de rappel Version-1 .

La constante suivante indique le numéro de version de la version en cours:

MQCBD_CURRENT_VERSION
Version actuelle de la structure de descripteur de rappel.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCBD_VERSION_1.

Valeurs initiales et déclarations de langue pour MQCBD

Structure du descripteur de rappel-Valeurs initiales

Tableau 481. Valeurs initiales des zones dans MQCBD

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID MQCBD_STRUC_ID	'CBD¬'
Version	MQCBD_VERSION_1	1

Tableau 481. Valeurs initiales des zones dans MQCBD (suite)

Nom de zone	Nom de la constante	Valeur de la constante
CallBackType	MQCBT_MESSAGE_CONSUMER	1
Options	MQCBDO_AUCUN	0
CallbackArea	Aucun	Pointeur null ou blancs null
CallbackFunction	Aucun	Pointeur null ou blancs null
CallbackName	Aucun	Chaîne nulle ou blancs
MaxMsgLength	MQCBD_LONGUEUR_MSG_COMPLET	-1

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur "chaîne nulle" ou "blancs" indique la chaîne nulle dans le langage de programmation C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQCBD_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCBD MyCBD = {MQCBD_DEFAULT};
```

Déclaration C
Structure de descripteur de rappel-Déclaration en langage C

```
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 */
 };
```

Déclaration COBOL

```
** 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.
```

Déclaration PL/I

```
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-Chaîne de longueur variable

Le tableau suivant récapitule les zones de la structure.

Zone	Description	Topic
VSPtr	Pointeur vers la chaîne de longueur variable	VSPtr
VSOffset	Décalage en octets de la chaîne de longueur variable à partir du début de la structure qui contient cette structure MQCHARV	Décalage VSOffset
VSLength	Longueur en octets de la chaîne de longueur variable adressée par la zone VSPtr ou VSOffset.	Longueur VSLength
VSBufSize	Taille en octets de la mémoire tampon adressée par la zone VSPtr ou VSOffset.	VSBufSize
VSCCSID	Identificateur de jeu de caractères de la chaîne de longueur variable adressée par la zone VSPtr ou VSOffset.	VSCCSID

Présentation de MQCHARV

Disponibilité: AIX, HP-UX, Solaris, Linux, IBM i, Windows, plus les clients WebSphere MQ MQI connectés à ces systèmes.

Objectif: utilisez la structure MQCHARV pour décrire une chaîne de longueur variable.

Jeu de caractères et codage: les données du MQCHARV doivent être dans le codage du gestionnaire de files d'attente local fourni par MQENC_NATIVE et dans le jeu de caractères de la zone VSCCSID de la structure. Si l'application s'exécute en tant que client MQ , la structure doit être dans le codage du client. Certains jeux de caractères ont une représentation qui dépend du codage. Si VSCCSID est l'un de ces jeux de caractères, le codage utilisé est le même que celui des autres zones du MQCHARV. Le jeu de caractères identifié par VSCCSID peut être un jeu de caractères codé sur deux octets (DBCS).

Utilisation: La structure MQCHARV traite les données qui peuvent être discontiguës à la structure qui les contient. Pour traiter ces données, les zones déclarées avec le type de données de pointeur peuvent être utilisées. Notez que COBOL ne prend pas en charge le type de données de pointeur dans tous les environnements. Pour cette raison, les données peuvent également être traitées à l'aide de zones qui contiennent le décalage des données à partir du début de la structure contenant le MQCHARV.

Programmation COBOL

Si vous souhaitez porter une application entre des environnements, vous devez vérifier si le type de données de pointeur est disponible dans tous les environnements prévus. Si ce n'est pas le cas, l'application doit traiter les données à l'aide des zones de décalage au lieu des zones de pointeur.

Dans les environnements où les pointeurs ne sont pas pris en charge, vous pouvez déclarer les zones de pointeur en tant que chaînes d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets

tout-null. Ne modifiez pas cette valeur initiale si vous utilisez les zones de décalage. Pour ce faire, sans modifier les fichiers de stockage fournis, vous pouvez utiliser les éléments suivants:

```
COPY CMQCHRVV REPLACING POINTER BY ==BINARY PIC S9(9)==.
```

où CMQCHRVV peut être échangé pour le copybook à utiliser.

Zones pour MQCHARV

La structure MQCHARV contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

VSBufSize (MQLONG)
Il s'agit de la taille en octets de la mémoire tampon adressée par la zone VSPtr ou VSOffset.

Lorsque la structure MQCHARV est utilisée comme champ de sortie dans un appel de fonction, cette zone doit être initialisée avec la longueur de la mémoire tampon fournie. Si la valeur de VSLength est supérieure à VSBufSize , seuls VSBufSize octets de données sont renvoyés à l'appelant dans la mémoire tampon.

Cette valeur doit être une valeur supérieure ou égale à zéro, ou la valeur spéciale suivante reconnue:

MQVS_USE_VSLENGTH
Lorsqu'elle est spécifiée, la longueur de la mémoire tampon est extraite de la zone VSLength de la structure MQCHARV. N'utilisez pas cette valeur lorsque vous utilisez la structure comme zone de sortie et qu'une mémoire tampon est fournie.

Il s'agit de la valeur initiale de cette zone.

VSCCSID (MQLONG)
Il s'agit de l'identificateur de jeu de caractères de la chaîne de longueur variable adressée par la zone VSPtr ou VSOffset.

La valeur initiale de cette zone est MQCCSI_APPL, qui est définie par MQ pour indiquer qu'elle doit être remplacée par l'identificateur de jeu de caractères true du processus en cours. Par conséquent, la valeur MQCCSI_APPL n'est jamais associée à une chaîne de longueur variable. La valeur initiale de cette zone peut être modifiée en définissant une valeur différente pour la constante MQCCSI_APPL de votre unité de compilation par les moyens appropriés pour le langage de programmation de votre application.

Longueur VSLength (MQLONG)
Longueur en octets de la chaîne de longueur variable adressée par la zone VSPtr ou VSOffset.

La valeur initiale de cette zone est 0. La valeur doit être supérieure ou égale à zéro ou la valeur spéciale suivante qui est reconnue:

MQVS_NULL_TERMINATED

Si MQVS_NULL_TERMINATED n'est pas spécifié, les octets VSLength sont inclus dans la chaîne. Si des caractères nuls sont présents, ils ne délimitent pas la chaîne.

Si MQVS_NULL_TERMINATED est spécifié, la chaîne est délimitée par la première valeur null rencontrée dans la chaîne. La valeur null elle-même n'est pas incluse dans cette chaîne.

Remarque : Le caractère null utilisé pour mettre fin à une chaîne si MQVS_NULL_TERMINATED est spécifié est une valeur null provenant du jeu de codes spécifié par VSCCSID.

Par exemple, dans UTF-16 (UCS-2 CCSID 1200 et 13488), il s'agit du codage Unicode à deux octets dans lequel une valeur null est représentée par un nombre de 16 bits de zéros. En UTF-16 , il est courant de rechercher des octets uniques définis sur zéro qui font partie de caractères (caractères ASCII 7 bits par exemple), mais les chaînes ne se terminent par une valeur nulle que lorsque deux octets'zéro'sont trouvés sur une limite d'octet pair. Il est possible d'obtenir deux octets'zéro'sur une limite impaire lorsqu'ils font partie de caractères valides. Par exemple, x'01'x'00 x'00'00'x'30'représente deux caractères Unicode valides et ne termine pas la chaîne.

Décalage VS (MQLONG)
Le décalage peut être positif ou négatif. Vous pouvez utiliser la zone VSPtr ou VSOffset pour spécifier la chaîne de longueur variable, mais pas les deux. Décalage en octets de la chaîne de longueur variable à partir du début de MQCHARV ou de la structure qui la contient.

Lorsque la structure MQCHARV est imbriquée dans une autre structure, cette valeur correspond au décalage en octets de la chaîne de longueur variable à partir du début de la structure qui contient cette structure MQCHARV. Lorsque la structure MQCHARV n'est pas imbriquée dans une autre structure, par exemple, si elle est spécifiée en tant que paramètre dans un appel de fonction, le décalage est relatif au début de la structure MQCHARV.

La valeur initiale de cette zone est 0.

VSPtr (MQPTR)
Il s'agit d'un pointeur vers la chaîne de longueur variable.

Vous pouvez utiliser la zone VSPtr ou VSOffset pour spécifier la chaîne de longueur variable, mais pas les deux.

La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

Valeurs initiales et déclarations de langue pour MQCHARV

Valeurs initiales des zones dans MQCHARV

Nom de zone	Nom de la constante	Valeur de la constante
VSPtr	Aucun	Pointeur NULL ou octets NULL.
VSOffset	Aucun	0
VSBufSize	MQVS_USE_VSLENGTH	0
VSLength	Aucun	0
VSCCSID	MQCCSI_APPL	-3

Remarque : Dans le langage de programmation C, la variable de macro MQCHARV_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCHARV MyVarStr = {MQCHARV_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL pour MQCHARV

```
** 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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Redéfinition de MQCCSI_APPL

Les exemples suivants montrent comment remplacer la valeur de MQCCSI_APPL dans différents langages de programmation. Vous pouvez modifier la valeur de MQCCSI_APPL en supprimant la nécessité de définir le VSCCSID pour chaque chaîne de longueur variable séparément.

Dans ces exemples, le CCSID est défini sur 1208 ; remplacez-le par la valeur requise. Il s'agit de la valeur par défaut, que vous pouvez remplacer en définissant le VSCCSID dans n'importe quelle instance spécifique de MQCHARV.

Utilisation de C

```
#define MQCCSI_APPL 1208
#include <cmqc.h>
```

Utilisation COBOL

```
COPY CMQXYZV REPLACING -3 BY 1208.
```

Utilisation de PL/I

```
%MQCCSI_APPL = '1208';
%include syslib(cmqp);
```

Utilisation de l'assembleur System/390

```
MQCCSI_APPL EQU 1208
         CMQA LIST=NO
```

MQCIH-en-tête de pont CICS

Le tableau suivant récapitule les zones de la structure.

Tableau 482. Zones dans MQCIH		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
StrucLength	Longueur de la structure MQCIH	StrucLength
Encoding	Réservé	Codage
CodedCharSetId	Réservé	CodedCharSetId
Format	Nom de format MQ des données qui suivent MQCIH	Format
Flags	Indicateurs	Indicateurs
ReturnCode	Code retour du pont	ReturnCode
CompCode	Code achèvement MQ ou CICS EIBRESP	CompCode
Reason	Code anomalie ou code retour MQ ou CICS EIBRESP2	Motif
UOWControl	Contrôle de l'unité de travail	Contrôle d'unité de travail
GetWaitInterval	Intervalle d'attente pour l'appel MQGET émis par la tâche du pont	GetWaitIntervalle
LinkType	Type de lien	LinkType
OutputDataLength	Longueur des données COMMAREA de sortie	OutputDataLongueur
FacilityKeepTime	Heure de publication de la fonction de pont	FacilityKeepHeure
ADSDescriptor	Descripteur ADS d'envoi/réception	ADSDescripteur
ConversationalTask	Indique si la tâche peut être conversationnelle	ConversationalTask
TaskEndStatus	Statut en fin de tâche	StatutTaskEnd
Facility	Jeton de fonction de pont	Installation
Function	Nom d'appel MQ ou fonction CICS EIBFN	Fonction
AbendCode	Code de fin anormale	AbendCode
Authenticator	Mot de passe ou passticket	Authentificateur
Reserved1	Réservé	Reserved1
ReplyToFormat	Nom de format MQ du message de réponse	FormatReplyTo
RemoteSysId	ID système CICS distant à utiliser	IDRemoteSys
RemoteTransId	CICS RTRANSID à utiliser	IDRemoteTrans
TransactionId	Transaction à associer	TransactionId
FacilityLike	Attributs de terminal émulé	FacilityLike
AttentionId	Clé AID	AttentionId
StartCode	Code de démarrage de transaction	StartCode
CancelCode	Code de fin anormale de transaction	CancelCode
NextTransactionId	Transaction suivante à associer	NextTransactionID

Tableau 482. Zones dans MQCIH (suite)		
Zone	**Description**	**Topic**
Reserved2	Réservé	Reserved2
Reserved3	Réservé	Reserved3
Remarque : Les zones restantes ne sont pas présentes si *Version* est inférieur à MQCIH_VERSION_2.		
CursorPosition	Position de curseur	CursorPosition
ErrorOffset	Décalage de l'erreur dans le message	ErrorOffset
InputItem	Réservé	InputItem
Reserved4	Réservé	Reserved4

Présentation de MQCIH

Disponibilité: AIX, HP-UX, z/OS, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ MQI connectés à ces systèmes.

Objectif: La structure MQCIH décrit les informations pouvant être présentes au début d'un message envoyé au pont CICS via WebSphere MQ for z/OS.

Nom de format: MQFMT_CICS.

Version: la version actuelle de MQCIH est MQCIH_VERSION_2. Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions qui suivent.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQCIH, avec la valeur initiale de la zone *Version* définie sur MQCIH_VERSION_2.

Jeu de caractères et codage: des conditions spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQCIH et les données de message d'application:

- Les applications qui se connectent au gestionnaire de files d'attente qui possède la file d'attente de pont CICS doivent fournir une structure MQCIH qui se trouve dans le jeu de caractères et le codage du gestionnaire de files d'attente. En effet, la conversion de données de la structure MQCIH n'est pas effectuée dans ce cas.
- Les applications qui se connectent à d'autres gestionnaires de files d'attente peuvent fournir une structure MQCIH qui se trouve dans l'un des jeux de caractères et codages pris en charge ; l'agent MCA récepteur connecté au gestionnaire de files d'attente propriétaire de la file d'attente de pont CICS convertit la structure MQCIH.
- Les données de message d'application qui suivent la structure MQCIH doivent être dans le même jeu de caractères et dans le même codage que la structure MQCIH. Vous ne pouvez pas utiliser les zones *CodedCharSetId* et *Encoding* de la structure MQCIH pour spécifier le jeu de caractères et le codage des données de message d'application.

  Vous devez fournir un exit de conversion de données pour convertir les données de message d'application si les données ne sont pas l'un des formats intégrés pris en charge par le gestionnaire de files d'attente.

Syntaxe: si l'application requiert des valeurs identiques aux valeurs initiales affichées dans Tableau 484, à la page 291et que le pont s'exécute avec AUTH=LOCAL ou AUTH=IDENTIFY, vous pouvez omettre la structure MQCIH du message. Dans tous les autres cas, la structure doit être présente.

Le pont accepte une structure MQCIH version-1 ou version-2 , mais pour les transactions 3270, vous devez utiliser une structure version-2 .

L'application doit s'assurer que les zones documentées en tant que zones de demande ont des valeurs appropriées dans le message envoyé au pont ; ces zones sont entrées dans le pont.

Les zones documentées en tant que zones de réponse sont définies par le pont CICS dans le message de réponse que le pont envoie à l'application. Les informations d'erreur sont renvoyées dans les zones *ReturnCode*, *Function*, *CompCode*, *Reason*et *AbendCode* , mais elles ne sont pas toutes définies dans tous les cas. Tableau 483, à la page 282 indique les zones qui sont définies pour différentes valeurs de *ReturnCode*.

Tableau 483. Contenu des zones d'informations d'erreur dans la structure MQCIH pour MQCIH

ReturnCode	***Function***	***CompCode***	***Reason***	***AbendCode***
MQCRC_OK	-	-	-	-
MQCRC_BRIDGE_ERREUR	-	-	MQFB_CICS_ *	-
MQCRC_MQ_API_ERREUR MQCRC_BRIDGE_TIMEOUT	Nom d'appel MQ	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

Zones pour MQCIH

La structure MQCIH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

AbendCode (MQCHAR4)
AbendCode est une zone de réponse. La longueur de cette zone est indiquée par MQ_ABEND_CODE_LENGTH. La valeur initiale de cette zone est de 4 caractères blancs.

La valeur renvoyée dans cette zone est significative uniquement si la zone *ReturnCode* a la valeur MQCRC_APPLICATION_ABEND ou MQCRC_BRIDGE_ABEND. Si tel est le cas, *AbendCode* contient la valeur ABCODE CICS .

Descripteur DSA (MQLONG)
Cette zone indique si les descripteurs ADS doivent être envoyés sur les demandes SEND et RECEIVE BMS.

Les valeurs suivantes sont définies :

MQCADSD_NONE
N'envoyez pas ou ne recevez pas de descripteurs ADS.

MQCADSD_SEND
Envoi de descripteurs ADS.

MQCADSD_RECV
Descripteurs ADS de réception.

MQCADSD_MSGFORMAT
Utilisez le format de message pour les descripteurs ADS.

Ceci permet d'envoyer ou de recevoir les descripteurs ADS en utilisant la forme longue du descripteur ADS. Le format long comporte des zones alignées sur des limites de 4 octets.

Définissez la zone *ADSDescriptor* comme suit:

- Si vous n'utilisez pas de descripteurs ADS, définissez la zone sur MQCADSD_NONE.
- Si vous utilisez des descripteurs ADS avec le *même* CCSID dans chaque environnement, définissez la zone sur la somme de MQCADSD_SEND et MQCADSD_RECV.
- Si vous utilisez des descripteurs ADS avec des CCSID *différents* dans chaque environnement, définissez la zone sur la somme de MQCADSD_SEND, MQCADSD_RECV et MQCADSD_MSGFORMAT.

Il s'agit d'une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est MQCADSD_NONE.

AttentionId (MQCHAR4)
La valeur de cette zone détermine la valeur initiale de la clé AID au démarrage de la transaction. Il s'agit d'une valeur de 1 octet, alignée à gauche.

AttentionId est une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ_ATTENTION_ID_LENGTH. La valeur initiale de cette zone est de quatre blancs.

Authentificateur (MQCHAR8)
La valeur de cette zone est le mot de passe ou le mot de passe.

Si l'authentification par identificateur d'utilisateur est active pour le pont CICS , *Authenticator* est utilisé avec l'identificateur d'utilisateur dans le contexte d'identité MQMD pour authentifier l'expéditeur du message.

Il s'agit d'une zone de demande. La longueur de cette zone est indiquée par MQ_AUTHENTICATOR_LENGTH. La valeur initiale de cette zone est 8 blancs.

CancelCode (MQCHAR4)
La valeur de cette zone est le code de fin anormale à utiliser pour mettre fin à la transaction (normalement une transaction conversationnelle qui demande plus de données). Sinon, cette zone est mise à blanc.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ_CANCEL_CODE_LENGTH. La valeur initiale de cette zone est de quatre blancs.

CodedCharSetId (MQLONG)
CodedCharSetId est une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

L'ID de jeu de caractères pour les structures prises en charge qui suivent une structure MQCIH est identique à l'ID de jeu de caractères de la structure MQCIH elle-même et provient de tout en-tête WebSphere MQ précédent.

CompCode (MQLONG)
Cette zone est une zone de réponse. Sa valeur initiale est MQCC_OK

La valeur renvoyée dans cette zone dépend de *ReturnCode*; voir Tableau 483, à la page 282.

ConversationalTask (MQLONG)
Cette zone indique si la tâche doit être autorisée à émettre des demandes d'informations supplémentaires ou à arrêter la tâche et à émettre un message de fin anormale.

La valeur doit être l'une des options suivantes:

MQCCT_OUI
La tâche est conversationnelle.

MQCCT_NO
La tâche n'est pas conversationnelle.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est MQCCT_NO.

CursorPosition (MQLONG)
La valeur de cette zone indique la position initiale du curseur lorsque la transaction est démarrée. Pour les transactions conversationnelles, la position du curseur est dans le vecteur RECEIVE.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0. Cette zone n'est pas présente si *Version* est inférieur à MQCIH_VERSION_2.

Codage (MQLONG)
Cette zone est une zone réservée ; sa valeur n'est pas significative. Sa valeur initiale est 0.

Le codage des structures prises en charge qui suivent une structure MQCIH est identique au codage de la structure MQCIH elle-même et provient de tout en-tête WebSphere MQ précédent.

ErrorOffset (MQLONG)
La zone ErrorOffset affiche la position des données non valides détectées par l'exit de pont. Cette zone indique le décalage entre le début du message et l'emplacement des données non valides.

ErrorOffset est une zone de réponse utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0. Cette zone n'est pas présente si *Version* est inférieur à MQCIH_VERSION_2.

Fonction (MQBYTE8)
Cette zone affiche le jeton de la fonction de pont de 8 octets.

Un jeton de fonction de pont permet à plusieurs transactions d'une pseudo-conversation d'utiliser la même fonction de pont (terminal 3270 virtuel). Dans le premier ou le seul message d'une pseudo-conversation, définissez la valeur MQCFAC_NONE. Cette valeur indique à CICS d'allouer une nouvelle fonction de pont pour ce message. Un jeton de fonction de pont est renvoyé dans les messages de réponse lorsqu'un *FacilityKeepTime* différent de zéro est spécifié dans le message d'entrée. Les messages d'entrée suivants dans une pseudo-conversation doivent ensuite utiliser le même jeton de fonction de pont.

La valeur spéciale suivante est définie:

MQCFAC_NONE
Aucun jeton de fonction spécifié.

Pour le langage de programmation C, la constante MQCFAC_NONE_ARRAY est également définie et a la même valeur que MQCFAC_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Cette zone est à la fois une zone de demande et une zone de réponse utilisées uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ_FACILITY_LENGTH. La valeur initiale de cette zone est MQCFAC_NONE.

Heure FacilityKeep(MQLONG)
FacilityKeep: durée, en secondes, pendant laquelle la fonction de pont est conservée après la fin de la transaction utilisateur.

Pour les transactions pseudo-conversationnelles, indiquez une valeur qui correspond à la durée prévue d'une pseudo-conversation ; indiquez zéro pour la dernière transaction d'une pseudo-conversation et zéro pour les autres types de transaction.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La valeur initiale de cette zone est 0.

FacilityLike (MQCHAR4)
FacilityLike est le nom d'un terminal installé qui doit être utilisé comme modèle pour la fonction de pont.

Une valeur vide signifie que *FacilityLike* est extraite de la définition de profil de transaction de pont ou qu'une valeur par défaut est utilisée.

Cette zone est une zone de demande utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ_FACILITY_LIKE_LENGTH. La valeur initiale de cette zone est de quatre blancs.

Indicateurs (MQLONG)
Cette zone est une zone de demande. La valeur initiale de cette zone est MQCIH_NONE.

La valeur doit être:

MQCIH_NONE
Aucun indicateur.

EXPIRATION_MOT_DE_PASSE_MQCIH
Le message de réponse contient:

- Les mêmes options de rapport d'expiration que le message de demande.
- Heure d'expiration restante du message de demande sans ajustement effectué pour le temps de traitement du pont.

Si vous omettez cette valeur, le délai d'expiration est défini sur *unlimited*.

MQCIH_REPLY_WITHOUT_NULLS
La longueur du message de réponse d'une demande de programme DPL CICS est ajustée pour exclure les valeurs nulles de fin (X'00') à la fin de la zone de communication renvoyée par le programme DPL. Si cette valeur n'est pas définie, les valeurs nulles peuvent être significatives et la zone de communication complète est renvoyée.

MQCIH_SYNC_ON_RETURN
La liaison CICS pour les demandes DPL utilise l'option SYNCONRETURN, ce qui entraîne CICS à prendre un point de synchronisation lorsque le programme se termine s'il est envoyé à une autre région CICS . Le pont ne spécifie pas la région CICS à laquelle la demande doit être envoyée ; elle est contrôlée par la définition de programme CICS ou les fonctions d'équilibrage de charge.

Format (MQCHAR8)
Cette zone affiche le nom de format WebSphere MQ des données qui suivent la structure MQCIH.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de codage de la zone *Format* dans MQMD.

Ce nom de format est également utilisé pour le message de réponse si la zone *ReplyToFormat* a la valeur MQFMT_NONE.

- Pour les demandes DPL, *Format* doit être le nom de format de la zone de communication.
- Pour les demandes 3270, *Format* doit être CSQCBDCIet le pont définit le format sur CSQCBDCO pour les messages de réponse.

Les exits de conversion de données pour ces formats doivent être installés sur le gestionnaire de files d'attente où ils doivent être exécutés.

Si le message de demande génère un message de réponse d'erreur, le nom de format du message de réponse d'erreur est MQFMT_STRING.

Cette zone est une zone de demande. La longueur de cette zone est indiquée par MQ_FORMAT_LENGTH. La valeur initiale de cette zone est MQFMT_NONE.

Fonction (MQCHAR4)
Cette zone est une zone de réponse. La longueur de cette zone est indiquée par MQ_FUNCTION_LENGTH. La valeur initiale de cette zone est MQCFUNC_NONE.

La valeur renvoyée dans cette zone dépend de *ReturnCode*; voir Tableau 483, à la page 282. Les valeurs suivantes sont possibles lorsque *Function* contient un nom d'appel WebSphere MQ :

MQCFUNC_MQCONN
Appel MQCONN.

MQCFUNC_MQGET
Appel MQGET.

MQCFUNC_MQINQ
Appel MQINQ.

MQCFUNC_MQOPEN
Appel MQOPEN.

MQCFUNC_MQPUT
Appel MQPUT.

MQCFUNC_MQPUT1
Appel MQPUT1 .

MQCFUNC_NONE
Pas d'appel.

Dans tous les cas, pour le langage de programmation C, les constantes MQCFUNC_ * _ARRAY sont également définies ; ces constantes ont les mêmes valeurs que les constantes MQCFUNC_ * correspondantes, mais sont des tableaux de caractères au lieu de chaînes.

Intervalle GetWait(MQLONG)
Cette zone est une zone de demande. Sa valeur initiale est MQCGWI_DEFAULT.

Cette zone s'applique uniquement lorsque *UOWControl* a la valeur MQCUOWC_FIRST. Elle permet à l'application émettrice de spécifier la durée approximative, en millisecondes, pendant laquelle les appels MQGET émis par le pont attendent les deuxième et suivants messages de demande pour l'unité de travail démarrée par ce message. Cette fonction remplace l'intervalle d'attente par défaut utilisé par le pont. Vous pouvez utiliser les valeurs spéciales suivantes:

MQCGWI_VALEUR PAR DEFAUT
Intervalle d'attente par défaut.

Cette valeur entraîne l'attente du pont CICS pendant le temps spécifié lors du démarrage du pont.

MQWI_ILLIMITÉ
Intervalle d'attente illimité.

InputItem (MQLONG)
Cette zone est une zone réservée. La valeur doit être 0.

Cette zone n'est pas présente si *Version* est inférieur à MQCIH_VERSION_2.

LinkType (MQLONG)
Cette zone est une zone de demande. Sa valeur initiale est MQCLT_PROGRAM.

Cette valeur indique le type d'objet que le pont tente de lier. Il doit s'agir de l'une des valeurs suivantes:

PROGRAMME MQCL
Programme DPL.

TRANSACTION MQCL
Transaction 3270.

ID NextTransaction(MQCHAR4)
Cette valeur correspond au nom de la transaction suivante renvoyée par la transaction utilisateur (généralement par EXEC CICS RETURN TRANSID). S'il n'y a pas de transaction suivante, cette zone est mise à blanc.

Cette zone est une zone de réponse utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ_TRANSACTION_ID_LENGTH. La valeur initiale de cette zone est de quatre blancs.

OutputDataLongueur (MQLONG)
Cette zone est une zone de demande utilisée uniquement pour les programmes DPL. Sa valeur initiale est MQCODL_AS_INPUT.

Cette valeur correspond à la longueur des données utilisateur à renvoyer au client dans un message de réponse. Cette longueur inclut le nom de programme à 8 octets. La longueur de la zone de communication transmise au programme lié est la longueur maximale de cette zone et la longueur des données utilisateur du message de demande, moins 8.

Remarque : La longueur des données utilisateur dans un message correspond à la longueur du message, à l'exclusion de la structure MQCIH.

Si la longueur des données utilisateur dans le message de demande est inférieure à *OutputDataLength*, l'option DATALENGTH de la commande LINK est utilisée, ce qui permet à LINK d'être livré efficacement à une autre région CICS .

Vous pouvez utiliser la valeur spéciale suivante:

MQCODL_AS_ENTREE
La longueur de sortie est identique à la longueur d'entrée.

Cette valeur peut être nécessaire même si aucune réponse n'est demandée, afin de s'assurer que la zone de communication transmise au programme lié est de taille suffisante.

Motif (MQLONG)
Cette zone est une zone de réponse. Sa valeur initiale est MQRC_NONE.

La valeur renvoyée dans cette zone dépend de *ReturnCode*; voir Tableau 483, à la page 282.

ID RemoteSys(MQCHAR4)
Cette zone affiche l'identificateur de système CICS du système CICS qui traite la demande.

Si cette zone est vide, la demande du système CICS est traitée sur le même système CICS que le moniteur de pont. Le SYSID utilisé est renvoyé dans le message de réponse.

Pour une pseudo-conversation 3270, tous les messages suivants de la conversation doivent spécifier le SYSID distant renvoyé dans la réponse initiale. S'il est spécifié, le SYSID doit:

- Soyez actif.
- Accédez à la file d'attente des demandes WebSphere MQ .
- Est accessible par les liens CICS ISC à partir du système CICS du moniteur de pont.

ID RemoteTrans(MQCHAR4)
Cette zone est facultative. La longueur de cette zone est indiquée par MQ_TRANSACTION_ID_LENGTH.

Si elle est spécifiée, la zone est utilisée comme valeur RTRANSID de CICS START.

Format ReplyTo(MQCHAR8)
La valeur de cette zone est le nom de format WebSphere MQ du message de réponse envoyé en réponse au message en cours.

Les règles de codage de cette zone sont les mêmes que celles de codage de la zone *Format* dans MQMD.

Cette zone est une zone de demande utilisée uniquement pour les programmes DPL. La longueur de cette zone est indiquée par MQ_FORMAT_LENGTH. La valeur initiale de cette zone est MQFMT_NONE.

Reserved1 (MQCHAR8)
Cette zone est une zone réservée. La valeur doit être à 8 blancs.

Reserved2 (MQCHAR8)
Cette zone est une zone réservée. La valeur doit être à 8 blancs.

Reserved3 (MQCHAR8)
Cette zone est une zone réservée. La valeur doit être à 8 blancs.

Reserved4 (MQLONG)
Cette zone est une zone réservée. La valeur doit être 0.

Cette zone n'est pas présente si *Version* est inférieur à MQCIH_VERSION_2.

ReturnCode (MQLONG)
La valeur de cette zone est le code retour du pont CICS décrivant le résultat du traitement effectué par le pont. Cette zone est une zone de réponse, avec une valeur initiale de MQCRC_OK.

Les zones *Function*, *CompCode*, *Reason*et *AbendCode* peuvent contenir des informations supplémentaires (voir Tableau 483, à la page 282). La valeur est l'une des suivantes :

MQCRC_APPLICATION_ABEND
(5, X'005') L'application s'est terminée de manière anormale.

MQCRC_BRIDGE_ABEND
(4, X'004') Le pont CICS s'est arrêté de manière anormale.

MQCRC_BRIDGE_ERREUR
(3, X'003') Le pont CICS a détecté une erreur.

MQCRC_DÉLAI_BRIDGE_ATTENTE
(8, X'008') Deuxième message ou message ultérieur dans l'unité d'oeuvre en cours qui n'a pas été reçu dans le délai imparti.

MQCRC_CICS_EXEC_ERROR
(1, X'001') L'instruction EXEC CICS a détecté une erreur.

MQCRC_MQ_API_ERREUR
(2, X'002') L'appel MQ a détecté une erreur.

MQCRC_OK
(0, X'000') Aucune erreur.

MQCRC_PROGRAM_NOT_AVAILABLE
(7, X'007') Programme non disponible.

MQCRC_SECURITY_ERROR
(6, X'006') Une erreur de sécurité s'est produite.

MQCRC_TRANSID_NON DISPONIBLE
(9, X'009') Transaction non disponible.

StartCode (MQCHAR4)
La valeur de cette zone est un indicateur spécifiant si le pont émule une transaction de terminal ou une transaction lancée avec START.

Il doit s'agir de l'une des valeurs suivantes :

MQCSC_DEBUT
Début.

Données MQCSC_STARTDATA
Données de démarrage.

PUT MQCSC_TERMINE
Entrée de terminal.

MQCSC_NONE
Aucune.

Dans tous les cas, pour le langage de programmation C, les constantes MQCSC_ * _ARRAY sont également définies ; ces constantes ont les mêmes valeurs que les constantes MQCSC_ * correspondantes, mais sont des tableaux de caractères au lieu de chaînes.

Dans la réponse du pont, cette zone est définie sur le code de début correspondant à l'ID transaction suivant contenu dans la zone *NextTransactionId* . Les codes de début suivants sont possibles dans la réponse:

- MQCSC_DEBUT
- Données MQCSC_STARTDATA
- PUT MQCSC_TERMINE

Pour CICS Transaction Server Version 1.2, cette zone est une zone de demande uniquement ; sa valeur dans la réponse n'est pas définie.

Pour CICS Transaction Server Version 1.3 et les éditions suivantes, cette zone est à la fois une zone de demande et une zone de réponse.

Cette zone est utilisée uniquement pour les transactions 3270. La longueur de cette zone est indiquée par MQ_START_CODE_LENGTH. La valeur initiale de cette zone est MQCSC_NONE.

StrucId (MQCHAR4)
Cette zone est une zone de demande, avec la valeur initiale MQCIH_STRUC_ID.

La valeur doit être:

ID_STRUC_MQCIH
Identificateur de la structure d'en-tête d'informations CICS .

Pour le langage de programmation C, la constante MQCIH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQCIH_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

StrucLength (MQLONG)
Cette zone est une zone de demande, avec la valeur initiale MQCIH_LENGTH_2.

Il doit s'agir de l'une des valeurs suivantes :

MQCIH_LENGTH_1
Longueur de la structure d'en-tête d'informations version-1 CICS .

MQCIH_LENGTH_2
Longueur de la structure d'en-tête d'informations version-2 CICS .

La constante suivante indique la longueur de la version en cours:

MQCIH_LONGUEUR_EN_COURS
Longueur de la version actuelle de la structure d'en-tête d'informations CICS .

Statut TaskEnd(MQLONG)
Cette zone est une zone de réponse indiquant le statut de la transaction utilisateur à la fin de la tâche. La zone est utilisée uniquement pour les transactions 3270 et sa valeur initiale est MQCTES_NOSYNC.

L'une des valeurs suivantes est renvoyée:

MQCTES_NOSYNC
Non synchronisé.

La transaction utilisateur n'est pas encore terminée et n'est pas synchronisée. La zone *MsgType* dans MQMD est MQMT_REQUEST dans ce cas.

MQCTES_COMMIT
Validation de l'unité de travail.

La transaction utilisateur n'est pas encore terminée, mais elle a synchronisé la première unité de travail. La zone *MsgType* dans MQMD est MQMT_DATAGRAM dans ce cas.

MQCTES_BACKOUT
Unité de travail d'arrière-système.

La transaction utilisateur n'est pas encore terminée. L'unité de travail en cours est annulée. La zone *MsgType* dans MQMD est MQMT_DATAGRAM dans ce cas.

MQCTES_ENDTASK
Fin de la tâche.

La transaction utilisateur s'est arrêtée (ou s'est arrêtée de manière anormale). La zone *MsgType* dans MQMD est MQMT_REPLY dans ce cas.

TransactionId (MQCHAR4)
Cette zone est une zone de demande. Sa longueur est donnée par MQ_TRANSACTION_ID_LENGTH. La valeur initiale de cette zone est de quatre blancs.

Si *LinkType* a la valeur MQCLT_TRANSACTION, *TransactionId* est l'identificateur de transaction de la transaction utilisateur à exécuter ; indiquez une valeur non vide dans ce cas.

Si *LinkType* a la valeur MQCLT_PROGRAM, *TransactionId* est le code de transaction sous lequel tous les programmes de l'unité de travail doivent être exécutés. Si vous indiquez une valeur vide, le code de transaction par défaut du pont CICS DPL (CKBP) est utilisé. Si la valeur n'est pas vide, vous devez l'avoir définie dans CICS en tant que transaction locale avec un programme initial CSQCBP00. Cette zone s'applique uniquement lorsque *UOWControl* a la valeur MQCUOWC_FIRST ou MQCUOWC_ONLY.

Contrôle d'unité de travail (MQLONG)
Cette zone est une zone de demande qui contrôle le traitement de l'unité de travail effectué par le pont CICS . La valeur initiale de cette zone est MQCUOWC_ONLY.

Vous pouvez demander au pont d'exécuter une seule transaction ou un ou plusieurs programmes au sein d'une unité de travail. La zone indique si le pont CICS démarre une unité de travail, exécute la fonction demandée dans l'unité de travail en cours ou met fin à l'unité de travail en la validant ou en la sauvegardant. Diverses combinaisons sont prises en charge, pour optimiser les flux de transmission de données.

Il doit s'agir de l'une des valeurs suivantes :

MQCUOWC_ONLY
Démarrez l'unité de travail, exécutez la fonction, puis validez l'unité de travail.

MQCUOWC_CONTINUE
Données supplémentaires pour l'unité d'oeuvre en cours (3270 uniquement).

MQCUOWC_FIRST
Démarrer l'unité de travail et exécuter la fonction.

MQCUOWC_MILIEU
Exécuter la fonction dans l'unité de travail en cours

MQCUOWC_LAST
Exécutez la fonction, puis validez l'unité de travail.

MQCUOWC_COMMIT
Validez l'unité d'oeuvre (DPL uniquement).

MQCUOWC_BACKOUT
Revenez à l'unité d'oeuvre (DPL uniquement).

Version (MQLONG)
Cette zone est une zone de demande. Sa valeur initiale est MQCIH_VERSION_2.

Il doit s'agir de l'une des valeurs suivantes :

MQCIH_VERSION_1
Structure d'en-tête d'informations Version-1 CICS .

MQCIH_VERSION_2
Version-2 Structure d'en-tête d'informations CICS .

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

VERSION_MQCIH_CURRENT_VERSION
Version actuelle de la structure d'en-tête d'informations CICS .

Valeurs initiales et déclarations de langage pour MQCIH

Tableau 484. Valeurs initiales des zones dans MQCIH pour MQCIH

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQCIH	'CIH¬'
Version	MQCIH_VERSION_2	2
StrucLength	MQCIH_LENGTH_2	180
Encoding	Aucun	0
CodedCharSetId	Aucun	0
Format	MQFMT_AUCUN	Espaces vides
Flags	MQCIH_NONE	0
ReturnCode	MQCRC_OK	0
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
UOWControl	MQCUOWC_ONLY	273
GetWaitInterval	MQCGWI_VALEUR PAR DEFAUT	-2
LinkType	PROGRAMME MQCL	1
OutputDataLength	MQCODL_AS_ENTREE	-1
FacilityKeepTime	Aucun	0
ADSDescriptor	MQCADSD_NONE	0
ConversationalTask	MQCCT_NO	0
TaskEndStatus	MQCTES_NOSYNC	0
Facility	MQCFAC_NONE	Valeurs NULL
Function	MQCFUNC_NONE	Espaces vides
AbendCode	Aucun	Espaces vides
Authenticator	Aucun	Espaces vides
Reserved1	Aucun	Espaces vides
ReplyToFormat	MQFMT_AUCUN	Espaces vides
RemoteSysId	Aucun	Espaces vides
RemoteTransId	Aucun	Espaces vides
TransactionId	Aucun	Espaces vides
FacilityLike	Aucun	Espaces vides
AttentionId	Aucun	Espaces vides
StartCode	MQCSC_NONE	Espaces vides
CancelCode	Aucun	Espaces vides
NextTransactionId	Aucun	Espaces vides
Reserved2	Aucun	Espaces vides

Tableau 484. Valeurs initiales des zones dans MQCIH pour MQCIH (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
Reserved3	Aucun	Espaces vides
CursorPosition	Aucun	0
ErrorOffset	Aucun	0
InputItem	Aucun	0
Reserved4	Aucun	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQCIH_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCIH MyCIH = {MQCIH_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Options de création de descripteur de message

Le tableau suivant récapitule les zones de la structure.

Tableau 485. Zones dans MQCMHO		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options	Options

Présentation de MQCMHO

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OS et clients WebSphere MQ .

Objectif: la structure **MQCMHO** permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont créés. La structure est un paramètre d'entrée dans l'appel **MQCRTMH** .

Jeu de caractères et codage: les données dans **MQCMHO** doivent être dans le jeu de caractères de l'application et le codage de l'application (**MQENC_NATIVE**).

Zones pour MQCMHO

La structure MQCMHO contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

Options (MQLONG)
Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQCMHO_DEFAULT_VALIDATION.

L'une des options suivantes peut être spécifiée:

MQCMHO_VALIDATION

Lorsque **MQSETMP** est appelé pour définir une propriété dans ce descripteur de message, le nom de la propriété est validé pour s'assurer qu'il:

- ne contient pas de caractères non valides.
- ne démarre pas JMS ou usr.JMS , à l'exception de ce qui suit:
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType
  - JMSXGroupID
  - JMSXGroupSeq

  Ces noms sont réservés aux propriétés JMS.
- n'est pas l'un des mots clés suivants, dans un mélange de majuscules ou de minuscules:
  - et
  - ENTRE
  - ESCAPE
  - FALSE
  - IN
  - IS
  - SIMILAIRE A
  - NON
  - NULL
  - ou

– TRUE

- ne commence pas le corps. ou root. (sauf pour Root.MQMD.)

Si la propriété est MQ-defined (mq. *) et que le nom est reconnu, les zones de descripteur de propriété sont définies sur les valeurs correctes pour la propriété. Si la propriété n'est pas reconnue, la zone *Support* du descripteur de propriété est définie sur **MQPD_OPTIONAL**.

MQCMHO_VALEUR_PAR_DÉFAUT

Cette valeur indique que le niveau de validation par défaut des noms de propriété est atteint.

Le niveau de validation par défaut est équivalent au niveau spécifié par **MQCMHO_VALIDATE**.

Cette valeur est la valeur par défaut.

MQCMHO_NO_VALIDATION

Aucune validation n'est effectuée sur le nom de la propriété. Voir la description de **MQCMHO_VALIDATE**.

Option par défaut: Si aucune des options décrites ci-dessus n'est requise, l'option suivante peut être utilisée:

MQCMHO_AUCUN

Toutes les options prennent leurs valeurs par défaut. Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. **MQCMHO_NONE** aide la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

StrucId (MQCHAR4)
Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQCMHO_STRUC_ID.

Il s'agit de l'identificateur de structure ; la valeur doit être:

MQCMHO_STRUC_ID
Identificateur de la structure des options de création de descripteur de message.

Pour le langage de programmation C, la constante **MQCMHO_STRUC_ID_ARRAY** est également définie ; elle a la même valeur que **MQCMHO_STRUC_ID**, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Version (MQLONG)
Cette zone est toujours une zone d'entrée. Sa valeur initiale est MQCMHO_VERSION_1.

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQCMHO_VERSION_1
Version-1 crée une structure d'options de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

MQCMHO_CURRENT_VERSION
Version actuelle de la structure des options de création de descripteur de message.

Valeurs initiales et déclarations de langage pour MQCMHO

Tableau 486. Valeurs initiales des zones dans MQCMHO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	MQCMHO_STRUC_ID	'CMHO'
Version	MQCMHO_VERSION_1	1

Tableau 486. Valeurs initiales des zones dans MQCMHO (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
Options	MQCMHO_VALEUR_PAR_DÉFAUT	0

Remarques :

1. Dans le langage de programmation C, la variable macroMQCMHO_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCMHO MyCMHO = {MQCMHO_DEFAULT};
```

Déclaration C

```
struct tagMQCMHO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQCRTMH */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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-Options de connexion

Le tableau suivant récapitule les zones de la structure.

Tableau 487. Zones dans MQCNO		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version

Tableau 487. Zones dans MQCNO (suite)		
Zone	**Description**	**Topic**
Options	Options qui contrôlent l'action de MQCONNX	Options
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQCNO_VERSION_2.		
ClientConnOffset	Décalage de la structure MQCD pour la connexion client	DécalageClientConn
ClientConnPtr	Adresse la structure MQCD pour la connexion client	ClientConnPtr
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQCNO_VERSION_3.		
ConnTag	Balise de connexion du gestionnaire de files d'attente	ConnTag
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQCNO_VERSION_4.		
SSLConfigPtr	Adresse la structure MQSCO pour la connexion client	SSLConfigPtr
SSLConfigOffset	Décalage de la structure MQSCO pour la connexion client	SSLConfigOffset
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQCNO_VERSION_5.		
ConnectionId	ID de connexion unique	ConnectionId
SecurityParmsOffset	Paramètres de sécurité	DécalageSecurityParms
SecurityParmsPtr	Paramètres de sécurité	SecurityParmsPtr

Tâches associées
Utilisation de MQCONNX

Présentation de MQCNO

Disponibilité: Toutes les versions sauf MQCNO_VERSION_4: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus WebSphere MQ les clients MQI connectés à ces systèmes.

Objectif: La structure MQCNO permet à l'application de spécifier des options relatives à la connexion au gestionnaire de files d'attente local. La structure est un paramètre d'entrée-sortie sur l'appel MQCONNX. Pour plus d'informations sur l'utilisation des descripteurs partagés et de l'appel MQCONNX, voir Connexions partagées (indépendantes de l'unité d'exécution) avec MQCONNX.

Version: les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQCNO, mais avec la valeur initiale de la zone *Version* définie sur MQCNO_VERSION_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone *Version* sur le numéro de version de la version requise.

Jeu de caractères et codage: les données de MQCNO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client WebSphere MQ MQI, la structure doit se trouver dans le jeu de caractères et le codage du client.

Zones pour MQCNO

La structure MQCNO contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

ClientConnDécalage (MQLONG)
ClientConnLe décalage est le décalage en octets d'une structure de définition de canal MQCD à partir du début de la structure MQCNO. Le décalage peut être positif ou négatif. Cette zone est une zone d'entrée avec une valeur initiale de 0.

Utilisez *ClientConnOffset* uniquement lorsque l'application émettant l'appel MQCONNX s'exécute en tant que client WebSphere MQ MQI. Pour plus d'informations sur l'utilisation de cette zone, voir la description de la zone *ClientConnPtr* .

Cette zone est ignorée si *Version* est inférieur à MQCNO_VERSION_2.

ClientConnPtr (MQPTR)
ClientConnPtr est une zone d'entrée. Sa valeur initiale est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

Utilisez *ClientConnOffset* et *ClientConnPtr* uniquement lorsque l'application émettant l'appel MQCONNX s'exécute en tant que client WebSphere MQ MQI. En spécifiant l'une ou l'autre de ces zones, l'application peut contrôler la définition du canal de connexion client en fournissant une structure de définition de canal MQCD qui contient les valeurs requises.

Si l'application s'exécute en tant que client WebSphere MQ MQI, mais qu'elle ne fournit pas de structure MQCD, la variable d'environnement MQSERVER est utilisée pour sélectionner la définition de canal. Si MQSERVER n'est pas défini, la table de canaux client est utilisée.

Si l'application n'est pas exécutée en tant que client WebSphere MQ MQI, *ClientConnOffset* et *ClientConnPtr* sont ignorés.

Si l'application fournit une structure MQCD, définissez les zones répertoriées sur les valeurs requises ; les autres zones de MQCD sont ignorées. Vous pouvez compléter les chaînes de caractères avec des blancs à la longueur de la zone ou les terminer par un caractère NULL. Pour plus d'informations sur les zones de la structure MQCD, voir «Zones», à la page 1047 .

Zone dans MQCD	**Valeur**
ChannelName	Nom de canal.
Version	Numéro de version de la structure. Ne doit pas être inférieur à MQCD_VERSION_7.
TransportType	Tout type de transport pris en charge.
ModeName	Nom de mode LU 6.2 .
TpName	Nom du programme de transaction LU 6.2 .
SecurityExit	Nom de l'exit de sécurité de canal.
SendExit	Nom de l'exit d'émission de canal.
ReceiveExit	Nom de l'exit de réception de canal.
MaxMsgLength	Longueur maximale en octets des messages pouvant être envoyés via le canal de connexion client.
SecurityUserData	Données utilisateur pour l'exit de sécurité.
SendUserData	Données utilisateur pour l'exit d'émission.
ReceiveUserData	Données utilisateur de l'exit de réception.
UserIdentifier	ID utilisateur à utiliser pour établir une session LU 6.2 .
Password	Mot de passe à utiliser pour établir une session LU 6.2 .
ConnectionName	nom de la connexion.

Zone dans MQCD	Valeur
HeartbeatInterval	Durée en secondes entre les flux de pulsations.
StrucLength	Longueur de la structure MQCD.
ExitNameLength	Longueur des noms d'exit traités par *SendExitPtr* et *ReceiveExitPtr*. Doit être supérieur à zéro si *SendExitPtr* ou *ReceiveExitPtr* est défini sur une valeur qui n'est pas le pointeur null.
ExitDataLength	Longueur des données de sortie traitées par *SendUserDataPtr* et *ReceiveUserDataPtr*. Doit être supérieur à zéro si *SendUserDataPtr* ou *ReceiveUserDataPtr* est défini sur une valeur qui n'est pas le pointeur null.
SendExitsDefined	Nombre d'exits d'émission traités par *SendExitPtr*. Si la valeur est zéro, *SendExit* et *SendUserData* fournissent le nom et les données de l'exit. Si la valeur est supérieure à zéro, *SendExitPtr* et *SendUserDataPtr* fournissent les noms et les données des exits, et *SendExit* et *SendUserData* doivent être vides.
ReceiveExitsDefined	Nombre d'exits de réception traités par *ReceiveExitPtr*. Si la valeur est zéro, *ReceiveExit* et *ReceiveUserData* fournissent le nom et les données de l'exit. Si la valeur est supérieure à zéro, *ReceiveExitPtr* et *ReceiveUserDataPtr* fournissent les noms et les données des exits, et *ReceiveExit* et *ReceiveUserData* doivent être vides.
SendExitPtr	Adresse du nom du premier exit d'émission.
SendUserDataPtr	Adresse des données pour le premier exit d'émission.
ReceiveExitPtr	Adresse du nom du premier exit de réception.
ReceiveUserDataPtr	Adresse des données pour le premier exit de réception.
LongRemoteUserIdLength	Longueur de l'ID utilisateur éloigné long.
LongRemoteUserIdPtr	Adresse de l'ID utilisateur distant long.
RemoteSecurityId	Identificateur de sécurité distant.
SSLCipherSpec	SSL CipherSpec.
SSLPeerNamePtr	Adresse du nom d'homologue SSL.
SSLPeerNameLength	Longueur du nom d'homologue SSL.
KeepAliveInterval	Valeur transmise à la pile de communications pour la temporisation du signal de présence pour le canal
LocalAddress	Adresse de communication locale, y compris l'adresse IP de l'adaptateur de réseau local à utiliser, et plage de ports à utiliser pour les connexions sortantes.

Fournissez la structure de définition de canal de l'une des deux manières suivantes:

- En utilisant la zone de décalage *ClientConnOffset*

  Dans ce cas, l'application doit déclarer une structure composée contenant un MQCNO suivi de la structure de définition de canal MQCD et définir *ClientConnOffset* sur le décalage de la structure de définition de canal à partir du début de MQCNO. Vérifiez que ce décalage est correct. *ClientConnPtr* doit être défini sur le pointeur null ou sur les octets null.

  Utilisez *ClientConnOffset* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui

n'est pas portable dans des environnements différents (par exemple, le langage de programmation COBOL).

Pour le langage de programmation Visual Basic, une structure composée appeléeMQCNOCD est fourni dans le fichier d'en-tête CMQXB.BAS; cette structure contient une structure MQCNO suivie d'une structure MQCD. Initialisez MQCNOCD en appelant la sous-routine MQCNOCD_DEFAULTS. MQCNOCD est utilisé avec leVariante MQCONNXAny de l'appel MQCONNX ; voir la description de l'appel MQCONNX pour plus de détails.

- A l'aide de la zone de pointeur *ClientConnPtr*

  Dans ce cas, l'application peut déclarer la structure de définition de canal séparément de la structure MQCNO et définir *ClientConnPtr* sur l'adresse de la structure de définition de canal. Définissez *ClientConnOffset* sur zéro.

  Utilisez *ClientConnPtr* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

  Dans le langage de programmation C, vous pouvez utiliser la variable de macro MQCD_CLIENT_CONN_DEFAULT pour fournir des valeurs initiales pour la structure qui conviennent mieux à l'utilisation sur l'appel MQCONNX que les valeurs initiales fournies par MQCD_DEFAULT.

Quelle que soit la technique choisie, vous ne pouvez utiliser qu'une seule des options *ClientConnOffset* et *ClientConnPtr*; l'appel échoue avec le code anomalie MQRC_CLIENT_CONN_ERROR si les deux sont différents de zéro.

Une fois l'appel MQCONNX terminé, la structure MQCD n'est plus référencée.

Cette zone est ignorée si *Version* est inférieur à MQCNO_VERSION_2.

Remarque : Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

ConnectionId (MQBYTE24)
ConnectionId est un identificateur unique de 24 octets qui permet à WebSphere MQ d'identifier de manière fiable une application. Une application peut utiliser cet identificateur pour la corrélation dans les appels PUT et GET. Ce paramètre de sortie a une valeur initiale de 24 octets nuls dans tous les langages de programmation.

Le gestionnaire de files d'attente affecte un ID unique à toutes les connexions, mais celles-ci sont établies. Si un MQCONNX établit la connexion avec un MQCNO de version 5, l'application peut déterminer l' ConnectionId à partir du MQCNO renvoyé. L'identificateur affecté est unique parmi tous les autres identificateurs générés par WebSphere MQ , tels que CorrelId, MsgIDet GroupId.

Utilisez ConnectionId pour identifier les unités d'oeuvre à exécution longue à l'aide de la commande PCF Inquire Connection ou de la commande MQSC DISPLAY CONN. L' ConnectionId utilisé par les commandes MQSC (CONN) est dérivé de l' ConnectionId renvoyé ici. Les commandes d'interrogation et d'arrêt de connexion PCF peuvent utiliser le ConnectionId renvoyé ici sans modification.

Vous pouvez utiliser l' ConnectionId pour forcer la fin d'une unité d'oeuvre de longue durée, en spécifiant l' ConnectionId à l'aide de la commande PCF Arrêter la connexion ou de la commande MQSC STOP CONN. Pour plus d'informations sur l'utilisation de ces commandes, voir Arrêter la connexion et STOP CONN .

Cette zone n'est pas renvoyée si la version est inférieure à MQCNO_VERSION_5.

La longueur de cette zone est indiquée par MQ_CONNECTION_ID_LENGTH.

ConnTag (MQBYTE128)
ConnTag est une balise que le gestionnaire de files d'attente associe aux ressources qui sont affectées par l'application lors de cette connexion. Chaque application ou instance d'application doit utiliser une valeur différente pour la balise, afin que le gestionnaire de files d'attente puisse sérialiser correctement l'accès aux ressources affectées. Cette zone est une zone d'entrée et sa valeur initiale est MQCT_NONE.

Voir les descriptions des options MQCNO_ * _CONN_TAG_ * pour plus de détails sur les valeurs à utiliser par les différentes applications. La balise cesse d'être valide lorsque l'application s'arrête ou émet l'appel MQDISC.

Remarque : Les valeurs de balise de connexion commençant par MQ en majuscules, en minuscules ou en casse mixte en ASCII ou EBCDIC sont réservées aux produits IBM . N'utilisez pas de valeurs de balise de connexion commençant par ces lettres.

Utilisez la valeur spéciale suivante si vous n'avez besoin d'aucune balise:

MQCT_AUCUN
La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQCT_NONE_ARRAY est également définie ; cette constante a la même valeur que MQCT_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Cette zone est utilisée lors de la connexion à un gestionnaire de files d'attente z/OS . Dans les autres environnements, indiquez la valeur MQCT_NONE.

La longueur de cette zone est indiquée par MQ_CONN_TAG_LENGTH. Cette zone est ignorée si *Version* est inférieur à MQCNO_VERSION_3.

Options (MQLONG)
Options qui contrôlent l'action de MQCONNX.

Options de comptabilité

Les options suivantes contrôlent le type de comptabilité si l'attribut de gestionnaire de files d'attente *AccountingConnOverride* est défini sur MQMON_ENABLED:

MQCNO_ACCOUNTING_MQI_ENABLED
Lorsque la collecte des données de surveillance est désactivée dans la définition du gestionnaire de files d'attente en définissant l'attribut *MQIAccounting* sur MQMON_OFF, la définition de cet indicateur active la collecte des données de comptabilité MQI.

MQCNO_ACCOUNTING_MQI_DISABLED
Lorsque la collecte des données de surveillance est désactivée dans la définition du gestionnaire de files d'attente en définissant l'attribut *MQIAccounting* sur MQMON_OFF, la définition de cet indicateur arrête la collecte des données de comptabilité MQI.

MQCNO_ACCOUNTING_Q_ENABLED
Lorsque la collecte des données de comptabilité des files d'attente est désactivée dans la définition du gestionnaire de files d'attente en définissant l'attribut *MQIAccounting* sur MQMON_OFF, la définition de cet indicateur active la collecte des données de comptabilité pour les files d'attente qui spécifient un gestionnaire de files d'attente dans la zone *MQIAccounting* de leur définition de file d'attente.

MQCNO_ACCOUNTING_Q_DISABLED
Lorsque la collecte des données de comptabilité des files d'attente est désactivée dans la définition de gestionnaire de files d'attente en définissant l'attribut *MQIAccounting* sur MQMON_OFF, cet indicateur désactive la collecte des données de comptabilité pour les files d'attente qui spécifient un gestionnaire de files d'attente dans la zone *MQIAccounting* de leur définition de file d'attente.

Si aucun de ces indicateurs n'est défini, la prise en compte de la connexion est celle définie dans les attributs du gestionnaire de files d'attente.

Options de liaison

Les options suivantes contrôlent le type de liaison WebSphere MQ à utiliser. Spécifiez une seule de ces options:

MQCNO_STANDARD_BINDING

L'application et l'agent du gestionnaire de files d'attente local (le composant qui gère les opérations de mise en file d'attente) s'exécutent dans des unités d'exécution distinctes (généralement, dans des processus distincts). Cette disposition maintient l'intégrité du gestionnaire de files d'attente, c'est-à-dire qu'elle protège le gestionnaire de files d'attente des programmes errants.

Si le gestionnaire de files d'attente prend en charge plusieurs types de liaison et que vous définissez MQCNO_STANDARD_BINDING, le gestionnaire de files d'attente utilise l'attribut *DefaultBindType* dans la strophe *Connection* du fichier `qm.ini` (ou l'entrée de registre Windows équivalente) pour sélectionner le type réel de liaison. Si cette section n'est pas définie, si la valeur ne peut pas être utilisée ou si elle n'est pas appropriée pour l'application, le gestionnaire de files d'attente sélectionne un type de liaison approprié. Le gestionnaire de files d'attente définit le type de liaison réel utilisé dans les options de connexion.

Utilisez MQCNO_STANDARD_BINDING dans les situations où l'application n'a peut-être pas été entièrement testée ou n'est pas fiable ou n'est pas fiable. MQCNO_STANDARD_BINDING est la valeur par défaut.

Cette option est prise en charge dans tous les environnements.

Si vous établissez une liaison à la bibliothèque `mqm` , une connexion serveur standard utilisant le type de liaison par défaut est tentée en premier. Si le chargement de la bibliothèque du serveur sous-jacent a échoué, une connexion client est tentée à la place.

- Si la variable d'environnement MQ_CONNECT_TYPE est spécifiée, l'une des options suivantes peut être fournie pour modifier le comportement de MQCONN ou MQCONNX si MQCNO_STANDARD_BINDING est spécifié. (sauf si MQCNO_FASTPATH_BINDING est spécifié avec MQ_CONNECT_TYPE défini sur LOCAL ou STANDARD pour permettre aux connexions fastpath d'être rétromigrées par l'administrateur sans modification associée de l'application:

Valeur	**Explication**
client	Une connexion client uniquement est tentée.
Fastpath	Cette valeur était prise en charge dans les éditions précédentes, mais elle est désormais ignorée si elle est spécifiée.
LOCAL	Une connexion serveur uniquement est tentée. Les connexions Fastpath sont rétromigrées vers une connexion serveur standard.
Standard	Prise en charge pour la compatibilité avec les éditions précédentes. Cette valeur est désormais traitée comme étant LOCAL.

- Si la variable d'environnement MQ_CONNECT_TYPE n'est pas définie lorsque MQCONNX est appelé, une connexion serveur standard utilisant le type de liaison par défaut est tentée. Si le chargement de la bibliothèque du serveur échoue, une connexion client est tentée.

MQCNO_FASTPATH_BINDING

L'application et l'agent du gestionnaire de files d'attente local font partie de la même unité d'exécution. Cela contraste avec la méthode classique de liaison, où l'application et l'agent du gestionnaire de files d'attente local s'exécutent dans des unités d'exécution distinctes.

MQCNO_FASTPATH_BINDING est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison ; le traitement se poursuit comme si l'option n'avait pas été spécifiée.

MQCNO_FASTPATH_BINDING peut être avantageux dans les situations où plusieurs processus consomment plus de ressources que la ressource globale utilisée par l'application.Une application qui utilise la liaison fastpath est appelée *application sécurisée*.

Prenez en compte les points importants suivants lorsque vous décidez d'utiliser la liaison de raccourci:

- L'utilisation de l'option MQCNO_FASTPATH_BINDING n'empêche pas une application de modifier ou d'altérer des messages et d'autres zones de données appartenant au gestionnaire de files d'attente. Utilisez cette option uniquement dans les situations où vous avez intégralement évalué ces problèmes.
- L'application ne doit pas utiliser de signaux asynchrones ou d'interruptions de temporisateur (telles que `sigkill`) avec MQCNO_FASTPATH_BINDING. Il existe également des restrictions sur l'utilisation des segments de mémoire partagée.
- L'application doit utiliser l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente.
- L'application doit se terminer avant que vous n'arrêtiez le gestionnaire de files d'attente à l'aide de la commande `endmqm` .
- Sous IBM i, le travail doit s'exécuter sous un profil utilisateur appartenant au groupe `QMQMADM` . En outre, le programme ne doit pas s'arrêter anormalement, sinon des résultats imprévisibles peuvent se produire.
- Sur les systèmes UNIX , l'identificateur d'utilisateur `mqm` doit être l'identificateur d'utilisateur effectif et l'identificateur de groupe `mqm` doit être l'identificateur de groupe effectif. Pour que l'application s'exécute de cette manière, configurez le programme de sorte qu'il appartienne à l'identificateur d'utilisateur `mqm` et à l'identificateur de groupe `mqm` , puis définissez les bits d'autorisation `setuid` et `setgid` sur le programme.

  WebSphere MQ Object Authority Manager (OAM) utilise toujours l'ID utilisateur réel pour la vérification des droits.
- Sous Windows, le programme doit être membre du groupe `mqm` . La liaison Fastpath n'est pas prise en charge pour les applications 64 bits.

L'option MQCNO_FASTPATH_BINDING est prise en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linuxet Windows. Sous z/OS, l'option est acceptée mais ignorée.

Pour plus d'informations sur les implications de l'utilisation d'applications de confiance, voir Restrictions pour les applications de confiance .

MQCNO_LIEN_PARTAGE

Avec MQCNO_SHARED_BINDING, l'application et l'agent du gestionnaire de files d'attente locales partagent certaines ressources. MQCNO_SHARED_BINDING est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison. Le traitement se poursuit comme si l'option n'avait pas été spécifiée.

MQCNO_LIEN_ISOLÉ_LIAISON

Dans ce cas, le processus d'application et l'agent du gestionnaire de files d'attente local sont isolés les uns des autres car ils ne partagent pas de ressources. MQCNO_ISOLATED_BINDING est ignoré si le gestionnaire de files d'attente ne prend pas en charge ce type de liaison. Le traitement se poursuit comme si l'option n'avait pas été spécifiée.

LIEN_CLIENT_MQCNO_BINDING

Spécifiez cette option pour que l'application tente une connexion client uniquement. Cette option présente les limitations suivantes:

- MQCNO_CLIENT_BINDING est rejeté sur z/OS avec MQRC_OPTIONS_ERROR.
- MQCNO_CLIENT_BINDING est rejeté avec MQRC_OPTIONS_ERROR s'il est spécifié avec une option de liaison MQCNO autre que MQCNO_STANDARD_BINDING.
- MQCNO_CLIENT_BINDING n'est pas disponible pour Java ou .NET car ils disposent de leurs propres mécanismes pour choisir le type de liaison.
- Si la variable d'environnement MQ_CONNECT_TYPE n'est pas définie lorsque MQCONNX est appelé, une connexion serveur standard utilisant le type de liaison par défaut est tentée. Si le chargement de la bibliothèque du serveur échoue, une connexion client est tentée.

MQCNO_LIEN_LOCAL_LOCAL

Indiquez cette option pour que l'application tente de se connecter au serveur. Si MQCNO_FASTPATH_BINDING, MQCNO_ISOLATED_BINDING ou MQCNO_SHARED_BINDING est également spécifié, la connexion est de ce type et est documentée dans cette section. Sinon, une connexion serveur standard est tentée à l'aide du type de liaison par défaut. MQCNO_LOCAL_BINDING présente les limitations suivantes:

- MQCNO_LOCAL_BINDING est ignoré sur z/OS.
- MQCNO_LOCAL_BINDING est rejeté avec MQRC_OPTIONS_ERROR s'il est spécifié avec une option de reconnexion MQCNO autre que MQCNO_RECONNECT_AS_DEF.
- MQCNO_LOCAL_BINDING n'est pas disponible pour Java ou .NET car ils disposent de leurs propres mécanismes pour choisir le type de liaison.
- Si la variable d'environnement MQ_CONNECT_TYPE n'est pas définie lorsque MQCONNX est appelé, une connexion serveur standard utilisant le type de liaison par défaut est tentée. Si le chargement de la bibliothèque du serveur échoue, une connexion client est tentée.

Sous AIX, HP-UX, Solaris, Linuxet Windows, vous pouvez utiliser la variable d'environnement MQ_CONNECT_TYPE avec le type de liaison spécifié par la zone *Options* pour contrôler le type de liaison utilisé. Si vous spécifiez cette variable d'environnement, elle doit avoir la valeur FASTPATH ou STANDARD ; s'il a une valeur différente, il est ignoré. La valeur de la variable d'environnement est sensible à la casse ; voir Variable d'environnement MQCONNX pour plus d'informations.

La variable d'environnement et la zone *Options* interagissent comme suit:

- Si vous omettez la variable d'environnement ou que vous lui attribuez une valeur qui n'est pas prise en charge, l'utilisation de la liaison raccourci est déterminée uniquement par la zone *Options* .
- Si vous attribuez à la variable d'environnement une valeur prise en charge, la liaison fastpath est utilisée uniquement si *les deux* variables d'environnement et la zone *Options* spécifient la liaison fastpath.

Options de balise de connexion

Ces options sont prises en charge uniquement lors de la connexion à un gestionnaire de files d'attente z/OS et contrôlent l'utilisation de la balise de connexion *ConnTag*. Vous ne pouvez spécifier qu'une seule des options suivantes:

MQCNO_SERIALIZE_CONN_TAG_Q_MGR

Cette option demande l'utilisation exclusive de la balise de connexion dans le gestionnaire de files d'attente local. Si la balise de connexion est déjà utilisée dans le gestionnaire de files d'attente local, l'appel MQCONNX échoue avec le code anomalie MQRC_CONN_TAG_IN_USE. Le résultat de l'appel n'est pas affecté par l'utilisation de la balise de connexion ailleurs dans le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

MQCNO_SERIALIZE_CONN_TAG_QSG

Cette option demande l'utilisation exclusive de la balise de connexion dans le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local. Si la balise de connexion est déjà utilisée dans le groupe de partage de files d'attente, l'appel MQCONNX échoue avec le code anomalie MQRC_CONN_TAG_IN_USE.

MQCNO_RESTRICT_CONN_TAG_Q_MGR

Cette option demande l'utilisation partagée de la balise de connexion dans le gestionnaire de files d'attente local. Si la balise de connexion est déjà utilisée dans le gestionnaire de files d'attente local, l'appel MQCONNX peut aboutir si l'application demandeuse s'exécute dans la même portée de traitement que l'utilisateur existant de la balise. Si cette condition n'est pas satisfaite, l'appel MQCONNX échoue avec le code anomalie MQRC_CONN_TAG_IN_USE. Le résultat de l'appel n'est pas affecté par l'utilisation de la balise de connexion ailleurs dans le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

- Les applications doivent s'exécuter dans le même espace adresse MVS pour partager la balise de connexion. Si l'application utilisant la balise de connexion est une application client, MQCNO_RESTRICT_CONN_TAG_Q_MGR n'est pas autorisé.

MQCNO_RESTRICT_CONN_TAG_QSG

Cette option demande l'utilisation partagée de la balise de connexion dans le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local. Si la balise de connexion est déjà utilisée dans le groupe de partage de files d'attente, l'appel MQCONNX peut aboutir à condition que l'application demandeuse s'exécute dans la même portée de traitement et soit connectée au même gestionnaire de files d'attente que l'utilisateur existant de la balise.

Si ces conditions ne sont pas satisfaites, l'appel MQCONNX échoue avec le code anomalie MQRC_CONN_TAG_IN_USE.

- Les applications doivent s'exécuter dans le même espace adresse MVS pour partager la balise de connexion. Si l'application utilisant la balise de connexion est une application client, MQCNO_RESTRICT_CONN_TAG_QSG n'est pas autorisé.

Si aucune de ces options n'est spécifiée, *ConnTag* n'est pas utilisé. Ces options ne sont pas valides si *Version* est inférieur à MQCNO_VERSION_3.

Options de partage de descripteur

Ces options sont prises en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux et Windows. Ils contrôlent le partage des descripteurs entre différentes unités d'exécution (unités de traitement parallèle) au sein d'un même processus. Vous ne pouvez spécifier qu'une seule des options suivantes:

MQCNO_HANDLE_SHARE_NONE

Cette option indique que les descripteurs de connexion et d'objet ne peuvent être utilisés que par l'unité d'exécution à l'origine de l'allocation du descripteur (c'est-à-dire l'unité d'exécution qui a émis l'appel MQCONN, MQCONNX ou MQOPEN). Les descripteurs ne peuvent pas être utilisés par d'autres unités d'exécution appartenant au même processus.

MQCNO_HANDLE_SHARE_BLOCK

Cette option indique que les descripteurs de connexion et d'objet alloués par une unité d'exécution d'un processus peuvent être utilisés par d'autres unités d'exécution appartenant au même processus. Toutefois, une seule unité d'exécution à la fois peut utiliser une poignée particulière, c'est-à-dire que seule l'utilisation en série d'une poignée est autorisée. Si une unité d'exécution tente d'utiliser un descripteur déjà utilisé par une autre unité d'exécution, l'appel se bloque (attend) jusqu'à ce que le descripteur soit disponible.

MQCNO_HANDLE_SHARE_NO_BLOCK

Identique à MQCNO_HANDLE_SHARE_BLOCK, sauf que si le descripteur est utilisé par une autre unité d'exécution, l'appel se termine immédiatement avec MQCC_FAILED et MQRC_CALL_IN_PROGRESS au lieu de se bloquer jusqu'à ce que le descripteur soit disponible.

Une unité d'exécution peut avoir zéro ou un descripteur non partagé:

- Chaque appel MQCONN ou MQCONNX qui spécifie MQCNO_HANDLE_SHARE_NONE renvoie un nouveau descripteur non partagé sur le premier appel, et le même descripteur non partagé sur le deuxième appel et les appels ultérieurs (en supposant qu'il n'y ait pas d'appel MQDISC). Le code anomalie est MQRC_ALREADY_CONNECTED pour le deuxième appel et les appels ultérieurs.
- Chaque appel MQCONNX qui spécifie MQCNO_HANDLE_SHARE_BLOCK ou MQCNO_HANDLE_SHARE_NO_BLOCK renvoie un nouveau descripteur partagé sur chaque appel.

Les descripteurs d'objet héritent des mêmes propriétés de partage que le descripteur de connexion indiqué dans l'appel MQOPEN qui a créé le descripteur d'objet. En outre, les unités de travail héritent des

mêmes propriétés de partage que le descripteur de connexion utilisé pour démarrer l'unité de travail ; si l'unité de travail est démarrée dans une unité d'exécution à l'aide d'un descripteur partagé, l'unité de travail peut être mise à jour dans une autre unité d'exécution à l'aide du même descripteur.

Si vous ne spécifiez pas d'option de partage de descripteur, la valeur par défaut est déterminée par l'environnement:

- Dans l'environnement Microsoft Transaction Server (MTS), la valeur par défaut est MQCNO_HANDLE_SHARE_BLOCK.
- Dans d'autres environnements, la valeur par défaut est la même que MQCNO_HANDLE_SHARE_NONE.

Options de reconnexion

Les options de reconnexion déterminent si une connexion peut être reconnectée. Seules les connexions client sont reconnectables.

MQCNO_RECONNECT_AS_DEF

L'option de reconnexion est résolue avec sa valeur par défaut. Si aucune valeur par défaut n'est définie, la valeur de cette option est DISABLED . La valeur de l'option est transmise au serveur et peut être interrogée par PCF et MQSC.

MQCNO_RECONNECT

L'application peut être reconnectée à n'importe quel gestionnaire de files d'attente cohérent avec la valeur du paramètre `QmgrName` de `MQCONNX`. Utilisez l'option MQCNO_RECONNECT uniquement s'il n'y a pas d'affinité entre l'application client et le gestionnaire de files d'attente avec lequel elle a initialement établi une connexion. La valeur de l'option est transmise au serveur et peut être interrogée par PCF et MQSC.

MQCNO_RECONNECT_DISABLED

L'application ne peut pas être reconnectée. La valeur de l'option n'est pas transmise au serveur.

MQCNO_RECONNECT_Q_MGR

L'application ne peut être reconnectée qu'au gestionnaire de files d'attente avec lequel elle s'est connectée à l'origine. Utilisez cette valeur si un client peut être reconnecté, mais qu'il existe une affinité entre l'application client et le gestionnaire de files d'attente avec lequel il a établi une connexion. Choisissez cette valeur si vous souhaitez qu'un client se reconnecte automatiquement à l'instance de secours d'un gestionnaire de files d'attente haute disponibilité. La valeur de l'option est transmise au serveur et peut être interrogée par PCF et MQSC.

Utilisez les options MQCNO_RECONNECT, MQCNO_RECONNECT_DISABLED et MQCNO_RECONNECT_Q_MGR uniquement pour les connexions client. Si les options sont utilisées pour une connexion de liaison, `MQCONNX` échoue avec le code achèvement `MQCC_FAILED` et le code anomalie `MQRC_OPTIONS_ERROR`. La reconnexion automatique du client n'est pas prise en charge par WebSphere MQ classes for Java

Options de partage de conversation

Les options suivantes s'appliquent uniquement aux connexions client TCP/IP. Pour les canaux SNA, SPX et NetBios , ces valeurs sont ignorées et le canal s'exécute comme dans les versions précédentes du produit

MQCNO_NO_CONV_SHARING

Cette option n'autorise pas le partage de conversation.

Vous pouvez utiliser MQCNO_NO_NO_CONV_SHARING dans des situations où les conversations sont fortement chargées et, par conséquent, où un conflit est possible à l'extrémité de connexion serveur

de l'instance de canal sur laquelle les conversations partagées existent. MQCNO_NO_CNV_SHARING se comporte comme sharecnv (1) lorsqu'il est connecté à un canal qui prend en charge le partage des conversations, et sharecnv (0) lorsqu'il est connecté à un canal qui ne prend pas en charge le partage des conversations.

MQCNO_TOUS_CONVS_SHARE

Cette option permet le partage des conversations ; l'application ne limite pas le nombre de connexions sur l'instance de canal. Il s'agit de la valeur par défaut.

Si l'application indique que l'instance de canal peut être partagée, mais que la définition *SharingConversations* (SHARECNV) sur l'extrémité de connexion serveur du canal est définie sur une, aucun partage n'a lieu et aucun avertissement n'est envoyé à l'application.

De même, si l'application indique que le partage est autorisé mais que la définition *SharingConversations* de la connexion serveur est définie sur zéro, aucun avertissement n'est émis et l'application présente le même comportement qu'un client dans les versions du produit antérieures à la version 7.0; le paramètre d'application relatif au partage des conversations est ignoré.

MQCNO_NO_CONV_SHARING et MQCNO_ALL_CONVS_SHARE s'excluent mutuellement. Si les deux options sont spécifiées sur une connexion particulière, la connexion est rejetée avec le code anomalie MQRC_OPTIONS_ERROR.

Options de définition de canal

Les options suivantes contrôlent l'utilisation de la structure de définition de canal transmise dans le MQCNO:

MQCNO_CD_FOR_OUTPUT_ONLY

Cette option permet d'utiliser la structure de définition de canal dans le MQCNO uniquement pour renvoyer le nom de canal utilisé lors d'un appel MQCONNX réussi.

Si une structure de définition de canal valide n'est pas fournie, l'appel échoue avec le code anomalie MQRC_CD_ERROR.

Si l'application n'est pas exécutée en tant que client, l'option est ignorée.

Le nom de canal renvoyé peut être utilisé lors d'un appel MQCONNX ultérieur à l'aide de l'option MQCNO_USE_CD_SELECTION pour se reconnecter à l'aide de la même définition de canal. Cela peut être utile lorsqu'il existe plusieurs définitions de canal applicables dans la table des canaux client.

MQCNO_USE_CD_SELECTION

Cette option permet à l'appel MQCONNX de se connecter à l'aide du nom de canal contenu dans la structure de définition de canal transmise dans le MQCNO.

Si la variable d'environnement MQSERVER est définie, la définition de canal qu'elle définit est utilisée. Si MQSERVER n'est pas défini, la table de canaux client est utilisée.

Si aucune définition de canal avec un nom de canal et un nom de gestionnaire de files d'attente correspondants n'est trouvée, l'appel échoue avec le code anomalie MQRC_Q_MGR_NAME_ERROR.

Si une structure de définition de canal valide n'est pas fournie, l'appel échoue avec le code anomalie MQRC_CD_ERROR.

Si l'application n'est pas exécutée en tant que client, l'option est ignorée.

Option par défaut

Si vous n'avez besoin d'aucune des options décrites ci-dessus, vous pouvez utiliser l'option suivante:

MQCNO_AUCUN

Aucune option n'est spécifiée.

Utilisez MQCNO_NONE pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option MQCNO_ *, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

SecurityParmsDécalage (MQLONG)
SecurityParmsLe décalage est le décalage en octets de la structure MQCSP à partir du début de la structure MQCNO. Le décalage peut être positif ou négatif. Cette zone est une zone d'entrée, avec une valeur initiale de 0.

Cette zone est ignorée si *Version* est inférieure à MQCNO_VERSION_5.

La structure MQCSP est définie dans «MQCSP-Paramètres de sécurité», à la page 314.

SecurityParmsPtr (PMQCSP)
SecurityParmsPtr est l'adresse de la structure MQCSP, utilisée pour spécifier un ID utilisateur et un mot de passe pour l'authentification par le service d'autorisation. Cette zone est une zone d'entrée et sa valeur initiale est un pointeur nul ou des octets nuls.

Cette zone est ignorée si *Version* est inférieure à MQCNO_VERSION_5.

La structure MQCSP est définie dans «MQCSP-Paramètres de sécurité», à la page 314.

SSLConfigOffset (MQLONG)
SSLConfigOffset est le décalage en octets d'une structure MQSCO depuis le début de la structure MQCNO. Le décalage peut être positif ou négatif. Cette zone est une zone d'entrée, avec une valeur initiale de 0.

Utilisez *SSLConfigOffset* uniquement lorsque l'application émettant l'appel MQCONNX s'exécute en tant que client WebSphere MQ MQI. Pour plus d'informations sur l'utilisation de cette zone, voir la description de la zone *SSLConfigPtr* .

Cette zone est ignorée si *Version* est inférieur à MQCNO_VERSION_4.

SSLConfigPtr (PMQSCO)
SSLConfigPtr est une zone d'entrée. Sa valeur initiale est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

Utilisez *SSLConfigPtr* et *SSLConfigOffset* uniquement lorsque l'application émettant l'appel MQCONNX s'exécute en tant que client WebSphere MQ MQI et que le protocole de canal est TCP/IP. Si l'application n'est pas exécutée en tant que client WebSphere MQ , ou si le protocole de canal n'est pas TCP/IP, *SSLConfigPtr* et *SSLConfigOffset* sont ignorés.

En spécifiant *SSLConfigPtr* ou *SSLConfigOffset*, plus *ClientConnPtr* ou *ClientConnOffset*, l'application peut contrôler l'utilisation de SSL pour la connexion client. Lorsque les informations SSL sont spécifiées de cette manière, les variables d'environnement MQSSLKEYR et MQSSLCRYP sont ignorées ; toutes les informations liées à SSL dans la table de définition de canal du client (CCDT) sont également ignorées.

Les informations SSL ne peuvent être spécifiées que sur:

- Premier appel MQCONNX du processus client, ou
- Un appel MQCONNX ultérieur lorsque toutes les connexions SSL/TLS précédentes au gestionnaire de files d'attente ont été conclues à l'aide de MQDISC.

Il s'agit des seuls états dans lesquels l'environnement SSL à l'échelle du processus peut être initialisé. Si un appel MQCONNX est émis en spécifiant des informations SSL alors que l'environnement SSL existe déjà, les informations SSL de l'appel sont ignorées et la connexion est établie à l'aide de l'environnement SSL existant ; l'appel renvoie le code achèvement MQCC_WARNING et le code anomalie MQRC_SSL_ALREADY_INITIALISEE dans ce cas.

Vous pouvez fournir la structure MQSCO de la même manière que la structure MQCD, soit en spécifiant une adresse dans *SSLConfigPtr*, soit en spécifiant un décalage dans *SSLConfigOffset*; voir la

description de *ClientConnPtr* pour plus de détails sur la procédure à suivre. Toutefois, vous ne pouvez pas utiliser plus d'une des options *SSLConfigPtr* et *SSLConfigOffset*; l'appel échoue avec le code anomalie MQRC_SSL_CONFIG_ERROR. si les deux sont différents de zéro.

Une fois l'appel MQCONNX terminé, la structure MQSCO n'est plus référencée.

Cette zone est ignorée si *Version* est inférieur à MQCNO_VERSION_4.

Remarque : Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

StrucId (MQCHAR4)
StrucId est toujours une zone d'entrée. Sa valeur initiale est MQCNO_STRUC_ID.

La valeur doit être:

ID_STRUC_MQCNO
Identificateur de la structure des options de connexion.

Pour le langage de programmation C, la constante MQCNO_STRUC_ID_ARRAY est également définie ; cette constante a la même valeur que MQCNO_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Version (MQLONG)
La version est toujours une zone d'entrée. Sa valeur initiale est MQCNO_VERSION_1.

Il doit s'agir de l'une des valeurs suivantes :

MQCNO_VERSION_1
Structure d'options de connexion Version-1 .

MQCNO_VERSION_2
Structure d'options de connexion Version-2 .

MQCNO_VERSION_3
Structure d'options de connexion Version-3 .

MQCNO_VERSION_4
Structure des options de connexion Version-4 .

MQCNO_VERSION_5
Structure d'options de connexion Version-5 .

Cette version de la structure MQCNO étend MQCNO_VERSION_3 sur z/OSet MQCNO_VERSION_4 sur toutes les autres plateformes.

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

MQCNO_CURRENT_VERSION
Version actuelle de la structure des options de connexion.

Valeurs initiales et déclarations de langue pour MQCNO

Tableau 488. Valeurs initiales des zones dans MQCNO pour MQCNO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQCNO	'CNO¬'
Version	MQCNO_VERSION_1	1
Options	MQCNO_AUCUN	0
ClientConnOffset	Aucun	0
ClientConnPtr	Aucun	Pointeur null ou octets null

Tableau 488. Valeurs initiales des zones dans MQCNO pour MQCNO (suite)

Nom de zone	Nom de la constante	Valeur de la constante
ConnTag	MQCT_AUCUN	Valeurs NULL
SSLConfigPtr	Aucun	Pointeur null ou octets null
SSLConfigOffset	Aucun	0
ConnectionId	Aucun	Pointeur null ou octets null
SecurityParmsOffset	Aucun	Pointeur null ou octets null
SecurityParmsPtr	Aucun	Pointeur null ou octets null

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQCNO_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCNO MyCNO = {MQCNO_DEFAULT};
```

Déclaration C

```
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 */
 };
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Paramètres de sécurité

Le tableau suivant récapitule les zones de la structure.

Tableau 489. Zones dans MQCSP

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
AuthenticationType	Type d'authentification	AuthenticationType
Reserved1	Obligatoire pour l'alignement des pointeurs sous IBM i	Reserved1
CSPUserIdPtr	Adresse de l'ID utilisateur	CSPUserIdPtr
CSPUserIdOffset	Décalage de l'ID utilisateur	CSPUserIdDécalage
CSPUserIdLength	Longueur de l'ID utilisateur	CSPUserIdLongueur
Reserved2	Obligatoire pour l'alignement des pointeurs sous IBM i	Reserved2
CSPPasswordPtr	Adresse du mot de passe	CSPPasswordPtr
CSPPasswordOffset	Décalage du mot de passe	CSPPasswordOffset
CSPPasswordLength	Longueur du mot de passe	CSPPasswordLength

Présentation de MQCSP

Disponibilité: Tous les produits WebSphere MQ .

Objectif: La structure MQCSP permet au service d'autorisation d'authentifier un ID utilisateur et un mot de passe. Vous spécifiez la structure des paramètres de sécurité de connexion MQCSP sur un appel MQCONNX.

Jeu de caractères et codage: les données dans MQCSP doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et MQENC_NATIVE, respectivement.

Zones pour MQCSP

La structure MQCSP contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

AuthenticationType (MQLONG)
AuthenticationType est une zone d'entrée. Sa valeur initiale est MQCSP_AUTH_NONE.

Il s'agit du type d'authentification à effectuer. Les valeurs admises sont :

MQCSP_AUTH_NONE
N'utilisez pas les zones d'ID utilisateur et de mot de passe.

MQCSP_AUTH_USER_ID_AND_PWD
Authentifiez les zones d'ID utilisateur et de mot de passe.

CSPPasswordLength (MQLONG)
Cette zone indique la longueur du mot de passe à utiliser pour l'authentification.

La longueur maximale du mot de passe dépend de la plateforme. Voir ID utilisateur. Si la longueur du mot de passe est supérieure à la longueur maximale autorisée, la demande d'authentification échoue avec MQRC_NOT_AUTHORIZED.

Cette zone est une zone d'entrée. La valeur initiale de cette zone est 0.

CSPPasswordOffset (MQLONG)
Il s'agit du décalage en octets du mot de passe à utiliser pour l'authentification. Le décalage peut être positif ou négatif.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

CSPPasswordPtr (MQPTR)
Il s'agit de l'adresse en octets du mot de passe à utiliser pour l'authentification.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQCNO_VERSION_5.

CSPUserIdLongueur (MQLONG)
Cette zone indique la longueur de l'ID utilisateur à utiliser pour l'authentification.

La longueur maximale de l'ID utilisateur dépend de la plateforme. Voir ID utilisateur. Si la longueur de l'ID utilisateur est supérieure à la longueur maximale autorisée, la demande d'authentification échoue avec MQRC_NOT_AUTHORIZED.

Cette zone est une zone d'entrée. La valeur initiale de cette zone est 0.

CSPUserIdDécalage (MQLONG)
Décalage en octets de l'ID utilisateur à utiliser pour l'authentification. Le décalage peut être positif ou négatif.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

CSPUserIdPtr (MQPTR)
Il s'agit de l'adresse en octets de l'ID utilisateur à utiliser pour l'authentification.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQCNO_VERSION_5.

Reserved1 (MQBYTE4)
Zone réservée, requise pour l'alignement des pointeurs sur IBM i.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est nulle.

Reserved2 (MQBYTE8)
Zone réservée, requise pour l'alignement des pointeurs sur IBM i.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est nulle.

StrucId (MQCHAR4)
Identificateur de structure.

La valeur doit être:

MQCSP_STRUC_ID
Identificateur de la structure des paramètres de sécurité.

Pour le langage de programmation C, la constante MQCSP_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQCSP_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCSPSTRUC_ID.

Version (MQLONG)
Numéro de version de la structure.

La valeur doit être:

MQCSP_VERSION_1
Structure des paramètres de sécurité Version-1 .

La constante suivante indique le numéro de version de la version en cours:

MQCSP_CURRENT_VERSION
Version actuelle de la structure des paramètres de sécurité.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCSP_VERSION_1.

Valeurs initiales et déclarations de langue pour MQCSP

Tableau 490. Valeurs initiales des zones dans MQCSP pour MQCSP

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	MQCSP_STRUC_ID	'CSP¬'
Version	MQCSP_VERSION_1	1
AuthenticationType	Aucun	MQCSP_AUTH_NONE
Reserved1	Aucun	Chaîne nulle ou blancs
CSPUserIdPtr	Aucun	Pointeur null ou octets null
CSPUserIdOffset	Aucun	0
CSPUserIdLength	Aucun	0
Reserved2	Aucun	Chaîne nulle ou blancs
CSPPasswordPtr	Aucun	Pointeur null ou octets null
CSPPasswordOffset	Aucun	0
CSPPasswordLength	Aucun	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQCSP_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCSP MyCSP = {MQCSP_DEFAULT};
```

Déclaration C

```
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 */
  };
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration Visual Basic

```
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-Structure des options de rappel de contrôle

Le tableau suivant récapitule les zones de la structure. Structure spécifiant la fonction de rappel de contrôle.

Tableau 491. Zones dans MQCTLO		
Zone	**Description**	**Topic**
StrucID	Identificateur de structure	StrucID
Version	Numéro de version de structure	Version
Options	Options	Options
Reserved	Réservé, zone	Options
ConnectionArea	Zone de la fonction de rappel à utiliser	ConnectionArea

Présentation de MQCTLO

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OSet WebSphere MQ clients MQI connectés à ces systèmes. Présentation de la structure MQCTLO.

Objectif: La structure MQCTLO est utilisée pour spécifier les options relatives à une fonction de rappel de contrôle.

La structure est un paramètre d'entrée et de sortie dans l'appel «MQCTL-Rappels de contrôle», à la page 663 .

Version: la version actuelle de MQCTLO est MQCTLO_VERSION_1.

Jeu de caractères et codage: les données de MQCTLO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Zones pour MQCTLO

Liste alphabétique des zones de la structure MQCTLO.

La structure MQCTLO contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

ConnectionArea (MQPTR)
Structure des options de contrôle-Zone ConnectionArea

Il s'agit d'une zone disponible que la fonction de rappel peut utiliser.

Le gestionnaire de files d'attente ne prend aucune décision en fonction du contenu de cette zone et est transmis tel quel à la zone «ConnectionArea (MQPTR)», à la page 265 de la structure MQCBC, qui est un paramètre d'entrée du rappel.

Cette zone est ignorée pour toutes les opérations autres que MQOP_START et MQOP_START_WAIT.

Il s'agit d'une zone d'entrée et de sortie de la fonction de rappel. La valeur initiale de cette zone est un pointeur NULL ou des octets NULL.

Options (MQLONG)
Structure des options de contrôle-Zone Options

Options qui contrôlent l'action de MQCTL.

MQCTLO_FAIL_IF_QUIESCING
Forcez l'échec de l'appel MQCTL si le gestionnaire de files d'attente ou la connexion est à l'état de mise au repos.

Spécifiez MQGMO_FAIL_IF_QUIESCING, dans les options MQGMO transmises à l'appel MQCB, pour envoyer une notification aux consommateurs de message lorsqu'ils sont mis au repos.

MQCTLO_AFFINITÉ_UNITÉS d'exécution
Cette option informe le système que l'application requiert que tous les consommateurs de messages, pour la même connexion, soient appelés sur la même unité d'exécution. Cette unité d'exécution sera utilisée pour tous les appels des consommateurs jusqu'à l'arrêt de la connexion.

Option par défaut: si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

MQCTLO_NONE
Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut. MQCTLO_NONE est défini pour faciliter la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de la zone *Options* est MQCTLO_NONE.

Réservé (MQLONG)

Il s'agit d'une zone réservée. La valeur doit être zéro.

StrucId (MQCHAR4)
Structure des options de contrôle-Zone StrucId

Il s'agit de l'identificateur de structure ; la valeur doit être:

MQCTLO_STRUC_ID
Identificateur de la structure des options de contrôle.

Pour le langage de programmation C, la constante MQCTLO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQCTLO_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCTLO_STRUC_ID.

Version (MQLONG)
Structure des options de contrôle-Zone Version

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQCTLO_VERSION_1
Version-1 Structure des options de contrôle.

La constante suivante indique le numéro de version de la version en cours:

MQCTLO_CURRENT_VERSION
Version actuelle de la structure des options de contrôle.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQCTLO_VERSION_1.

Valeurs initiales et déclarations de langage pour MQCTLO
Structure des options de contrôle-Valeurs initiales

Tableau 492. Valeurs initiales des zones dans MQCTLO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	MQCTLO_STRUC_ID	'CTLO'
Version	MQCTLO_VERSION_1	1
Options	MQCTLO_NONE	Valeurs NULL
Reserved	Réservé, zone	

Tableau 492. Valeurs initiales des zones dans MQCTLO (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
ConnectionArea	Aucun	Pointeur null ou octets null

Remarques :

1. Dans le langage de programmation C, la variable macroMQCTLO_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQCTLO MyCTLO = {MQCTLO_DEFAULT};
```

Déclaration C
Structure des options de contrôle-Déclaration en langage C

```
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 */
 };
```

Déclaration COBOL

```
** 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
```

Déclaration PL/I

```
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 */
```

En-tête MQDH-Distribution

Le tableau suivant récapitule les zones de la structure.

Tableau 493. Zones dans MQDH		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId

Tableau 493. Zones dans MQDH (suite)		
Zone	**Description**	**Topic**
Version	Numéro de version de structure	Version
StrucLength	Longueur de la structure MQDH plus les enregistrements suivants	StrucLength
Encoding	Codage numérique des données qui suit une grappe d'enregistrements MQPMR	Codage
CodedCharSetId	Identificateur de jeu de caractères des données qui suit la grappe d'enregistrements MQPMR	CodedCharSetId
Format	Nom du format des données qui suit la grappe d'enregistrements MQPMR	Format
Flags	Indicateurs généraux	Indicateurs
PutMsgRecFields	Indicateurs identifiant les zones MQPMR présentes	PutMsgRecFields
RecsPresent	Nombre d'enregistrements d'objet présents	RecsPresent
ObjectRecOffset	Décalage du premier enregistrement d'objet à partir du début de MQDH	DécalageObjectRec
PutMsgRecOffset	Décalage du premier enregistrement d'insertion de message à partir du début de MQDH	PutMsgRecOffset

Présentation de MQDH

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ connectés à ces systèmes.

Objectif: La structure MQDH décrit les données supplémentaires présentes dans un message lorsque ce message est un message de liste de distribution stocké dans une file d'attente de transmission. Un message de liste de distribution est un message envoyé à plusieurs files d'attente de destination. Les données supplémentaires sont constituées de la structure MQDH suivie d'un tableau d'enregistrements MQOR et d'un tableau d'enregistrements MQPMR.

Cette structure est utilisée par les applications spécialisées qui placent les messages directement dans les files d'attente de transmission ou qui suppriment les messages des files d'attente de transmission (par exemple, les agents MCA).

Les applications qui souhaitent placer des messages dans des listes de distribution ne doivent pas utiliser cette structure. A la place, ils doivent utiliser la structure MQOD pour définir les destinations dans la liste de distribution et la structure MQPMO pour spécifier les propriétés de message ou recevoir des informations sur les messages envoyés aux destinations individuelles.

Nom de format: MQFMT_DIST_HEADER.

Jeu de caractères et codage: les données dans MQDH doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE.

Définissez le jeu de caractères et le codage de la MQDH dans les zones *CodedCharSetId* et *Encoding* dans:

- MQMD (si la structure MQDH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQDH (tous les autres cas).

Utilisation: Lorsqu'une application insère un message dans une liste de distribution et que certaines ou toutes les destinations sont distantes, le gestionnaire de files d'attente préfixe les données de message d'application avec les structures MQXQH et MQDH et place le message dans la file d'attente de transmission appropriée. Les données se trouvent donc dans l'ordre suivant lorsque le message se trouve dans une file d'attente de transmission:

- Structure MQXQH
- Structure MQDH plus tableaux d'enregistrements MQOR et MQPMR
- Données de message d'application

En fonction des destinations, le gestionnaire de files d'attente peut générer plusieurs messages de ce type et les placer dans des files d'attente de transmission différentes. Dans ce cas, les structures MQDH de ces messages identifient différents sous-ensembles des destinations définies par la liste de distribution ouverte par l'application.

Une application qui insère un message de liste de distribution directement dans une file d'attente de transmission doit se conformer à la séquence décrite ci-dessus et doit s'assurer que la structure MQDH est correcte. Si la structure MQDH n'est pas valide, le gestionnaire de files d'attente peut faire échouer l'appel MQPUT ou MQPUT1 avec le code anomalie MQRC_DH_ERROR.

Vous pouvez stocker des messages dans une file d'attente sous forme de liste de distribution uniquement si vous avez défini la file d'attente comme pouvant prendre en charge les messages de liste de distribution (voir l'attribut de file d'attente *DistLists* décrit dans «Attributs des files d'attente», à la page 824). Si une application insère un message de liste de distribution directement dans une file d'attente qui ne prend pas en charge les listes de distribution, le gestionnaire de files d'attente divise le message de liste de distribution en messages individuels et place ces messages dans la file d'attente.

Zones pour MQDH

La structure MQDH contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

CodedCharSetId (MQLONG)

Il s'agit de l'identificateur de jeu de caractères des données qui suivent les tableaux des enregistrements MQOR et MQPMR ; il ne s'applique pas aux données de type caractère dans la structure MQDH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Vous pouvez utiliser la valeur spéciale suivante:

MQCCSI_HÉRITER
Hérite de l'identificateur de jeu de caractères de cette structure.

Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, l'appel MQGET ne renvoie pas la valeur MQCCSI_INHERIT.

Vous ne pouvez pas utiliser MQCCSI_INHERIT si la valeur de la zone *PutApplType* dans MQMD est MQAT_BROKER.

Cette valeur est prise en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus les clients WebSphere MQ connectés à ces systèmes.

La valeur initiale de cette zone est MQCCSI_UNDEFINED.

Codage (MQLONG)

Il s'agit du codage numérique des données qui suit les tableaux des enregistrements MQOR et MQPMR ; il ne s'applique pas aux données numériques de la structure MQDH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

Indicateurs (MQLONG)

Vous pouvez spécifier l'indicateur suivant:

MQDHF_NEW_MSG_IDS
Générez un nouvel identificateur de message pour chaque destination de la liste de distribution. Définissez cette option uniquement si aucun enregistrement de message d'insertion n'est présent ou si les enregistrements sont présents mais qu'ils ne contiennent pas la zone *MsgId* .

L'utilisation de cet indicateur diffère la génération des identificateurs de message jusqu'au moment où le message de liste de distribution est finalement divisé en messages individuels. Cela réduit la quantité d'informations de contrôle qui doivent être transmises avec le message de liste de distribution.

Lorsqu'une application insère un message dans une liste de distribution, le gestionnaire de files d'attente définit MQDHF_NEW_MSG_IDS dans la MQDH qu'elle génère lorsque les deux conditions suivantes sont remplies:

- Aucun enregistrement d'insertion de message n'est fourni par l'application ou les enregistrements fournis ne contiennent pas la zone *MsgId* .
- La zone *MsgId* dans MQMD est MQMI_NONE ou la zone *Options* dans MQPMO inclut MQPMO_NEW_MSG_ID

Si aucun indicateur n'est nécessaire, spécifiez ce qui suit:

MQDHF_NONE
Aucun indicateur n'a été spécifié. MQDHF_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La valeur initiale de cette zone est MQDHF_NONE.

Format (MQCHAR8)

Il s'agit du nom de format des données qui suivent les tableaux des enregistrements MQOD et MQPMR (selon la dernière occurrence).

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La valeur initiale de cette zone est MQFMT_NONE.

ObjectRec(MQLONG)

Indique le décalage en octets du premier enregistrement du tableau des enregistrements d'objet MQOR contenant les noms des files d'attente de destination. Ce tableau contient *RecsPresent* enregistrements. Ces enregistrements (plus les octets ignorés entre le premier enregistrement d'objet et la zone précédente) sont inclus dans la longueur indiquée par la zone *StrucLength* .

Une liste de distribution doit toujours contenir au moins une destination. Par conséquent, *ObjectRecOffset* doit toujours être supérieur à zéro.

La valeur initiale de cette zone est 0.

PutMsgRecFields (MQLONG)

Vous pouvez spécifier aucun ou plusieurs des indicateurs suivants:

ID MQPMRF_MSG_ID
La zone d'identificateur de message est présente.

ID_CORREL_MQPMRF_
La zone d'identificateur de corrélation est présente.

ID_GROUPE_MQPMRF_ID
La zone d'identificateur de groupe est présente.

MQPMRF_FEEDBACK
Un champ de commentaires est présent.

MQPMRF_COMPTING_TOKEN
La zone de jeton de comptabilité est présente.

Si aucune zone MQPMR n'est présente, indiquez ce qui suit:

MQPMRF_AUCUN
Aucune zone d'enregistrement d'insertion de message n'est présente. MQPMRF_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La valeur initiale de cette zone est MQPMRF_NONE.

PutMsgRecOffset (MQLONG)

Indique le décalage en octets du premier enregistrement dans le tableau des enregistrements de message d'insertion MQPMR contenant les propriétés de message. S'il est présent, ce tableau contient *RecsPresent* enregistrements. Ces enregistrements (plus les octets ignorés entre le premier enregistrement de message inséré et la zone précédente) sont inclus dans la longueur indiquée par la zone *StrucLength* .

Les enregistrements de message d'insertion sont facultatifs ; si aucun enregistrement n'est fourni, *PutMsgRecOffset* est égal à zéro et *PutMsgRecFields* a la valeur MQPMRF_NONE.

La valeur initiale de cette zone est 0.

RecsPresent (MQLONG)

Il s'agit du nombre de destinations. Une liste de distribution doit toujours contenir au moins une destination. Par conséquent, *RecsPresent* doit toujours être supérieur à zéro.

La valeur initiale de cette zone est 0.

StrucId (MQCHAR4)

La valeur doit être:

ID_STRUC_MQDH_
Identificateur de la structure d'en-tête de distribution.

Pour le langage de programmation C, la constante MQDH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQDH_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQDH_STRUC_ID.

StrucLength (MQLONG)

Il s'agit du nombre d'octets entre le début de la structure MQDH et le début des données de message suivant les tableaux d'enregistrements MQOR et MQPMR. Les données se produisent dans l'ordre suivant:

- Structure MQDH
- Tableau d'enregistrements MQOR
- Tableau d'enregistrements MQPMR
- Données de message

Les tableaux des enregistrements MQOR et MQPMR sont traités par des décalages contenus dans la structure MQDH. Si ces décalages génèrent des octets inutilisés entre une ou plusieurs structures MQDH, les tableaux d'enregistrements et les données de message, ces octets inutilisés doivent être inclus dans la valeur de *StrucLength*, mais le contenu de ces octets n'est pas conservé par le gestionnaire de files d'attente. Le tableau des enregistrements MQPMR doit précéder le tableau des enregistrements MQOR.

La valeur initiale de cette zone est 0.

Version (MQLONG)

La valeur doit être:

MQDH_VERSION_1
Numéro de version de la structure d'en-tête de distribution.

La constante suivante indique le numéro de version de la version en cours:

MQDH_CURRENT_VERSION
Version actuelle de la structure d'en-tête de distribution.

La valeur initiale de cette zone est MQDH_VERSION_1.

Valeurs initiales et déclarations de langage pour MQDH

Tableau 494. Valeurs initiales des zones dans MQDH pour MQDH

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQDH_	'DH¬¬'
Version	MQDH_VERSION_1	1
StrucLength	Aucun	0
Encoding	Aucun	0
CodedCharSetId	MQCCSI_UNDEFINI	0
Format	MQFMT_AUCUN	Espaces vides
Flags	MQDHF_NONE	0
PutMsgRecFields	MQPMRF_AUCUN	0
RecsPresent	Aucun	0
ObjectRecOffset	Aucun	0
PutMsgRecOffset	Aucun	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQDH_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQDH MyDH = {MQDH_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration Visual Basic

```
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
```

En-tête MQDLH-Dead-letter

Le tableau suivant récapitule les zones de la structure.

Tableau 495. Zones dans MQDLH

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Reason	Message d'erreur arrivé dans la file d'attente de messages non livrés	Motif
DestQName	Nom de la file d'attente de réponses originale	DestQName
DestQMgrName	Nom du gestionnaire de files d'attente de destination original	DestQMgrName
Encoding	Codage numérique des données qui suit MQDLH	Codage
CodedCharSetId	Identificateur de jeu de caractères des données qui suit MQDLH	CodedCharSetId
Format	Nom du format des données qui suit MQDLH	Format
PutApplType	Type de l'application qui insère le message dans la file d'attente de messages non livrés	PutApplType
PutApplName	Nom de l'application qui insère le message dans la file d'attente de messages non livrés	PutApplName
PutDate	Date à laquelle le message a été inséré dans la file d'attente de messages non livrés	PutDate
PutTime	Heure à laquelle le message a été inséré dans la file d'attente de messages non livrés	PutTime

Présentation de MQDLH

Disponibilité: toutes les plateformes WebSphere MQ .

Objectif: La structure MQDLH décrit les informations qui préfixes les données de message d'application des messages de la file d'attente de rebut (message non distribué). Un message peut arriver dans la file d'attente de rebut soit parce que le gestionnaire de files d'attente ou l'agent de canal de message l'a redirigé vers la file d'attente, soit parce qu'une application a inséré le message directement dans la file d'attente.

Nom de format: MQFMT_DEAD_LETTER_HEADER.

Jeu de caractères et codage: les zones de la structure MQDLH sont dans le jeu de caractères et le codage donnés par les zones *CodedCharSetId* et *Encoding* . Ils sont spécifiés dans la structure

d'en-tête qui précède le MQDLH, ou dans la structure MQMD si le MQDLH se trouve au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

Si vous utilisez les classes WMQ pour Java / JMS et que la page de codes définie dans le MQMD n'est pas prise en charge par la machine virtuelle Java, le MQDLH est écrit dans le jeu de caractères UTF-8 .

Utilisation: Les applications qui placent des messages directement dans la file d'attente de rebut doivent préfixer les données de message avec une structure MQDLH et initialiser les zones avec les valeurs appropriées. Toutefois, le gestionnaire de files d'attente ne requiert pas la présence d'une structure MQDLH ou la spécification de valeurs valides pour les zones.

Si un message est trop long pour être inséré dans la file d'attente de rebut, l'application doit effectuer l'une des opérations suivantes:

- Tronque les données de message pour qu'elles tiennent dans la file d'attente des messages non livrés.
- Enregistrez le message dans la mémoire secondaire et placez un message de rapport d'exception dans la file d'attente des messages non livrés.
- Supprimez le message et renvoyez une erreur à son émetteur. Si le message est (ou peut être) un message critique, ne le faites que s'il est connu que l'émetteur possède toujours une copie du message ; par exemple, un message reçu par un agent de canal de transmission à partir d'un canal de communication.

Parmi ces éléments, celui qui convient (le cas échéant) dépend de la conception de l'application.

Le gestionnaire de files d'attente effectue un traitement spécial lorsqu'un message qui est un segment est inséré avec une structure MQDLH à l'avant ; voir la description de la structure MQMDE pour plus de détails.

Insertion de messages dans la file d'attente de rebut: lorsqu'un message est inséré dans la file d'attente de rebut, la structure MQMD utilisée pour l'appel MQPUT ou MQPUT1 doit être identique à celle associée au message (généralement le MQMD renvoyé par l'appel MQGET), à l'exception des éléments suivants:

- Définissez les zones *CodedCharSetId* et *Encoding* sur le jeu de caractères et le codage utilisés pour les zones de la structure MQDLH.
- Définissez la zone *Format* sur MQFMT_DEAD_LETTER_HEADER pour indiquer que les données commencent par une structure MQDLH.
- Définissez les zones de contexte (*AccountingToken*, *ApplIdentityData* , *ApplOriginData*, *PutApplName*, *PutApplType* , *PutDate*, *PutTime*, *UserIdentifier*) à l'aide d'une option de contexte adaptée aux circonstances:
  - Une application qui insère dans la file d'attente de rebut un message qui n'est lié à aucun message précédent doit utiliser l'option MQPMO_DEFAULT_CONTEXT ; le gestionnaire de files d'attente doit alors définir toutes les zones de contexte du descripteur de message sur leurs valeurs par défaut.
  - Une application serveur qui insère dans la file d'attente de rebut un message qu'elle vient de recevoir doit utiliser l'option MQPMO_PASS_ALL_CONTEXT pour conserver les informations de contexte d'origine.
  - Une application serveur plaçant dans la file d'attente des messages non livrés une *réponse* à un message qu'elle vient de recevoir doit utiliser l'option MQPMO_PASS_IDENTITY_CONTEXT ; cette option conserve les informations d'identité mais définit les informations d'origine comme étant celles de l'application serveur.
  - Un agent de canal de transmission de messages qui insère dans la file d'attente de rebut un message qu'il a reçu de son canal de communication doit utiliser l'option MQPMO_SET_ALL_CONTEXT pour conserver les informations de contexte d'origine.

Dans la structure MQDLH elle-même, définissez les zones comme suit:

- Définissez les zones *CodedCharSetId*, *Encoding*et *Format* sur les valeurs qui décrivent les données qui suivent la structure MQDLH, généralement les valeurs du descripteur de message d'origine.
- Définissez les zones de contexte *PutApplType*, *PutApplName*, *PutDate*et *PutTime* sur les valeurs appropriées à l'application qui place le message dans la file d'attente de rebut ; ces valeurs ne sont pas liées au message d'origine.
- Définissez d'autres zones selon les besoins.

Assurez-vous que toutes les zones ont des valeurs valides et que les zones de type caractère sont remplies avec des blancs à la longueur définie de la zone ; ne mettez pas fin prématurément aux données de type caractère en utilisant un caractère null, car le gestionnaire de files d'attente ne convertit pas les caractères null et suivants en blancs dans la structure MQDLH.

Obtention de messages de la file d'attente de rebut: les applications qui extraient des messages de la file d'attente de rebut doivent vérifier que les messages commencent par une structure MQDLH. L'application peut déterminer si une structure MQDLH est présente en examinant la zone *Format* dans le descripteur de message MQMD ; si la zone a la valeur MQFMT_DEAD_LETTER_HEADER, les données de message commencent par une structure MQDLH. Sachez également que les messages que les applications reçoivent de la file d'attente de rebut peuvent être tronqués s'ils étaient trop longs à l'origine pour la file d'attente.

Zones pour MQDLH

La structure MQDLH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

CodedCharSetId (MQLONG)
CodedCharSetId est l'identificateur de jeu de caractères des données qui transitent par la structure MQDLH (généralement les données du message d'origine) ; il ne s'applique pas aux données de type caractère dans la structure MQDLH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

MQCCSI_HÉRITER
Les données de type caractères des données qui suivent cette structure sont dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI_INHERIT n'est pas renvoyée par l'appel MQGET.

Vous ne pouvez pas utiliser MQCCSI_INHERIT si la valeur de la zone *PutApplType* dans MQMD est MQAT_BROKER.

Cette valeur est prise en charge dans les environnements suivants: AIX, HP-UX, z/OS, IBM i, Solaris, Linux, Windows, plus les clients WebSphere MQ MQI connectés à ces systèmes.

La valeur initiale de cette zone est MQCCSI_UNDEFINED.

DestQMgrNom (MQCHAR48)
DestQMgrNom du gestionnaire de files d'attente qui était la destination d'origine du message.

La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

DestQName (MQCHAR48)
DestQName est le nom de la file d'attente de messages qui était la destination d'origine du message.

La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

Codage (MQLONG)
Le codage est le codage numérique des données qui suivent la structure MQDLH (généralement les données du message d'origine) ; il ne s'applique pas aux données numériques de la structure MQDLH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

Format (MQCHAR8)
Le format est le nom de format des données qui suivent la structure MQDLH (généralement les données du message d'origine).

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de codage de la zone *Format* dans MQMD.

La longueur de cette zone est indiquée par MQ_FORMAT_LENGTH. La valeur initiale de cette zone est MQFMT_NONE.

PutApplNom (MQCHAR28)
PutApplNom de l'application qui a inséré le message dans la file d'attente des messages non livrés.

Le format du nom dépend de la zone *PutApplType* . Le format peut varier d'une édition à l'autre. Voir la description de la zone *PutApplName* dans «MQMD-Descripteur de message», à la page 394.

Si le gestionnaire de files d'attente redirige le message vers la file d'attente de rebut, *PutApplName* contient les 28 premiers caractères du nom du gestionnaire de files d'attente, complétés par des blancs si nécessaire.

La longueur de cette zone est indiquée par MQ_PUT_APPL_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 28 caractères blancs dans les autres langages de programmation.

Type PutAppl(MQLONG)
PutApplType d'application qui place le message dans la file d'attente des messages non livrés.

Cette zone a la même signification que la zone *PutApplType* dans le descripteur de message MQMD (voir «MQMD-Descripteur de message», à la page 394 pour plus de détails).

Si le gestionnaire de files d'attente redirige le message vers la file d'attente de rebut, *PutApplType* a la valeur MQAT_QMGR.

La valeur initiale de cette zone est 0.

PutDate (MQCHAR8)
PutDate est la date à laquelle le message a été inséré dans la file d'attente des messages non livrés.

Le format utilisé pour la date à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- AAAAMMJJ

où les caractères représentent:

AAAA
année (quatre chiffres)

MM
mois de l'année (01 à 12)

JJ
jour du mois (01 à 31)

Le temps moyen de Greenwich (GMT) est utilisé pour les zones *PutDate* et *PutTime* , à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

La longueur de cette zone est indiquée par MQ_PUT_DATE_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et huit caractères blancs dans les autres langages de programmation.

PutTime (MQCHAR8)
PutTime est l'heure à laquelle le message a été inséré dans la file d'attente des messages non livrés.

Le format utilisé pour l'heure à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- HHMMSSTH

où les caractères représentent:

hh
heures (00 à 23)

MM
minutes (00 à 59)

SS
secondes (00 à 59 ; voir remarque)

T
dixièmes de seconde (0 à 9)

H
centièmes de seconde (0 à 9)

Remarque : Si l'horloge système est synchronisée selon une norme de temps très précise, il est possible, dans de rares cas, que 60 ou 61 soient renvoyés pour les secondes dans *PutTime*. Cela se produit lorsque des secondes bissextiles sont insérées dans la norme de temps globale.

Le temps moyen de Greenwich (GMT) est utilisé pour les zones *PutDate* et *PutTime* , à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

La longueur de cette zone est indiquée par MQ_PUT_TIME_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et huit caractères blancs dans les autres langages de programmation.

Motif (MQLONG)
La zone Raison identifie la raison pour laquelle le message a été placé dans la file d'attente de rebut et non dans la file d'attente de destination d'origine.

Identifie la raison pour laquelle le message a été placé dans la file d'attente de rebut au lieu de la file d'attente de destination d'origine. Il doit s'agir de l'une des valeurs MQFB_ * ou MQRC_ * (par exemple, MQRC_Q_FULL). Voir la description de la zone *Feedback* dans «MQMD-Descripteur de message», à la page 394 pour plus de détails sur les valeurs MQFB_ * communes qui peuvent se produire.

Si la valeur est comprise entre MQFB_IMS_FIRST et MQFB_IMS_LAST, le code d'erreur IMS réel peut être déterminé en soustrayant MQFB_IMS_ERROR de la valeur de la zone *Reason* .

Certaines valeurs MQFB_ * apparaissent uniquement dans cette zone. Ils concernent des messages de référentiel, des messages de déclenchement ou des messages de file d'attente de transmission qui ont été transférés dans la file d'attente de rebut. Il s'agit des fonctions suivantes :

MQFB_APPL_CANNOT_BE_STARTED (X'00000109')
Une application traitant un message de déclenchement ne peut pas démarrer l'application nommée dans la zone *ApplId* du message de déclenchement (voir «MQTM-Message de déclenchement», à la page 581).

Sous z/OS, la transaction CKTI CICS est un exemple d'application qui traite les messages de déclenchement.

MQFB_APPL_TYPE_ERROR (X'0000010B')
Une application traitant un message de déclenchement ne peut pas démarrer l'application car la zone *ApplType* du message de déclenchement n'est pas valide (voir «MQTM-Message de déclenchement», à la page 581).

Sous z/OS, la transaction CKTI CICS est un exemple d'application qui traite les messages de déclenchement.

MQFB_BIND_OPEN_CLUSRCVR_DEL (X'00000119')
Le message se trouvait dans SYSTEM.CLUSTER.TRANSMIT.QUEUE destiné à une file d'attente de cluster qui a été ouverte avec l'option MQOO_BIND_ON_OPEN, mais le canal récepteur de cluster distant à utiliser pour transmettre le message à la file d'attente de destination a été supprimé avant que le message ne puisse être envoyé. MQOO_BIND_ON_OPEN ayant été indiqué, seul le canal sélectionné lors de l'ouverture de la file d'attente peut être utilisé pour transmettre le message. Ce canal n'étant plus disponible, le message est placé dans la file d'attente des messages non livrés.

MQFB_NOT_A_REPOSITORY_MSG (X'00000118')
Le message n'est pas un message de référentiel.

MQFB_STOPPED_BY_CHAD_EXIT (X'00000115')
Le message a été arrêté par l'exit de définition automatique de canal.

MQFB_STOPPED_BY_MSG_EXIT (X'0000010D')
Le message a été arrêté par l'exit de message de canal.

MQFB_TM_ERROR (X'0000010A')
La zone *Format* de MQMD indique MQFMT_TRIGGER, mais le message ne commence pas par une structure MQTM valide. Par exemple, il se peut que l'identificateur mnémonique *StrucId* ne soit pas valide, que *Version* ne soit pas reconnu ou que la longueur du message de déclenchement ne soit pas suffisante pour contenir la structure MQTM.

Sous z/OS, la transaction CKTI CICS est un exemple d'application qui traite les messages de déclenchement et peut générer ce code retour.

MQFB_XMIT_Q_MSG_ERROR (X'0000010F')
Un agent MCA a détecté qu'un message de la file d'attente de transmission n'est pas au format correct. L'agent MCA place le message dans la file d'attente de rebut à l'aide de ce code retour.

La valeur initiale de cette zone est MQRC_NONE.

StrucId (MQCHAR4)
StrucId est l'identificateur de structure.

La valeur doit être:

ID_STRUC_MQDLH
Identificateur de la structure d'en-tête de rebut.

Pour le langage de programmation C, la constante MQDLH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQDLH_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

La valeur initiale de cette zone est MQDLH_STRUC_ID.

Version (MQLONG)
La version est le numéro de version de la structure.

La valeur doit être:

MQDLH_VERSION_1
Numéro de version de la structure d'en-tête de rebut.

La constante suivante indique le numéro de version de la version en cours:

MQDLH_CURRENT_VERSION
Version actuelle de la structure d'en-tête de rebut.

La valeur initiale de cette zone est MQDLH_VERSION_1.

Valeurs initiales et déclarations de langue pour MQDLH

Tableau 496. Valeurs initiales des zones dans MQDLH pour MQDLH

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQDLH	'DLH¬'
Version	MQDLH_VERSION_1	1
Reason	MQRC_NONE	0
DestQName	Aucun	Chaîne nulle ou blancs
DestQMgrName	Aucun	Chaîne nulle ou blancs
Encoding	Aucun	0
CodedCharSetId	MQCCSI_UNDEFINI	0
Format	MQFMT_AUCUN	Espaces vides
PutApplType	Aucun	0
PutApplName	Aucun	Chaîne nulle ou blancs
PutDate	Aucun	Chaîne nulle ou blancs
PutTime	Aucun	Chaîne nulle ou blancs

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQDLH_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQDLH MyDLH = {MQDLH_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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).
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Options de suppression de descripteur de message

Le tableau suivant récapitule les zones de la structure.

Tableau 497. Zones dans MQDMHO		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options	Options

Présentation de MQDMHO

Disponibilité: Tous les systèmes WebSphere MQ et les clients WebSphere MQ .

Objectif: La structure **MQDMHO** permet aux applications de spécifier des options qui contrôlent la façon dont les messages sont supprimés. La structure est un paramètre d'entrée dans l'appel **MQDLTMH** .

Jeu de caractères et codage: les données dans **MQDMHO** doivent être dans le jeu de caractères de l'application et le codage de l'application (**MQENC_NATIVE**).

Zones pour MQDMHO

La structure MQDMHO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

Options (MQLONG)

La valeur doit être:

MQDMHO_AUCUN

Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQDMHO_NONE**.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

MQDMHO_STRUC_ID

Identificateur de la structure des options de descripteur de message de suppression.

Pour le langage de programmation C, la constante **MQDMHO_STRUC_ID_ARRAY** est également définie ; elle a la même valeur que **MQDMHO_STRUC_ID**, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQDMHO_STRUC_ID**.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQDMHO_VERSION_1
Version-1 : supprime la structure d'options de descripteur de message.

La constante suivante indique le numéro de version de la version en cours:

MQDMHO_CURRENT_VERSION
Version actuelle de la structure d'options de suppression de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQDMHO_VERSION_1**.

Valeurs initiales et déclarations de langue pour MQDMHO

Tableau 498. Valeurs initiales des zones dans MQDMHO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	MQDMHO_STRUC_ID	'DMHO'
Version	MQDMHO_VERSION_1	1
Options	MQDMHO_AUCUN	0

Tableau 498. Valeurs initiales des zones dans MQDMHO (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**

Remarques :

1. Dans le langage de programmation C, la variable macroMQDMHO_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQDMHO MyDMHO = {MQDMHO_DEFAULT};
```

Déclaration C

```
typedef struct tagMQDMHO;
struct tagMQDMHO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQDLTMH */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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-Options de suppression de propriété de message

Le tableau suivant récapitule les zones de la structure. Structure MQDMPO-options de suppression de propriété de message

Tableau 499. Zones dans MQDMPO		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId

Tableau 499. Zones dans MQDMPO (suite)		
Zone	**Description**	**Topic**
Version	Numéro de version de structure	Version
Options	Options contrôlant l'action de MQDMPO	Options

Présentation de MQDMPO

Disponibilité: Tous les systèmes WebSphere MQ et les clients WebSphere MQ .

Objectif: La structure MQDMPO permet aux applications de spécifier des options qui contrôlent la façon dont les propriétés des messages sont supprimées. La structure est un paramètre d'entrée dans l'appel MQDLTMP.

Jeu de caractères et codage: les données de MQDMPO doivent être dans le jeu de caractères de l'application et le codage de l'application (MQENC_NATIVE).

Zones pour MQDMPO

Supprimer la structure des options de propriété de message-zones

La structure MQDMPO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

Options (MQLONG)
Supprimer la structure des options de propriété de message-Zone Options

Options d'emplacement: Les options suivantes sont liées à l'emplacement relatif de la propriété par rapport au curseur de propriété.

MQDMPO_DEL_FIRST
Supprime la première propriété qui correspond au nom spécifié.

MQDMPO_DEL_PROP_UNDER_CURSOR
Supprime la propriété pointée par le curseur de propriété ; il s'agit de la propriété qui a été interrogée pour la dernière fois à l'aide de l'option MQIMPO_INQ_FIRST ou MQIMPO_INQ_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé. Il est également réinitialisé lorsque le descripteur de message est spécifié dans la zone *MsgHandle* de la structure MQGMO sur un appel MQGET ou MQPMO sur un appel MQPUT.

Si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi, l'appel échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_PROPERTY_NOT_AVAILABLE. Si la propriété pointée par le curseur de propriété a déjà été supprimée, l'appel échoue également avec le code achèvement MQCC_FAILED et la raison MQRC_PROPERTY_NOT_AVAILABLE.

Si aucune des options n'est requise, l'option suivante peut être utilisée:

MQDMPO_AUCUN
Aucune option spécifiée.

Cette zone est toujours une zone d'entrée. La valeur initiale de cette zone est MQDMPO_DEL_FIRST.

StrucId (MQCHAR4)
Supprimer la structure des options de propriété de message-Zone StrucId

Il s'agit de l'identificateur de structure. La valeur doit être:

ID_STRUC_MQDMPO_
Identificateur de la structure des options de suppression des propriétés de message.

Pour le langage de programmation C, la constante MQDMPO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQDMPO_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQDMPO_STRUC_ID.

Version (MQLONG)
Supprimer la structure des options de propriété de message-Zone Version

Il s'agit du numéro de version de la structure. La valeur doit être:

MQDMPO_VERSION_1
Numéro de version de la structure des options de suppression de propriété de message.

La constante suivante indique le numéro de version de la version en cours:

MQDMPO_VERSION
Version actuelle de la structure des options de suppression des propriétés de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQDMPO_VERSION_1.

Valeurs initiales et déclarations de langage pour MQDMPO
Supprimer la structure des options de propriété de message-Valeurs initiales

Tableau 500. Valeurs initiales des zones dans MQDPMO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQDMPO_	'DMPO'
Version	MQDMPO_VERSION_1	1
Options	Options qui contrôlent l'action de MQDLTMP	MQDMPO_AUCUN

Remarques :

1. Dans le langage de programmation C, la variable macroMQDMPO_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQDMPO MyDMPO = {MQDMPO_DEFAULT};
```

Déclaration C
Supprimer la structure des options de propriété de message-Déclaration de langage C

```
typedef struct tagMQDMPO MQDMPO;
struct tagMQDMPO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQDLTMP */
};
```

Déclaration COBOL
Supprimer la structure des options de propriété de message-Déclaration en langage COBOL

```
**   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.
```

Déclaration PL/I
Supprimer la structure des options de propriété de message-Déclaration de langage PL/I

```
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 */
```

Déclaration High Level Assembler
Supprimer la structure des options de propriété de message-Déclaration du langage assembleur

```
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-en-tête PCF imbriqué

Le tableau suivant récapitule les zones de la structure.

Tableau 501. Zones dans MQEPH

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
StrucLength	Longueur de la structure MQEPH plus les structures de paramètres et MQCFH qui suivent	StrucLength
Encoding	Codage numérique des données qui suit la structure de paramètres PCF	Codage
CodedCharSetId	Identificateur de jeu de caractères des données qui suit la structure de paramètres PCF	CodedCharSetId
Format	Nom du format des données qui suit la structure de paramètres PCF	Format
Flags	Indicateurs	Indicateurs
PCFHeader	En-tête de format de commande programmable (PCF)	PCFHeader

Présentation de MQEPH

Disponibilité: toutes les plateformes WebSphere MQ .

Objectif: La structure MQEPH décrit les données supplémentaires présentes dans un message lorsqu'il s'agit d'un message PCF (Programmable Command Format). La zone *PCFHeader* définit les paramètres PCF qui suivent cette structure et permet de suivre les données de message PCF avec d'autres en-têtes.

Nom de format: MQFMT_EMBEDDED_PCF

Jeu de caractères et codage: les données de MQEPH doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE.

Définissez le jeu de caractères et le codage du MQEPH dans les zones *CodedCharSetId* et *Encoding* dans:

- Le MQMD (si la structure MQEPH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQEPH (tous les autres cas).

Syntaxe: vous ne pouvez pas utiliser de structures MQEPH pour envoyer des commandes au serveur de commandes ou à tout autre serveur acceptant les PCF du gestionnaire de files d'attente.

De même, le serveur de commandes ou tout autre serveur d'acceptation PCF du gestionnaire de files d'attente ne génère pas de réponses ou d'événements contenant des structures MQEPH.

Zones pour MQEPH

La structure MQEPH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

CodedCharSetId (MQLONG)

Il s'agit de l'identificateur de jeu de caractères des données qui suivent la structure MQEPH et les paramètres PCF associés ; il ne s'applique pas aux données de type caractère de la structure MQEPH elle-même.

La valeur initiale de cette zone est MQCCSI_UNDEFINED.

Codage (MQLONG)

Il s'agit du codage numérique des données qui suit la structure MQEPH et les paramètres PCF associés ; il ne s'applique pas aux données de type caractères de la structure MQEPH elle-même.

La valeur initiale de cette zone est 0.

Indicateurs (MQLONG)

Les valeurs suivantes sont disponibles :

MQEPH_NONE
Aucun indicateur n'a été spécifié. MQEPH_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

MQEPH_CCSID_IMBRIQUÉ
Le jeu de caractères des paramètres contenant des données de type caractères est spécifié individuellement dans la zone CodedCharSetId de chaque structure. Le jeu de caractères des zones StrucId et Format est défini par la zone CodedCharSetId dans la structure d'en-tête qui précède la structure MQEPH, ou par la zone CodedCharSetId dans le MQMD si MQEPH se trouve au début du message.

La valeur initiale de cette zone est MQEPH_NONE.

Format (MQCHAR8)

Il s'agit du nom de format des données qui suivent la structure MQEPH et les paramètres PCF associés.

La valeur initiale de cette zone est MQFMT_NONE.

En-tête de groupe de correctifs (MQCFH)

Il s'agit de l'en-tête PCF (Programmable Command Format) qui définit les paramètres PCF qui suivent la structure MQEPH. Cela vous permet de suivre les données de message PCF avec d'autres en-têtes.

L'en-tête PCF est initialement défini avec les valeurs suivantes:

Tableau 502. Valeurs initiales des zones dans MQCFH

Nom de zone	Nom de la constante	Valeur de la constante
Type	MQCFT_AUCUN	0
StrucLength	MQCFH_LONGUEUR_STRUCTURE	36
Version	MQCFH_VERSION_3	3
StrucLength	Aucun	0

Tableau 502. Valeurs initiales des zones dans MQCFH (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
Command	MQCMD_NONE	0
MsgSeqNumber	Aucun	1
Control	MQCFC_LAST	1
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
ParameterCount	Aucun	0

L'application doit remplacer Type de MQCFT_NONE par un type de structure valide pour l'utilisation de l'en-tête PCF imbriqué.

StrucId (MQCHAR4)

La valeur doit être:

ID_STRUC_MQEPH_ID
Identificateur de la structure d'en-tête de distribution.

Pour le langage de programmation C, la constante MQEPH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQDH_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQEPH_STRUC_ID.

StrucLength (MQLONG)

Il s'agit de la quantité de données précédant la structure d'en-tête suivante. Cela comprend :

- Longueur de l'en-tête MQEPH
- Longueur de tous les paramètres PCF suivant l'en-tête
- Tout remplissage vide suivant ces paramètres

StrucLength doit être un multiple de 4.

La partie de longueur fixe de la structure est définie par MQEPH_STRUC_LENGTH_FIXED.

La valeur initiale de cette zone est 68.

Version (MQLONG)

La valeur doit être:

MQEPH_VERSION_1
Numéro de version de la structure d'en-tête PCF intégrée.

La constante suivante indique le numéro de version de la version en cours:

MQCFH_VERSION_3
Version actuelle de la structure d'en-tête PCF intégrée.

La valeur initiale de cette zone est MQEPH_VERSION_1.

Valeurs initiales et déclarations de langue pour MQEPH

Tableau 503. Valeurs initiales des zones dans MQEPH pour MQEPH		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
StrucId	ID_STRUC_MQEPH_ID	'EPH¬'

Tableau 503. Valeurs initiales des zones dans MQEPH pour MQEPH (suite)

Nom de zone	Nom de la constante	Valeur de la constante
Version	MQEPH_VERSION_1	1
StrucLength	MQEPH_STRUC_LENGTH_FIXED	68
Encoding	Aucun	0
CodedCharSetId	MQCCSI_UNDEFINI	0
Format	MQFMT_AUCUN	Espaces vides
Flags	MQEPH_NONE	0
PCFHeader	Noms et valeurs tels que définis dans Tableau 502, à la page 341	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQEPH_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQEPH MyEPH = {MQEPH_DEFAULT};
```

Déclaration C

```
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 */
 };
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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
```

Options MQGMO-Get-message

Le tableau suivant récapitule les zones de la structure.

Tableau 504. Zones dans MQGMO

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options qui contrôlent l'action de MQGET	Zone MQGMO-Options
WaitInterval	Intervalle d'attente	WaitInterval
Signal1	Signal	Signal1
Signal2	Identificateur de signal	Signal2
ResolvedQName	Nom résolu de la file d'attente de destination	ResolvedQName
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQGMO_VERSION_2.		
MatchOptions	Options de contrôle des critères de sélection utilisés pour MQGET	MatchOptions
GroupStatus	Indicateur signalant si le message extrait fait partie d'un groupe	GroupStatus
SegmentStatus	Indicateur signalant si le message extrait est un segment d'un message logique	SegmentStatus
Segmentation	Indicateur signalant si une segmentation supplémentaire est autorisée pour le message extrait	Segmentation
Reserved1	Réservé	Reserved1
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQGMO_VERSION_3.		
MsgToken	Jeton de message	MsgToken

Tableau 504. Zones dans MQGMO (suite)		
Zone	**Description**	**Topic**
ReturnedLength	Longueur des données de message retournées (octets)	ReturnedLength
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQGMO_VERSION_4.		
Reserved2	Réservé	Reserved2
MsgHandle	Descripteur d'un message à remplir avec les propriétés du message extrait de la file d'attente.	MsgHandle

Présentation de MQGMO

Disponibilité: toutes les plateformes WebSphere MQ .

Objectif: La structure MQGMO permet à l'application de contrôler la façon dont les messages sont supprimés des files d'attente. La structure est un paramètre d'entrée-sortie dans l'appel MQGET.

Version: la version actuelle de MQGMO est MQGMO_VERSION_4. Certaines zones ne sont disponibles que dans certaines versions de MQGMO. Si vous devez porter des applications entre plusieurs environnements, vous devez vous assurer que la version de MQGMO est cohérente dans tous les environnements. Les zones qui existent uniquement dans des versions particulières de la structure sont identifiées comme telles dans «Options MQGMO-Get-message», à la page 345 et dans les descriptions de zone.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQGMO prise en charge par l'environnement, mais avec la valeur initiale de la zone *Version* définie sur MQGMO_VERSION_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , définissez la zone *Version* sur le numéro de version de la version requise.

Jeu de caractères et codage: les données de MQGMO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Zones pour MQGMO

La structure MQGMO contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

GroupStatus (MQCHAR)
Cet indicateur indique si le message extrait se trouve dans un groupe.

Elle a l'une des valeurs suivantes :

MQGS_NOT_IN_GROUPE
Le message n'est pas dans un groupe.

MQGS_MSG_IN_GROUPE
Le message se trouve dans un groupe, mais n'est pas le dernier du groupe.

MQGS_LAST_MSG_IN_GROUP
Le message est le dernier du groupe.

Il s'agit également de la valeur renvoyée si le groupe se compose d'un seul message.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est MQGS_NOT_IN_GROUP. Cette zone est ignorée si *Version* est inférieur à MQGMO_VERSION_2.

MatchOptions (MQLONG)

Ces options permettent à l'application de choisir les zones du paramètre *MsgDesc* à utiliser pour sélectionner le message renvoyé par l'appel MQGET. L'application définit les options requises dans cette zone, puis définit les zones correspondantes dans le paramètre *MsgDesc* sur les valeurs requises pour ces zones. Seuls les messages ayant ces valeurs dans le MQMD pour le message sont candidats à l'extraction à l'aide de ce paramètre *MsgDesc* dans l'appel MQGET. Les zones pour lesquelles l'option de correspondance correspondante n'est *pas* spécifiée sont ignorées lors de la sélection du message à renvoyer. Si vous n'indiquez aucun critère de sélection dans l'appel MQGET (c'est-à-dire *tout message* est acceptable), définissez *MatchOptions* sur MQMO_NONE.

- Sous z/OS, les critères de sélection pouvant être utilisés peuvent être limités par le type d'index utilisé pour la file d'attente. Pour plus de détails, voir l'attribut de file d'attente *IndexType* .

Si vous spécifiez MQGMO_LOGICAL_ORDER, seuls certains messages peuvent être renvoyés par l'appel MQGET suivant:

- S'il n'y a pas de groupe ou de message logique en cours, seuls les messages dont la valeur est *MsgSeqNumber* égale à 1 et *Offset* égale à 0 peuvent être renvoyés. Dans ce cas, vous pouvez utiliser une ou plusieurs des options de correspondance suivantes pour sélectionner les messages éligibles renvoyés:
  - ID MQMO_MATCH_MSG_ID
  - ID_CORREL_MQMO_MATCH_
  - ID_GROUPE_MQMO_MATCH
- S'il *existe* un groupe ou un message logique en cours, seul le message suivant du groupe ou du segment suivant du message logique peut être renvoyé, ce qui ne peut pas être modifié en spécifiant les options MQMO_ *.

Dans les deux cas ci-dessus, vous pouvez spécifier des options de correspondance qui ne s'appliquent pas, mais la valeur de la zone appropriée dans le paramètre *MsgDesc* doit correspondre à la valeur de la zone correspondante dans le message à renvoyer. L'appel échoue avec le code anomalie MQRC_MATCH_OPTIONS_ERROR si cette condition n'est pas satisfaite.

MatchOptions est ignoré si vous spécifiez MQGMO_MSG_UNDER_CURSOR ou MQGMO_BROWSE_MSG_UNDER_CURSOR.

L'obtention de messages basés sur une propriété de message n'est pas effectuée à l'aide des options de correspondance ; pour plus d'informations, voir «SelectionString (MQCHARV)», à la page 466 .

Vous pouvez spécifier une ou plusieurs des options de correspondance suivantes:

ID MQMO_MATCH_MSG_ID
Le message à extraire doit avoir un identificateur de message qui correspond à la valeur de la zone *MsgId* dans le paramètre *MsgDesc* de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de corrélation).

Si vous omettez cette option, la zone *MsgId* du paramètre *MsgDesc* est ignorée et tout identificateur de message correspond.

Remarque : L'identificateur de message MQMI_NONE est une valeur spéciale qui correspond à *n'importe quel* identificateur de message dans le MQMD du message. Par conséquent, la spécification de MQMO_MATCH_MSG_ID avec MQMI_NONE est identique à la *non* spécification de MQMO_MATCH_MSG_ID.

ID_CORREL_MQMO_MATCH_
Le message à extraire doit avoir un identificateur de corrélation qui correspond à la valeur de la zone *CorrelId* dans le paramètre *MsgDesc* de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de message).

Si vous omettez cette option, la zone *CorrelId* du paramètre *MsgDesc* est ignorée et tout identificateur de corrélation correspond.

Remarque : L'identificateur de corrélation MQCI_NONE est une valeur spéciale qui correspond à *tout* identificateur de corrélation dans le MQMD du message. Par conséquent, la spécification

de MQMO_MATCH_CORREL_ID avec MQCI_NONE est identique à la *non* spécification de MQMO_MATCH_CORREL_ID.

ID_GROUPE_MQMO_MATCH
Le message à extraire doit avoir un identificateur de groupe qui correspond à la valeur de la zone *GroupId* dans le paramètre *MsgDesc* de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de corrélation).

Si vous omettez cette option, la zone *GroupId* du paramètre *MsgDesc* est ignorée et tout identificateur de groupe correspond.

Remarque : L'identificateur de groupe MQGI_NONE est une valeur spéciale qui correspond à *n'importe quel* identificateur de groupe dans le MQMD pour le message. Par conséquent, la spécification de MQMO_MATCH_GROUP_ID avec MQGI_NONE est identique à la *non* spécification de MQMO_MATCH_GROUP_ID.

NUMERO MQMO_MATCH_MSG_SEQ_NO
Le message à extraire doit avoir un numéro de séquence de message qui correspond à la valeur de la zone *MsgSeqNumber* dans le paramètre *MsgDesc* de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, l'identificateur de groupe).

Si vous omettez cette option, la zone *MsgSeqNumber* du paramètre *MsgDesc* est ignorée et tout numéro de séquence de message correspond.

MQMO_MATCH_OFFSET
Le message à extraire doit avoir un décalage qui correspond à la valeur de la zone *Offset* dans le paramètre *MsgDesc* de l'appel MQGET. Cette correspondance s'ajoute à toute autre correspondance pouvant s'appliquer (par exemple, le numéro de séquence du message).

Si vous omettez cette option, la zone *Offset* du paramètre *MsgDesc* est ignorée et tout décalage correspond.

- Cette option n'est pas prise en charge sous z/OS.

MQMO_MATCH_MSG_TOKEN
Le message à extraire doit avoir un jeton de message qui correspond à la valeur de la zone *MsgToken* dans la structure MQGMO définie dans l'appel MQGET.

Vous pouvez spécifier cette option pour toutes les files d'attente locales. Si vous le spécifiez pour une file d'attente dont le *IndexType* est MQIT_MSG_TOKEN (file d'attente gérée par WLM), vous ne pouvez spécifier aucune autre option de correspondance avec MQMO_MATCH_MSG_TOKEN.

Vous ne pouvez pas spécifier MQMO_MATCH_MSG_TOKEN avec MQGMO_WAIT ou MQGMO_SET_SIGNAL. Si l'application souhaite attendre qu'un message arrive dans une file d'attente dont le *IndexType* est MQIT_MSG_TOKEN, indiquez MQMO_NONE.

Si vous omettez cette option, la zone *MsgToken* de MQGMO est ignorée et tout jeton de message correspond.

Si vous n'indiquez aucune des options décrites, vous pouvez utiliser l'option suivante:

MQMO_AUCUN
N'utilisez aucune correspondance pour sélectionner le message à renvoyer ; tous les messages de la file d'attente peuvent être extraits (mais soumis au contrôle des options MQGMO_ALL_MSGS_AVAILABLE, MQGMO_ALL_SEGMENTS_AVAILABLE et MQGMO_COMPLETE_MSG).

MQMO_NONE facilite la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option MQMO_ *, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQMO_MATCH_MSG_ID avec MQMO_MATCH_CORREL_ID. Cette zone est ignorée si *Version* est inférieur à MQGMO_VERSION_2.

Remarque : La valeur initiale de la zone *MatchOptions* est définie à des fins de compatibilité avec les gestionnaires de files d'attente MQSeries antérieurs. Toutefois, lors de la lecture d'une série

de messages à partir d'une file d'attente sans utiliser de critères de sélection, cette valeur initiale nécessite que l'application réinitialise les zones *MsgId* et *CorrelId* sur MQMI_NONE et MQCI_NONE avant chaque appel MQGET. Evitez de réinitialiser *MsgId* et *CorrelId* en définissant *Version* sur MQGMO_VERSION_2et *MatchOptions* sur MQMO_NONE.

MsgHandle (MQHMSG)

Si l'option MQGMO_PROPERTIES_AS_Q_DEF est spécifiée et que l'attribut de file d'attente PropertyControl n'est pas défini sur MQPROP_FORCE_MQRFH2 , il s'agit du descripteur d'un message qui sera rempli avec les propriétés du message extrait de la file d'attente. Le descripteur est créé par un appel MQCRTMH. Toutes les propriétés déjà associées à l'identificateur seront effacées avant l'extraction d'un message.

La valeur suivante peut également être spécifiée:
MQHM_NONE
Aucun descripteur de message fourni.

Aucun descripteur de message n'est requis dans l'appel MQGET si un descripteur de message valide est fourni et utilisé dans la sortie pour contenir les propriétés de message, le descripteur de message associé au descripteur de message est utilisé pour les zones d'entrée.

Si un descripteur de message est spécifié dans l'appel MQGET, il est toujours prioritaire sur le descripteur de message associé à un descripteur de message.

Si MQGMO_PROPERTIES_FORCE_MQRFH2 est spécifié ou que MQGMO_PROPERTIES_AS_Q_DEF est spécifié et que l'attribut de file d'attente PropertyControl est MQPROP_FORCE_MQRFH2 , l'appel échoue avec le code anomalie MQRC_MD_ERROR lorsqu'aucun paramètre de descripteur de message n'est spécifié.

Lors du retour de l'appel MQGET, les propriétés et le descripteur de message associés à cet identificateur de message sont mis à jour pour refléter l'état du message extrait (ainsi que le descripteur de message s'il a été fourni sur l'appel MQGET). Les propriétés du message peuvent ensuite être recherchées à l'aide de l'appel MQINQMP.

A l'exception des extensions de descripteur de message, lorsqu'elles sont présentes, une propriété qui peut être interrogée avec l'appel MQINQMP n'est pas contenue dans les données de message ; si le message dans la file d'attente contient des propriétés dans les données de message, elles sont supprimées des données de message avant que les données ne soient renvoyées à l'application.

Si aucun descripteur de message n'est fourni ou si la version est inférieure à MQGMO_VERSION_4 , vous devez fournir un descripteur de message valide dans l'appel MQGET. Toutes les propriétés de message (à l'exception de celles contenues dans le descripteur de message) sont renvoyées dans le sujet de données de message avec la valeur des options de propriété dans la structure MQGMO et l'attribut de file d'attente PropertyControl .

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQHM_NONE. Cette zone est ignorée si Version est inférieur à MQGMO_VERSION_4.

MsgToken (MQBYTE16)
Zone MsgToken -Structure MQGMO. Cette zone est utilisée par le gestionnaire de files d'attente pour identifier un message de manière unique.

Il s'agit d'une chaîne d'octets générée par le gestionnaire de files d'attente pour identifier un message de manière unique dans une file d'attente. Le jeton de message est généré lorsque le message est placé pour la première fois sur le gestionnaire de files d'attente et reste avec le message jusqu'à ce qu'il soit définitivement supprimé du gestionnaire de files d'attente, sauf si le gestionnaire de files d'attente est redémarré.

Lorsque le message est supprimé de la file d'attente, le *MsgToken* qui a identifié cette instance du message n'est plus valide et n'est jamais réutilisé. Si le gestionnaire de files d'attente est redémarré, le *MsgToken* qui a identifié un message dans la file d'attente avant le redémarrage peut ne pas être valide après le redémarrage. Toutefois, *MsgToken* n'est jamais réutilisé pour identifier une autre instance

de message. *MsgToken* est généré par le gestionnaire de files d'attente et n'est visible par aucune application externe.

Lorsqu'un message est renvoyé par un appel à MQGET dans lequel un MQGMO version 3 ou ultérieure est fourni, le *MsgToken* identifiant le message dans la file d'attente est renvoyé dans le MQGMO par le gestionnaire de files d'attente. Il existe une exception: lorsque le message est supprimé de la file d'attente en dehors du point de synchronisation, le gestionnaire de files d'attente peut ne pas renvoyer de *MsgToken* car il n'est pas utile d'identifier le message renvoyé lors d'un appel MQGET ultérieur. Les applications doivent uniquement utiliser *MsgToken* pour faire référence au message lors des appels MQGET suivants.

Si un *MsgToken* est fourni et que *MatchOption* MQMO_MATCH_MSG_TOKEN est spécifié et que ni MQGMO_MSG_UNDER_CURSOR ni MQGMO_BROWSE_MSG_UNDER_CURSOR ne sont spécifiés, seul le message identifié par ce *MsgToken* peut être renvoyé. L'option est valide sur toutes les files d'attente locales, indépendamment de INDXTYPE, et sur z/OS , vous devez utiliser INDXTYPE (MSGTOKEN) uniquement sur les files d'attente Workload Manager (WLM).

Tous les autres *MatchOptions* spécifiés sont vérifiés et, s'ils ne correspondent pas, MQRC_NO_MSG_AVAILABLE est renvoyé. Si MQGMO_BROWSE_NEXT est codé avec MQMO_MATCH_MSG_TOKEN, le message identifié par *MsgToken* est renvoyé uniquement s'il dépasse le curseur de navigation pour le descripteur appelant.

Si MQGMO_MSG_UNDER_CURSOR ou MQGMO_BROWSE_MSG_UNDER_CURSOR est spécifié, MQMO_MATCH_MSG_TOKEN est ignoré.

MQMO_MATCH_MSG_TOKEN n'est pas valide avec les options d'obtention de message suivantes:

- MQGMO_WAIT
- MQGMO_SET_SIGNAL

Pour un appel MQGET spécifiant MQMO_MATCH_MSG_TOKEN, un MQGMO de version 3 ou ultérieure doit être fourni à l'appel, sinon MQRC_WRONG_GMO_VERSION est renvoyé.

Si *MsgToken* n'est pas valide pour le moment, MQCC_FAILED avec MQRC_NO_MSG_AVAILABLE est renvoyé, sauf s'il existe une autre erreur.

Options (MQLONG)

Les options **MQGMO** contrôlent l'action de MQGET. Vous pouvez spécifier zéro ou plusieurs options. Si vous avez besoin de plusieurs valeurs facultatives:

- Ajoutez les valeurs (n'ajoutez pas la même constante plus d'une fois), ou
- Combinez les valeurs à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit).

Les combinaisons d'options non valides sont notées ; toutes les autres combinaisons sont valides.

Options d'attente: Les options suivantes concernent l'attente de l'arrivée de messages dans la file d'attente:

MQGMO_WAIT
L'application attend l'arrivée d'un message approprié. La durée maximale d'attente de l'application est spécifiée dans *WaitInterval*.

Important : Il n'y a pas d'attente, ni de délai, si un message approprié est disponible immédiatement.

Si les demandes MQGET sont inhibées ou que les demandes MQGET sont inhibées pendant l'attente, l'attente est annulée. L'appel se termine avec MQCC_FAILED et le code anomalie MQRC_GET_INHIBITED, qu'il y ait ou non des messages appropriés dans la file d'attente.

Vous pouvez utiliser MQGMO_WAIT avec les options MQGMO_BROWSE_FIRST ou MQGMO_BROWSE_NEXT .

Si plusieurs applications sont en attente dans la même file d'attente partagée, les règles suivantes sélectionnent l'application qui est activée lorsqu'un message approprié arrive:

Tableau 505. Règles d'activation des appels MQGET dans une file d'attente partagée.		
Nombre d'appels MQGET en attente d'activation		**Résultat**
Avec une option BROWSE	**Sans option BROWSE**[1]	
Aucun	Au moins un	Un appel MQGET sans option BROWSE est activé.
Au moins un	Aucun	Tous les appels MQGET avec une option BROWSE sont activés.
Au moins un	Au moins un	Un appel MQGET sans option BROWSE est activé. Le nombre d'appels MQGET avec une option BROWSE qui sont activés est imprévisible.

Si plusieurs appels MQGET sans option BROWSE sont en attente dans la même file d'attente, un seul appel est activé. Le gestionnaire de files d'attente tente d'accorder la priorité aux appels en attente dans l'ordre suivant:

1. Demandes get-wait spécifiques qui ne peuvent être satisfaites que par certains messages, par exemple, ceux avec un *MsgId* ou un *CorrelId* spécifique (ou les deux).
2. Demandes get-wait générales qui peuvent être satisfaites par n'importe quel message.

Remarque :

- Dans la première catégorie, aucune priorité supplémentaire n'est accordée aux demandes get-wait plus spécifiques. Par exemple, les demandes qui spécifient à la fois *MsgId* et *CorrelId*.
- Dans l'une ou l'autre catégorie, il n'est pas possible de prédire quelle application est sélectionnée. En particulier, l'application en attente la plus longue n'est pas nécessairement celle sélectionnée.
- La longueur du chemin et les considérations relatives à la planification de la priorité du système d'exploitation peuvent signifier qu'une application en attente dont la priorité du système d'exploitation est inférieure à celle attendue extrait le message.
- Il peut également arriver qu'une application qui n'est pas en attente récupère le message de préférence à un autre.

Sous z/OS, les points suivants s'appliquent:

- Si vous souhaitez que l'application procède à d'autres travaux en attendant l'arrivée du message, envisagez d'utiliser l'option de signal (MQGMO_SET_SIGNAL) à la place. Toutefois, l'option de signal est spécifique à l'environnement ; les applications que vous devez porter entre différents environnements ne doivent pas l'utiliser.
- S'il y a plusieurs appels MQGET en attente du même message, avec un mélange d'options d'attente et de signal, chaque appel en attente est considéré de la même manière. Une erreur s'est produite lors de la spécification de MQGMO_SET_SIGNAL avec MQGMO_WAIT. Il est également erroné de spécifier cette option avec un descripteur de file d'attente pour lequel un signal est en attente.
- Si vous spécifiez MQGMO_WAIT ou MQGMO_SET_SIGNAL pour une file d'attente dont le *IndexType* est MQIT_MSG_TOKEN, aucun critère de sélection n'est autorisé. Ce qui signifie que :
  - Si vous utilisez un version-1 MQGMO, définissez les zones *MsgId* et *CorrelId* dans le MQMD spécifié dans l'appel MQGET à MQMI_NONE et MQCI_NONE.
  - Si vous utilisez une version version-2 ou ultérieure MQGMO, définissez la zone *MatchOptions* sur MQMO_NONE.
- Pour un appel MQGET dans une file d'attente partagée et que l'appel est une demande de navigation ou une extraction destructive d'un message de groupe, et que ni *MsgId* ni *CorrelId* ne doivent être mis en correspondance, l'appel MQGET est réémis toutes les 200 millisecondes jusqu'à ce qu'un message approprié arrive dans la file d'attente ou que l'intervalle d'attente expire.

[1] Un appel MQGET spécifiant l'option MQGMO_LOCK est traité comme un appel non navigant.

Cette méthode entraîne une surcharge de traitement inattendue et n'est pas une méthode efficace d'extraction de messages lorsque les messages sont ajoutés rarement. Pour éviter cette surcharge pour le cas de navigation, spécifiez *MsgId* (s'il n'est pas indexé ou indexé par *MsgId*) ou *CorrelId* (s'il est indexé par *CorrelId*) correspondant sur l'appel MQGET .

MQGMO_WAIT est ignoré s'il est spécifié avec MQGMO_BROWSE_MSG_UNDER_CURSOR ou MQGMO_MSG_UNDER_CURSOR; aucune erreur n'est générée.

MQGMO_NO_WAIT
L'application n'attend pas si aucun message approprié n'est disponible. MQGMO_NO_WAIT est l'opposé de MQGMO_WAIT. MQGMO_NO_WAIT est défini pour faciliter la documentation du programme. Il s'agit de la valeur par défaut si aucune n'est spécifiée.

MQGMO_SET_SIGNAL
Utilisez cette option avec les zones *Signal1* et *Signal2* . Il permet aux applications de poursuivre avec d'autres travaux en attendant l'arrivée d'un message. Il permet également (si des fonctions de système d'exploitation appropriées sont disponibles) aux applications d'attendre que les messages arrivent dans plusieurs files d'attente.

Remarque : L'option MQGMO_SET_SIGNAL est spécifique à l'environnement ; ne l'utilisez pas pour les applications que vous souhaitez porter.

Dans deux cas, l'appel se termine de la même manière que si cette option n'avait pas été spécifiée:

1. Si un message actuellement disponible répond aux critères spécifiés dans le descripteur de message.
2. Si une erreur de paramètre ou une autre erreur synchrone est détectée.

Si aucun message répondant aux critères spécifiés dans le descripteur de message n'est actuellement disponible, le contrôle revient à l'application sans attendre l'arrivée d'un message. Les paramètres *CompCode* et *Reason* sont définis sur MQCC_WARNING et MQRC_SIGNAL_REQUEST_ACCEPTED. Les autres zones de sortie du descripteur de message et les paramètres de sortie de l'appel MQGET ne sont pas définis. Lorsqu'un message approprié arrive plus tard, le signal est délivré en déposant la BCE.

L'appelant doit ensuite émettre à nouveau l'appel MQGET pour extraire le message. L'application peut attendre ce signal, en utilisant des fonctions fournies par le système d'exploitation.

Si le système d'exploitation fournit un mécanisme d'attente multiple, vous pouvez l'utiliser pour attendre qu'un message arrive dans l'une des files d'attente.

Si un *WaitInterval* différent de zéro est spécifié, le signal est délivré après l'expiration de l'intervalle d'attente. Le gestionnaire de files d'attente peut également annuler l'attente, auquel cas le signal est fourni.

Plusieurs appels MQGET peuvent définir un signal pour le même message. L'ordre dans lequel les applications sont activées est le même que celui décrit pour MQGMO_WAIT.

Si plusieurs appels MQGET attendent le même message, chaque appel en attente est considéré comme identique. Les appels peuvent inclure un mélange d'options d'attente et de signal.

Dans certaines conditions, l'appel MQGET peut extraire un message et un signal résultant de l'arrivée du même message peut être distribué. Lorsqu'un signal est délivré, une application doit être préparée pour qu'aucun message ne soit disponible.

Un descripteur de file d'attente ne peut pas comporter plus d'une demande de signal en attente.

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO_UNLOCK
- MQGMO_WAIT

Pour un appel MQGET dans une file d'attente partagée et que l'appel est une demande de navigation ou une extraction destructive d'un message de groupe, et que ni *MsgId* ni *CorrelId* ne doivent être mis en correspondance, le signal ECB de l'utilisateur est envoyé MQEC_MSG_ARRIVED après 200 millisecondes.

Cela se produit, même si un message approprié peut ne pas être arrivé dans la file d'attente tant que l'intervalle d'attente n'est pas arrivé à expiration, lorsque la file d'attente est envoyée avec MQEC_WAIT_INTERVAL_EXPIRED. Lorsque MQEC_MSG_ARRIVED est envoyé, vous devez émettre à nouveau un deuxième appel MQGET pour extraire le message, le cas échéant.

Cette technique est utilisée pour vous assurer que vous êtes informé en temps utile de l'arrivée d'un message, mais elle peut apparaître comme une surcharge de traitement inattendue par rapport à une séquence d'appels similaire sur une file d'attente non partagée.

Il ne s'agit pas d'une méthode efficace d'extraction de messages lorsque les messages sont rarement ajoutés. Pour éviter cette surcharge pour le cas de navigation, spécifiez *MsgId* (s'il n'est pas indexé ou indexé par *MsgId*) ou *CorrelId* (s'il est indexé par *CorrelId*) correspondant sur l'appel MQGET .

Cette option est prise en charge sous z/OS uniquement.

MQGMO_FAIL_IF_QUIESCING
Forcez l'échec de l'appel MQGET si le gestionnaire de files d'attente est à l'état de mise au repos.

Sous z/OS, cette option force également l'échec de l'appel MQGET si la connexion (pour une application CICS ou IMS) est à l'état de mise au repos.

Si cette option est spécifiée avec MQGMO_WAIT ou MQGMO_SET_SIGNALet que l'attente ou le signal est en attente au moment où le gestionnaire de files d'attente passe à l'état de mise au repos:

- L'attente est annulée et l'appel renvoie le code achèvement MQCC_FAILED avec le code anomalie MQRC_Q_MGR_QUIESCING ou MQRC_CONNECTION_QUIESCING.
- Le signal est annulé avec un code achèvement de signal spécifique à l'environnement.

  Sous z/OS, le signal se termine par le code achèvement d'événement MQEC_Q_MGR_QUIESCING ou MQEC_CONNECTION_QUIESCING.

Si MQGMO_FAIL_IF_QUIESCING n'est pas spécifié et que le gestionnaire de files d'attente ou la connexion passe à l'état de mise au repos, l'attente ou le signal n'est pas annulé.

Options de point de synchronisation: Les options suivantes concernent la participation de l'appel MQGET dans une unité d'oeuvre:

MQGMO_SYNCPOINT
La demande consiste à fonctionner dans le cadre des protocoles d'unité de travail normaux. Le message est marqué comme étant indisponible pour d'autres applications, mais il est supprimé de la file d'attente uniquement lorsque l'unité de travail est validée. Le message est de nouveau disponible si l'unité d'oeuvre est annulée.

Vous pouvez laisser MQGMO_SYNCPOINT et MQGMO_NO_SYNCPOINT non définis. Dans ce cas, l'inclusion de la demande get dans les protocoles d'unité de travail est déterminée par l'environnement exécutant le gestionnaire de files d'attente. Il n'est pas déterminé par l'environnement qui exécute l'application. Sous z/OS, la demande d'obtention se trouve dans une unité de travail. Dans tous les autres environnements, la demande d'obtention ne se trouve pas dans une unité de travail.

En raison de ces différences, une application que vous souhaitez porter ne doit pas autoriser cette option à utiliser par défaut ; spécifiez explicitement MQGMO_SYNCPOINT ou MQGMO_NO_SYNCPOINT .

Cette option n'est pas valide avec l'une des options suivantes:

- 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
La demande doit fonctionner dans les protocoles d'unité de travail normaux, mais *uniquement* si le message extrait est persistant. Un message persistant a la valeur MQPER_PERSISTENT dans la zone *Persistence* de MQMD.

- Si le message est persistant, le gestionnaire de files d'attente traite l'appel comme si l'application avait spécifié MQGMO_SYNCPOINT.
- Si le message n'est pas persistant, le gestionnaire de files d'attente traite l'appel comme si l'application avait spécifié MQGMO_NO_SYNCPOINT.

Cette option n'est pas valide avec l'une des options suivantes:

- 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

Cette option est prise en charge dans les environnements suivants: AIX, HP-UX, z/OS, IBM i, Solaris et Linux, ainsi que les clients WebSphere MQ MQI connectés à ces systèmes.

MQGMO_NO_SYNCPOINT
La demande consiste à fonctionner en dehors des protocoles d'unité de travail normaux. Si vous obtenez un message sans option de navigation, il est immédiatement supprimé de la file d'attente. Le message ne peut pas être rendu disponible à nouveau en raison de l'annulation de l'unité d'oeuvre.

Cette option est utilisée si vous spécifiez MQGMO_BROWSE_FIRST ou MQGMO_BROWSE_NEXT.

Vous pouvez laisser MQGMO_SYNCPOINT et MQGMO_NO_SYNCPOINT non définis. Dans ce cas, l'inclusion de la demande get dans les protocoles d'unité de travail est déterminée par l'environnement exécutant le gestionnaire de files d'attente. Il n'est pas déterminé par l'environnement qui exécute l'application. Sous z/OS, la demande d'obtention se trouve dans une unité de travail. Dans tous les autres environnements, la demande d'obtention ne se trouve pas dans une unité de travail.

En raison de ces différences, une application que vous souhaitez porter ne doit pas autoriser cette option à utiliser la valeur par défaut ; spécifiez explicitement MQGMO_SYNCPOINT ou MQGMO_NO_SYNCPOINT .

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT

MQGMO_MARK_SKIP_BACKOUT
Annulation d'une unité de travail sans réinstanciation dans la file d'attente du message marqué avec cette option.

Cette option est prise en charge uniquement sous z/OS.

Si cette option est spécifiée, MQGMO_SYNCPOINT doit également être spécifié. MQGMO_MARK_SKIP_BACKOUT n'est pas valide avec l'une des options suivantes:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT

- MQGMO_LOCK
- MQGMO_NO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

Remarque : Sur IMS et CICS, vous devrez peut-être émettre un appel WebSphere MQ supplémentaire après l'annulation d'une unité de travail contenant un message marqué avec MQGMO_MARK_SKIP_BACKOUT. Vous devez émettre un appel WebSphere MQ avant de valider la nouvelle unité de travail contenant le message marqué. L'appel peut être n'importe quel appel WebSphere MQ de votre choix.

1. Sous IMS, si vous n'avez pas appliqué IMS APAR PN60855 et que vous exécutez une application IMS MPP ou BMP.
2. Sous CICS, si vous exécutez une application.

Dans les deux cas, émettez un appel WebSphere MQ avant de valider la nouvelle unité de travail contenant le message annulé.

Remarque : Dans une unité de travail, il ne peut y avoir qu'une seule demande d'obtention marquée comme annulation ignorée, ainsi qu'aucune ou plusieurs demandes d'obtention non marquées.

Si une application annule une unité de travail, un message qui a été extrait à l'aide de MQGMO_MARK_SKIP_BACKOUT n'est pas restauré à son état précédent. Les autres mises à jour de ressource sont annulées. Le message est traité comme s'il avait été extrait dans une nouvelle unité de travail démarrée par la demande d'annulation. Le message est extrait sans l'option MQGMO_MARK_SKIP_BACKOUT .

MQGMO_MARK_SKIP_BACKOUT est utile si, après la modification de certaines ressources, il apparaît que l'unité de travail ne peut pas aboutir. Si vous omettez cette option, l'annulation de l'unité d'oeuvre rétablit le message dans la file d'attente. La même séquence d'événements se produit à nouveau, lors de la prochaine extraction du message.

Toutefois, si vous spécifiez MQGMO_MARK_SKIP_BACKOUT sur l'appel MQGET d'origine, l'annulation de l'unité de travail annule les mises à jour apportées aux autres ressources. Le message est traité comme s'il avait été extrait sous une nouvelle unité de travail. L'application peut effectuer le traitement d'erreurs approprié. Il peut envoyer un message de rapport à l'expéditeur du message d'origine ou placer le message d'origine dans la file d'attente des messages non livrés. Il peut ensuite valider la nouvelle unité de travail. La validation de la nouvelle unité d'oeuvre supprime définitivement le message de la file d'attente d'origine.

MQGMO_MARK_SKIP_BACKOUT marque un message physique unique. Si le message appartient à un groupe de messages, les autres messages du groupe ne sont pas marqués. De même, si le message marqué est un segment d'un message logique, les autres segments du message logique ne sont pas marqués.

Tout message d'un groupe peut être marqué, mais si des messages sont extraits à l'aide de MQGMO_LOGICAL_ORDER, il est avantageux de marquer le premier message du groupe. Si l'unité de travail est annulée, le premier message (marqué) est déplacé vers la nouvelle unité de travail. Le deuxième message et les messages suivants du groupe sont réintégrés dans la file d'attente. Les messages laissés dans la file d'attente ne peuvent pas être extraits par une autre application à l'aide de MQGMO_LOGICAL_ORDER. Le premier message du groupe n'est plus dans la file d'attente. Toutefois, l'application qui a sauvegardé l'unité de travail peut extraire le deuxième message et les messages ultérieurs dans la nouvelle unité de travail à l'aide de l'option MQGMO_LOGICAL_ORDER . Le premier message a déjà été extrait.

Il peut arriver que vous deviez revenir sur la nouvelle unité de travail. Par exemple, parce que la file d'attente de rebut est saturée et que le message ne doit pas être supprimé. L'annulation de la nouvelle unité d'oeuvre réintroduit le message dans la file d'attente d'origine, ce qui empêche la perte du message. Toutefois, dans cette situation, le traitement ne peut pas continuer. Après l'annulation de la nouvelle unité de travail, l'application doit informer l'opérateur ou l'administrateur qu'une erreur irrémédiable s'est produite, puis terminer.

MQGMO_MARK_SKIP_BACKOUT ne fonctionne que si l'unité de travail contenant la demande get est interrompue par l'application qui l'a sauvegardé. Si l'unité de travail contenant la demande get est annulée en raison de l'échec de la transaction ou du système, MQGMO_MARK_SKIP_BACKOUT est ignoré. Tout message extrait à l'aide de cette option est réintégré dans la file d'attente de la même manière que les messages extraits sans cette option.

Options de navigation: Les options suivantes concernent la navigation dans les messages de la file d'attente:

MQGMO_BROWSE_FIRST
Lorsqu'une file d'attente est ouverte avec l'option MQOO_BROWSE, un curseur de navigation est établi, positionné logiquement avant le premier message de la file d'attente. Vous pouvez ensuite utiliser des appels MQGET en spécifiant l'option MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXTou MQGMO_BROWSE_MSG_UNDER_CURSOR pour extraire les messages de la file d'attente de façon non destructive. Le curseur de navigation marque la position, dans les messages de la file d'attente, à partir de laquelle le prochain appel MQGET avec MQGMO_BROWSE_NEXT recherche un message approprié.

MQGMO_BROWSE_FIRST n'est pas valide avec l'une des options suivantes:

- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

Un appel MQGET avec MQGMO_BROWSE_FIRST ignore la position précédente du curseur de navigation. Le premier message de la file d'attente qui remplit les conditions spécifiées dans le descripteur de message est extrait. Le message reste dans la file d'attente et le curseur de navigation est positionné sur ce message.

Après cet appel, le curseur de navigation est positionné sur le message qui a été renvoyé. Le message peut être supprimé de la file d'attente avant l'émission du prochain appel MQGET avec MQGMO_BROWSE_NEXT . Dans ce cas, le curseur de navigation reste à l'emplacement de la file d'attente occupé par le message, même si cette position est maintenant vide.

Utilisez l'option MQGMO_MSG_UNDER_CURSOR avec un appel MQGET non-browse pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par un appel MQGET non navigant, même si le même descripteur *Hobj* est utilisé. Elle n'est pas non plus déplacée par un appel browse MQGET qui renvoie le code achèvement MQCC_FAILEDou le code anomalie MQRC_TRUNCATED_MSG_FAILED.

Spécifiez l'option MQGMO_LOCK avec cette option pour verrouiller le message consulté.

Vous pouvez spécifier MQGMO_BROWSE_FIRST avec n'importe quelle combinaison valide des options MQGMO_* et MQMO_* qui contrôlent le traitement des messages dans les groupes et les segments de messages logiques.

Si vous spécifiez MQGMO_LOGICAL_ORDER, les messages sont consultés dans l'ordre logique. Si vous omettez cette option, les messages sont parcourus dans l'ordre physique. Si vous spécifiez MQGMO_BROWSE_FIRST, vous pouvez basculer entre l'ordre logique et l'ordre physique. Les appels MQGET suivants utilisant MQGMO_BROWSE_NEXT parcourent la file d'attente dans le même ordre que l'appel le plus récent qui a spécifié MQGMO_BROWSE_FIRST pour l'identificateur de file d'attente.

Le gestionnaire de files d'attente conserve deux ensembles d'informations de groupe et de segment pour les appels MQGET . Les informations de groupe et de segment pour les appels de navigation sont conservées séparément des informations pour les appels qui suppriment les messages de

la file d'attente. Si vous spécifiez MQGMO_BROWSE_FIRST, le gestionnaire de files d'attente ignore les informations de groupe et de segment à parcourir. Il analyse la file d'attente comme s'il n'existait aucun groupe en cours et aucun message logique en cours. Si l'appel MQGET aboutit, code achèvement MQCC_OK ou MQCC_WARNING, les informations de groupe et de segment à parcourir sont définies sur celles du message renvoyé. Si l'appel échoue, les informations de groupe et de segment restent les mêmes qu'avant l'appel.

MQGMO_BROWSE_NEXT
Permet d'avancer le curseur de navigation jusqu'au message suivant de la file d'attente qui répond aux critères de sélection spécifiés dans l'appel MQGET . Le message est renvoyé à l'application, mais reste dans la file d'attente.

MQGMO_BROWSE_NEXT n'est pas valide avec l'une des options suivantes:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

MQGMO_BROWSE_NEXT se comporte de la même manière que MQGMO_BROWSE_FIRST, s'il s'agit du premier appel à parcourir une file d'attente, une fois que la file d'attente a été ouverte pour consultation.

Le message sous le curseur peut être supprimé de la file d'attente avant l'émission du prochain appel MQGET avec MQGMO_BROWSE_NEXT . Le curseur de navigation reste logiquement à l'emplacement de la file d'attente occupé par le message, même si cette position est maintenant vide.

Les messages sont stockés dans la file d'attente de deux manières:

- FIFO dans la priorité (MQMDS_PRIORITY) ou
- FIFO *indépendamment* de la priorité (MQMDS_FIFO)

L'attribut de file d'attente *MsgDeliverySequence* indique la méthode qui s'applique (voir «Attributs des files d'attente», à la page 824 pour plus de détails).

Une file d'attente peut avoir un *MsgDeliverySequence* de MQMDS_PRIORITY. Un message arrive dans la file d'attente dont la priorité est supérieure à celle indiquée par le curseur de navigation. Dans ce cas, le message de priorité supérieure est introuvable lors du balayage en cours de la file d'attente à l'aide de MQGMO_BROWSE_NEXT. Il ne peut être trouvé qu'une fois que le curseur de navigation a été réinitialisé avec MQGMO_BROWSE_FIRSTou en rouvrant la file d'attente.

L'option MQGMO_MSG_UNDER_CURSOR peut être utilisée avec un appel MQGET non-browse, si nécessaire, pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par des appels MQGET non navigant utilisant le même descripteur *Hobj* .

Spécifiez l'option MQGMO_LOCK avec cette option pour verrouiller le message consulté.

Vous pouvez spécifier MQGMO_BROWSE_NEXT avec n'importe quelle combinaison valide des options MQGMO_* et MQMO_* qui contrôlent le traitement des messages dans les groupes et les segments de messages logiques.

Si vous spécifiez MQGMO_LOGICAL_ORDER, les messages sont consultés dans l'ordre logique. Si vous omettez cette option, les messages sont parcourus dans l'ordre physique. Si vous spécifiez MQGMO_BROWSE_FIRST, vous pouvez basculer entre l'ordre logique et l'ordre physique. Les appels MQGET suivants utilisant MQGMO_BROWSE_NEXT parcourent la file d'attente dans le même ordre que l'appel le plus récent qui a spécifié MQGMO_BROWSE_FIRST pour l'identificateur de file d'attente.

L'appel échoue avec le code anomalie MQRC_INCONSISTENT_BROWSE si cette condition n'est pas satisfaite.

Remarque : Faites attention lors de l'utilisation d'un appel MQGET pour parcourir les données au-delà de la fin d'un groupe de messages si MQGMO_LOGICAL_ORDER n'est pas spécifié. Par exemple, supposons que le dernier message du groupe précède le premier message du groupe dans la file d'attente. L'utilisation de MQGMO_BROWSE_NEXT pour parcourir les données au-delà de la fin du groupe, en spécifiant MQMO_MATCH_MSG_SEQ_NUMBER avec *MsgSeqNumber* défini sur 1 , renvoie le premier message du groupe déjà parcouru. Ce résultat peut se produire immédiatement ou un certain nombre d'appels MQGET par la suite s'il existe des groupes intermédiaires. La même remarque s'applique à un message logique qui ne se trouve pas dans un groupe.

Les informations de groupe et de segment pour les appels de navigation sont conservées séparément des informations pour les appels qui suppriment les messages de la file d'attente.

MQGMO_BROWSE_MSG_UNDER_CURSOR
Extrayez le message désigné par le curseur de navigation de manière non destructive, quelles que soient les options MQMO_* spécifiées dans la zone *MatchOptions* de MQGMO.

MQGMO_BROWSE_MSG_UNDER_CURSOR n'est pas valide avec l'une des options suivantes:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte pour consultation.

Le message désigné par le curseur de navigation est celui qui a été extrait pour la dernière fois à l'aide de l'option MQGMO_BROWSE_FIRST ou MQGMO_BROWSE_NEXT . L'appel échoue si aucun de ces appels n'a été émis pour cette file d'attente depuis son ouverture. L'appel échoue également si le message qui se trouvait sous le curseur de navigation a depuis été extrait de façon destructive.

La position du curseur de navigation n'est pas modifiée par cet appel.

L'option MQGMO_MSG_UNDER_CURSOR peut être utilisée avec un appel MQGET non navigant, pour supprimer le message de la file d'attente.

Le curseur de navigation n'est pas déplacé par un appel MQGET non navigant, même si le même descripteur *Hobj* est utilisé. Elle n'est pas non plus déplacée par un appel browse MQGET qui renvoie le code achèvement MQCC_FAILEDou le code anomalie MQRC_TRUNCATED_MSG_FAILED.

Si MQGMO_BROWSE_MSG_UNDER_CURSOR est spécifié avec MQGMO_LOCK:

- Si un message est déjà verrouillé, il doit être celui qui se trouve sous le curseur, de sorte qu'il est renvoyé sans déverrouillage ni nouveau verrouillage. Le message reste verrouillé.
- S'il n'y a pas de message verrouillé et qu'il y a un message sous le curseur de navigation, il est verrouillé et renvoyé à l'application. S'il n'y a pas de message sous le curseur de navigation, l'appel échoue.

Si MQGMO_BROWSE_MSG_UNDER_CURSOR est spécifié sans MQGMO_LOCK:

- Si un message est déjà verrouillé, il doit être celui qui se trouve sous le curseur. Le message est renvoyé à l'application, puis déverrouillé. Etant donné que le message est maintenant déverrouillé, il n'est pas garanti qu'il puisse être consulté à nouveau ou extrait de façon destructive par la même application. Il se peut qu'il ait été extrait de façon destructive par une autre application qui extrait des messages de la file d'attente.
- S'il n'y a pas de message verrouillé et qu'il y a un message sous le curseur de navigation, il est renvoyé à l'application. S'il n'y a pas de message sous le curseur de navigation, l'appel échoue.

Si MQGMO_COMPLETE_MSG est spécifié avec MQGMO_BROWSE_MSG_UNDER_CURSOR, le curseur de navigation doit identifier un message dont la zone *Offset* dans MQMD est zéro. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC_INVALID_MSG_UNDER_CURSOR.

Les informations de groupe et de segment pour les appels de navigation sont conservées séparément des informations pour les appels qui suppriment les messages de la file d'attente.

MQGMO_MSG_UNDER_CURSOR
Extrayez le message désigné par le curseur de navigation, quelles que soient les options MQMO_* spécifiées dans la zone *MatchOptions* de MQGMO. Le message est supprimé de la file d'attente.

Le message désigné par le curseur de navigation est celui qui a été extrait pour la dernière fois à l'aide de l'option MQGMO_BROWSE_FIRST ou MQGMO_BROWSE_NEXT .

Si MQGMO_COMPLETE_MSG est spécifié avec MQGMO_MSG_UNDER_CURSOR, le curseur de navigation doit identifier un message dont la zone *Offset* dans MQMD est zéro. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC_INVALID_MSG_UNDER_CURSOR.

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT
- MQGMO_UNLOCK

Il s'agit également d'une erreur si la file d'attente n'a pas été ouverte à la fois pour l'exploration et pour l'entrée. Si le curseur de navigation ne pointe pas vers un message récupérable, une erreur est renvoyée par l'appel MQGET .

MQGMO_MARK_BROWSE_HANDLE
Le message renvoyé par un MQGETréussi, ou identifié par le *MsgToken*renvoyé, est marqué. La marque est spécifique au descripteur d'objet utilisé dans l'appel.

Le message n'est pas supprimé de la file d'attente.

MQGMO_MARK_BROWSE_HANDLE est valide uniquement si l'une des options suivantes est également spécifiée:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT

MQGMO_MARK_BROWSE_HANDLE n'est pas valide avec l'une des options suivantes:

- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK
- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

Le message reste dans cet état jusqu'à ce que l'un des événements suivants se produise:

- La poignée d'objet concernée est fermée, normalement ou non.
- Le message n'est pas marqué pour ce descripteur par un appel à MQGET avec l'option MQGMO_UNMARK_BROWSE_HANDLE.
- Le message est renvoyé à partir d'un appel à MQGETdestructif, qui se termine par MQCC_OK ou MQCC_WARNING. L'état du message reste modifié même si le MQGET est ultérieurement annulé.
- Le message expire.

MQGMO_MARK_BROWSE_CO_OP
Le message renvoyé par un MQGETréussi ou identifié par le *MsgToken*renvoyé est marqué pour tous les descripteurs de l'ensemble coopérant.

La marque de niveau coopératif s'ajoute à toute marque de niveau de descripteur qui aurait pu être définie.

Le message n'est pas supprimé de la file d'attente.

MQGMO_MARK_BROWSE_CO_OP est valide uniquement si le descripteur d'objet utilisé a été renvoyé par un appel à MQOPEN qui a spécifié MQOO_CO_OP. Vous devez également spécifier l'une des options MQGMO suivantes:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK
- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

Si le message est déjà marqué et que l'option MQGMO_UNMARKED_BROWSE_MSG n'est pas spécifiée, l'appel échoue avec MQCC_FAILED et le code anomalie MQRC_MSG_MARKED_BROWSE_CO_OP.

Le message reste dans cet état jusqu'à ce que l'un des événements suivants se produise:

- Tous les descripteurs d'objet de l'ensemble coopérant sont fermés.
- Le message n'est pas marqué pour les navigateurs associés par un appel à MQGET avec l'option MQGMO_UNMARK_BROWSE_CO_OP.
- Le message est automatiquement démarqué par le gestionnaire de files d'attente.
- Le message est renvoyé à partir d'un appel à un MQGETnon navigant. L'état du message reste modifié même si le MQGET est ultérieurement annulé.
- Le message expire.

MQGMO_UNMARKED_BROWSE_MSG
Un appel à MQGET qui spécifie MQGMO_UNMARKED_BROWSE_MSG renvoie un message qui est considéré comme non marqué pour son descripteur. Il ne renvoie pas de message si le message a été marqué pour son descripteur. Il ne renvoie pas non plus le message si la file d'attente a été ouverte par un appel à MQOPEN, avec l'option MQOO_CO_OP, et que le message a été marqué par un membre de l'ensemble de coopération.

Cette option n'est pas valide avec l'une des options suivantes:

- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK
- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

MQGMO_UNMARK_BROWSE_CO_OP
Après un appel à MQGET qui spécifie cette option, le message n'est plus pris en compte par les descripteurs ouverts du jeu de descripteurs coopérants à marquer pour le jeu de descripteurs

coopérants. Le message est toujours considéré comme marqué au niveau de la poignée s'il a été marqué au niveau de la poignée avant cet appel.

L'utilisation de MQGMO_UNMARK_BROWSE_CO_OP est valide uniquement avec un descripteur renvoyé par un appel réussi à MQOPEN avec l'option MQOO_CO_OP. Le MQGET aboutit même si le message n'est pas considéré comme étant marqué par l'ensemble de descripteurs coopérant.

MQGMO_UNMARK_BROWSE_CO_OP n'est pas valide sur un appel MQGET non-browse, ou avec l'une des options suivantes:

- 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
Après un appel à MQGET qui spécifie cette option, le message localisé n'est plus considéré comme étant marqué par cet identificateur.

L'appel aboutit même si le message n'est pas marqué pour ce descripteur.

Cette option n'est pas valide sur un appel MQGET non-browse ou avec l'une des options suivantes:

- 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

Options de verrouillage: Les options suivantes concernent le verrouillage des messages dans la file d'attente:

MQGMO_LOCK
Verrouillez le message qui est parcouru, de sorte que le message devienne invisible pour tout autre descripteur ouvert pour la file d'attente. L'option ne peut être spécifiée que si l'une des options suivantes est également spécifiée:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_BROWSE_MSG_UNDER_CURSOR

Un seul message peut être verrouillé pour chaque descripteur de file d'attente. Il peut s'agir d'un message logique ou d'un message physique:

- Si vous spécifiez MQGMO_COMPLETE_MSG, tous les segments de message qui constituent le message logique sont verrouillés sur le descripteur de file d'attente. Les messages doivent tous être présents dans la file d'attente et disponibles pour extraction.
- Si vous omettez MQGMO_COMPLETE_MSG, un seul message physique est verrouillé pour l'identificateur de file d'attente. Si ce message se trouve être un segment d'un message logique, le segment verrouillé empêche les autres applications utilisant MQGMO_COMPLETE_MSG d'extraire ou de parcourir le message logique.

Le message verrouillé est toujours celui qui se trouve sous le curseur de navigation. Le message peut être supprimé de la file d'attente par un appel MQGET ultérieur qui spécifie l'option MQGMO_MSG_UNDER_CURSOR . D'autres appels MQGET utilisant le descripteur de file d'attente peuvent également supprimer le message (par exemple, un appel qui spécifie l'identificateur du message verrouillé).

Si l'appel renvoie le code achèvement MQCC_FAILEDou MQCC_WARNING avec le code anomalie MQRC_TRUNCATED_MSG_FAILED, aucun message n'est verrouillé.

Si l'application ne supprime pas le message de la file d'attente, le verrou est libéré par l'une des actions suivantes:

- Emission d'un autre appel MQGET pour cet indicateur, en spécifiant MQGMO_BROWSE_FIRST ou MQGMO_BROWSE_NEXT. Le verrou est libéré si l'appel se termine avec MQCC_OK ou MQCC_WARNING. Le message reste verrouillé si l'appel se termine par MQCC_FAILED. Cependant, les exceptions suivantes s'appliquent :
  - Le message n'est pas déverrouillé si MQCC_WARNING est renvoyé avec MQRC_TRUNCATED_MSG_FAILED.
  - Le message est déverrouillé si MQCC_FAILED est renvoyé avec MQRC_NO_MSG_AVAILABLE.

  Si vous spécifiez également MQGMO_LOCK, le message renvoyé est verrouillé. Si vous omettez MQGMO_LOCK, aucun message n'est verrouillé après l'appel.

  Si vous spécifiez MQGMO_WAITet qu'aucun message n'est immédiatement disponible, le message d'origine est déverrouillé avant le début de l'attente.
- Emission d'un autre appel MQGET pour ce descripteur, avec MQGMO_BROWSE_MSG_UNDER_CURSOR, sans MQGMO_LOCK. Le verrou est libéré si l'appel se termine avec MQCC_OK ou MQCC_WARNING. Le message reste verrouillé si l'appel se termine par MQCC_FAILED. Toutefois, l'exception suivante s'applique:
  - Le message n'est pas déverrouillé si MQCC_WARNING est renvoyé avec MQRC_TRUNCATED_MSG_FAILED.
- Emission d'un autre appel MQGET pour ce descripteur avec MQGMO_UNLOCK.
- Emission d'un appel MQCLOSE à l'aide de l'indicateur. MQCLOSE peut être implicite, en raison de l'arrêt de l'application.

Aucune option spéciale MQOPEN n'est requise pour spécifier MQGMO_LOCK, autre que MQOO_BROWSE, qui est nécessaire pour spécifier une option de navigation associée.

MQGMO_LOCK n'est pas valide avec l'une des options suivantes:

- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

MQGMO_LOCK n'est pas possible lorsque vous utilisez un client IBM WebSphere MQ sous HP Integrity NonStop Server sur un gestionnaire de files d'attente z/OS lorsqu'il est coordonné par TMF.

MQGMO_UNLOCK

Le message à déverrouiller doit avoir été préalablement verrouillé par un appel MQGET avec l'option MQGMO_LOCK . Si aucun message n'est verrouillé pour cet identificateur, l'appel se termine avec MQCC_WARNING et MQRC_NO_MSG_LOCKED.

Les paramètres *MsgDesc*, *BufferLength*, *Buffer*et *DataLength* ne sont pas vérifiés ou modifiés si vous spécifiez MQGMO_UNLOCK. Aucun message n'est renvoyé dans *Buffer*.

Aucune option d'ouverture spéciale n'est requise pour spécifier MQGMO_UNLOCK (bien que MQOO_BROWSE soit nécessaire pour émettre la demande de verrouillage en premier lieu).

Cette option n'est valide qu'avec les options suivantes:

- MQGMO_NO_WAIT
- MQGMO_NO_SYNCPOINT

Ces deux options sont supposées être spécifiées ou non.

Options de données de message: Les options suivantes concernent le traitement des données de message lorsque le message est lu à partir de la file d'attente:

MQGMO_ACCEPT_TRUNCATED_MSG
Si la mémoire tampon de messages est trop petite pour contenir le message complet, autorisez l'appel MQGET à remplir la mémoire tampon. MQGET remplit la mémoire tampon avec autant de messages que possible. Il émet un code achèvement d'avertissement et termine son traitement. Ce qui signifie que :

- Lors de la navigation dans les messages, le curseur de navigation est avancé jusqu'au message renvoyé.
- Lors de la suppression de messages, le message renvoyé est supprimé de la file d'attente.
- Le code anomalie MQRC_TRUNCATED_MSG_ACCEPTED est renvoyé si aucune autre erreur ne se produit.

Sans cette option, la mémoire tampon est toujours remplie avec autant de messages qu'elle peut contenir. Un code achèvement d'avertissement est émis, mais le traitement n'est pas terminé. Ce qui signifie que :

- Lors de la navigation dans les messages, le curseur de navigation n'est pas avancé.
- Lors de la suppression de messages, le message n'est pas supprimé de la file d'attente.
- Le code anomalie MQRC_TRUNCATED_MSG_FAILED est renvoyé si aucune autre erreur ne se produit.

MQGMO_CONVERT
Cette option convertit les données d'application du message pour qu'elles soient conformes aux valeurs *CodedCharSetId* et *Encoding* spécifiées dans le paramètre *MsgDesc* de l'appel MQGET . Les données sont converties avant d'être copiées dans le paramètre *Buffer* .

La zone *Format* spécifiée lors de l'insertion du message est prise en compte par le processus de conversion pour identifier la nature des données du message. Les données de message sont converties par le gestionnaire de files d'attente pour les formats intégrés et par un exit écrit par l'utilisateur pour les autres formats. Pour plus de détails sur l'exit de conversion de données, voir «Exit de conversion de données», à la page 898.

- Si la conversion aboutit, les zones *CodedCharSetId* et *Encoding* spécifiées dans le paramètre *MsgDesc* sont inchangées lors du retour de l'appel MQGET .
- Si seule la conversion échoue, les données de message sont renvoyées non converties. Les zones *CodedCharSetId* et *Encoding* de *MsgDesc* sont définies sur les valeurs du message non converti. Le code achèvement est MQCC_WARNING dans ce cas.

Dans les deux cas, ces zones décrivent l'identificateur de jeu de caractères et le codage des données de message renvoyées dans le paramètre *Buffer* .

Voir la zone *Format* décrite dans «MQMD-Descripteur de message», à la page 394 pour la liste des noms de format pour lesquels le gestionnaire de files d'attente effectue la conversion.

Options de groupe et de segment: Les options suivantes concernent le traitement des messages dans des groupes et des segments de messages logiques. Avant les descriptions d'option, voici quelques définitions des termes importants:

Message physique
Un message physique est la plus petite unité d'informations pouvant être placée dans une file d'attente ou supprimée d'une file d'attente. Il correspond souvent aux informations spécifiées ou extraites sur un seul appel MQPUT, MQPUT1ou MQGET . Chaque message physique possède son propre descripteur de message, MQMD. En règle générale, les messages physiques se distinguent par des

valeurs différentes pour l'identificateur de message, la zone *MsgId* dans MQMD. Le gestionnaire de files d'attente n'impose pas de valeurs différentes.

Message logique
Un message logique est une unité unique d'informations d'application. En l'absence de contraintes système, un message logique est identique à un message physique. Si les messages logiques sont volumineux, les contraintes du système peuvent rendre souhaitable ou nécessaire la division d'un message logique en deux ou plusieurs messages physiques, appelés segments.

Un message logique segmenté se compose d'au moins deux messages physiques ayant le même identificateur de groupe non null, zone *GroupId* dans MQMD. Ils ont le même numéro de séquence de message, zone *MsgSeqNumber* dans MQMD. Les segments se distinguent par des valeurs différentes pour le décalage de segment, zone *Offset* dans MQMD. Le décalage de segment est le décalage des données du message physique à partir du début des données du message logique. Chaque segment étant un message physique, les segments d'un message logique ont généralement des identificateurs de message différents.

Un message logique qui n'a pas été segmenté, mais pour lequel la segmentation a été autorisée par l'application émettrice, possède également un identificateur de groupe non null. Dans ce cas, il n'y a qu'un seul message physique avec cet identificateur de groupe si le message logique n'appartient pas à un groupe de messages. Les messages logiques, pour lesquels la segmentation a été inhibée par l'application émettrice, ont un identificateur de groupe nul, MQGI_NONE, sauf si le message logique appartient à un groupe de messages.

Groupe de messages
Un groupe de messages est un ensemble d'un ou de plusieurs messages logiques ayant le même identificateur de groupe non null. Les messages logiques du groupe se distinguent par des valeurs différentes pour le numéro de séquence de message. Le numéro de séquence est un entier compris entre 1 et n, où n est le nombre de messages logiques dans le groupe. Si un ou plusieurs des messages logiques sont segmentés, il y a plus de n messages physiques dans le groupe.

MQGMO_LOGICAL_ORDER
MQGMO_LOGICAL_ORDER contrôle l'ordre dans lequel les messages sont renvoyés par les appels MQGET successifs pour le descripteur de file d'attente. L'option doit être spécifiée à chaque appel.

Si MQGMO_LOGICAL_ORDER est spécifié pour des appels MQGET successifs pour le même descripteur de file d'attente, les messages des groupes sont renvoyés dans l'ordre de leur numéro de séquence de message. Les segments de messages logiques sont renvoyés dans l'ordre indiqué par leurs décalages de segment. Cet ordre peut être différent de l'ordre dans lequel ces messages et segments apparaissent dans la file d'attente.

Remarque : La spécification de MQGMO_LOGICAL_ORDER n'a aucune conséquence négative sur les messages qui n'appartiennent pas à des groupes et qui ne sont pas des segments. En effet, ces messages sont traités comme si chacun appartenait à un groupe de messages composé d'un seul message. Il est prudent de spécifier MQGMO_LOGICAL_ORDER lors de l'extraction de messages à partir de files d'attente qui contiennent un mélange de messages dans des groupes, des segments de message et des messages non segmentés dans des groupes.

Pour renvoyer les messages dans l'ordre requis, le gestionnaire de files d'attente conserve les informations de groupe et de segment entre les appels MQGET successifs. Les informations de groupe et de segment identifient le groupe de messages en cours et le message logique en cours pour l'identificateur de file d'attente. Il identifie également la position en cours dans le groupe et le message logique, et indique si les messages sont extraits dans une unité d'oeuvre. Etant donné que le gestionnaire de files d'attente conserve ces informations, l'application n'a pas besoin de définir les informations de groupe et de segment avant chaque appel MQGET . En particulier, cela signifie que l'application n'a pas besoin de définir les zones *GroupId*, *MsgSeqNumber*et *Offset* dans MQMD. Toutefois, l'application doit définir correctement l'option MQGMO_SYNCPOINT ou MQGMO_NO_SYNCPOINT à chaque appel.

Lorsque la file d'attente est ouverte, il n'y a pas de groupe de messages en cours ni de message logique en cours. Un groupe de messages devient le groupe de messages en cours lorsqu'un message comportant l'indicateur MQMF_MSG_IN_GROUP est renvoyé par l'appel MQGET .

Si MQGMO_LOGICAL_ORDER est spécifié lors d'appels successifs, ce groupe reste le groupe en cours jusqu'à ce qu'un message contenant:

- MQMF_LAST_MSG_IN_GROUP sans MQMF_SEGMENT (c'est-à-dire que le dernier message logique du groupe n'est pas segmenté), ou
- MQMF_LAST_MSG_IN_GROUP avec MQMF_LAST_SEGMENT (c'est-à-dire que le message renvoyé est le dernier segment du dernier message logique du groupe).

Lorsqu'un tel message est renvoyé, le groupe de messages est arrêté et, à la fin de l'appel à MQGET , il n'y a plus de groupe en cours. De la même manière, un message logique devient le message logique en cours lorsqu'un message comportant l'indicateur MQMF_SEGMENT est renvoyé par l'appel MQGET . Le message logique est arrêté lorsque le message comportant l'indicateur MQMF_LAST_SEGMENT est renvoyé.

Si aucun critère de sélection n'est spécifié, les appels MQGET successifs renvoient, dans l'ordre correct, les messages du premier groupe de messages de la file d'attente. Ils renvoient ensuite les messages pour le deuxième groupe de messages, et ainsi de suite, jusqu'à ce qu'il n'y ait plus de messages disponibles. Il est possible de sélectionner les groupes de messages particuliers renvoyés en spécifiant une ou plusieurs des options suivantes dans la zone *MatchOptions* :

- MQMO_MATCH_MSG_ID
- MQMO_MATCH_CORREL_ID
- MQMO_MATCH_GROUP_ID

Toutefois, ces options ne sont effectives que lorsqu'il n'y a pas de groupe de messages ou de message logique en cours. Voir la zone *MatchOptions* décrite dans «Options MQGMO-Get-message», à la page 345 pour plus de détails.

Le Tableau 506, à la page 365 présente les valeurs des zones *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber*et *Offset* que le gestionnaire de files d'attente recherche lorsqu'il tente de trouver un message à renvoyer lors de l'appel MQGET . Les règles s'appliquent à la fois à la suppression des messages de la file d'attente et à la consultation des messages de la file d'attente. Dans le tableau, l'un ou l'autre signifie Oui ou Non:

LOG ORD
Indique si l'option MQGMO_LOGICAL_ORDER est spécifiée sur l'appel.

Cur grp
Indique si un groupe de messages en cours existe avant l'appel.

Cur log msg
Indique si un message logique en cours existe avant l'appel.

Autres colonnes
Affiche les valeurs que le gestionnaire de files d'attente recherche. Précédent indique la valeur renvoyée pour la zone dans le message précédent pour le descripteur de file d'attente.

Tableau 506. Options MQGET relatives aux messages dans les groupes et les segments de messages logiques

Options que vous spécifiez	**Statut du groupe et du message de journal avant l'appel**		**Valeurs que le gestionnaire de files d'attente recherche**				
LOG ORD	**Cur grp**	**Cur log msg**	*MsgId*	*CorrelId*	*GroupId*	*MsgSeqNumber*	*Offset*
Oui	Non	Non	Contrôlé par *MatchOptions*	Contrôlé par *MatchOptions*	Contrôlé par *MatchOptions*	1	0

Tableau 506. Options MQGET relatives aux messages dans les groupes et les segments de messages logiques (suite)

Options que vous spécifiez	Statut du groupe et du message de journal avant l'appel		Valeurs que le gestionnaire de files d'attente recherche				
Oui	Non	Oui	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	1	Décalage précédent + longueur de segment précédent
Oui	Oui	Non	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	Numéro de séquence précédent + 1	0
Oui	Oui	Oui	Tout identificateur de message	Tout identificateur de corrélation	Identificateur de groupe précédent	Numéro de séquence précédent	Décalage précédent + longueur de segment précédent
Non	L'un ou l'autre	L'un ou l'autre	Contrôlé par *MatchOptions*	Contrôlé par *MatchOptions*	Contrôlé par *MatchOptions*	Contrôlé par *MatchOptions*	Contrôlé par *MatchOptions*

Si plusieurs groupes de messages sont présents dans la file d'attente et peuvent être renvoyés, ils sont renvoyés dans l'ordre déterminé par la position dans la file d'attente du premier segment du premier message logique de chaque groupe. En d'autres termes, les messages physiques dont les numéros de séquence sont 1 et les décalages 0 déterminent l'ordre dans lequel les groupes admissibles sont renvoyés.

L'option MQGMO_LOGICAL_ORDER affecte les unités de travail comme suit:

- Si le premier message logique ou segment d'un groupe est extrait dans une unité d'oeuvre, tous les autres messages logiques et segments du groupe doivent être extraits dans une unité d'oeuvre, si le même descripteur de file d'attente est utilisé. Toutefois, il n'est pas nécessaire de les extraire dans la même unité de travail. Cela permet à un groupe de messages composé de plusieurs messages physiques d'être scindé en plusieurs unités de travail consécutives pour l'identificateur de file d'attente.
- Si le premier message logique ou segment d'un groupe n'est *pas* extrait dans une unité de travail et que le même descripteur de file d'attente est utilisé, aucun des autres messages logiques et segments du groupe ne peut être extrait dans une unité de travail.

Si ces conditions ne sont pas satisfaites, l'appel MQGET échoue avec le code anomalie MQRC_INCONSISTENT_UOW.

Lorsque MQGMO_LOGICAL_ORDER est spécifié, le MQGMO fourni dans l'appel MQGET ne doit pas être inférieur à MQGMO_VERSION_2et le MQMD ne doit pas être inférieur à MQMD_VERSION_2. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC_WRONG_GMO_VERSION ou MQRC_WRONG_MD_VERSION, selon le cas.

Si MQGMO_LOGICAL_ORDER n'est *pas* spécifié pour les appels MQGET successifs de l'identificateur de file d'attente, les messages sont renvoyés indépendamment du fait qu'ils appartiennent à des groupes de messages ou qu'ils soient des segments de messages logiques. Cela signifie que les messages ou les segments d'un groupe ou d'un message logique particulier peuvent être renvoyés dans l'ordre, ou mélangés avec des messages ou des segments d'autres groupes ou messages logiques, ou avec des messages qui ne sont pas dans des groupes et qui ne sont pas des segments. Dans ce cas,

les messages particuliers renvoyés par les appels MQGET successifs sont contrôlés par les options MQMO_* spécifiées sur ces appels (voir la zone *MatchOptions* décrite dans «Options MQGMO-Get-message», à la page 345 pour plus de détails sur ces options).

Il s'agit de la technique qui peut être utilisée pour redémarrer un groupe de messages ou un message logique au milieu, après une défaillance du système. Lorsque le système redémarre, l'application peut définir les zones *GroupId*, *MsgSeqNumber*, *Offset*et *MatchOptions* sur les valeurs appropriées, puis émettre l'appel MQGET avec MQGMO_SYNCPOINT ou MQGMO_NO_SYNCPOINT défini, mais *sans* spécifier MQGMO_LOGICAL_ORDER. Si cet appel aboutit, le gestionnaire de files d'attente conserve les informations de groupe et de segment, et les appels MQGET suivants utilisant cet identificateur de file d'attente peuvent spécifier MQGMO_LOGICAL_ORDER comme normal.

Les informations de groupe et de segment que le gestionnaire de files d'attente conserve pour l'appel MQGET sont distinctes des informations de groupe et de segment qu'il conserve pour l'appel MQPUT. En outre, le gestionnaire de files d'attente conserve des informations distinctes pour:

- MQGET appels qui suppriment des messages de la file d'attente.
- MQGET appels qui parcourent les messages de la file d'attente.

Pour un descripteur de file d'attente donné, l'application peut combiner des appels MQGET qui spécifient MQGMO_LOGICAL_ORDER avec des appels MQGET qui ne le font pas. Notez toutefois les points suivants:

- Si vous omettez MQGMO_LOGICAL_ORDER, chaque appel MQGET réussi entraîne le gestionnaire de files d'attente à définir les informations de groupe et de segment sauvegardées sur les valeurs correspondant au message renvoyé ; cette opération remplace les informations de groupe et de segment existantes conservées par le gestionnaire de files d'attente pour l'identificateur de file d'attente. Seules les informations appropriées à l'action de l'appel (parcourir ou supprimer) sont modifiées.
- Si vous omettez MQGMO_LOGICAL_ORDER, l'appel n'échoue pas s'il existe un groupe de messages en cours ou un message logique ; l'appel peut aboutir avec un code achèvement MQCC_WARNING . Le Tableau 507, à la page 367 montre les différentes observations qui peuvent survenir. Dans ces cas, si le code achèvement n'est pas MQCC_OK, le code anomalie est l'un des suivants (selon le cas):
  - MQRC_INCOMPLETE_GROUP
  - MQRC_INCOMPLETE_MSG
  - MQRC_INCONSISTENT_UOW

  Remarque : Le gestionnaire de files d'attente ne vérifie pas les informations de groupe et de segment lors de la consultation d'une file d'attente ou lors de la fermeture d'une file d'attente qui a été ouverte pour consultation mais pas d'entrée ; dans ces cas, le code achèvement est toujours MQCC_OK (en supposant qu'il n'y a pas d'autres erreurs).

Tableau 507. Résultat lorsque l'appel MQGET ou MQCLOSE n'est pas cohérent avec les informations de groupe et de segment

L'appel en cours est	L'appel précédent était MQGET avec MQGMO_LOGICAL_ORDER	L'appel précédent était MQGET sans MQGMO_LOGICAL_ORDER
MQGET avec MQGMO_LOGICAL_ORDER	MQCC_FAILED	MQCC_FAILED
MQGET sans MQGMO_LOGICAL_ORDER	MQCC_WARNING	MQCC_OK
MQCLOSE avec un groupe non terminé ou un message logique	MQCC_WARNING	MQCC_OK

Les applications qui souhaitent extraire des messages et des segments dans l'ordre logique sont recommandées pour spécifier MQGMO_LOGICAL_ORDER, car il s'agit de l'option la plus simple à utiliser. Cette option dispense l'application de la nécessité de gérer les informations de groupe

et de segment, car le gestionnaire de files d'attente gère ces informations. Cependant, les applications spécialisées peuvent avoir besoin de plus de contrôle que celui fourni par l'option MQGMO_LOGICAL_ORDER , et cela peut être réalisé en ne spécifiant pas cette option. L'application doit ensuite s'assurer que les zones *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber*et *Offset* dans MQMD, et les options MQMO_* dans *MatchOptions* dans MQGMO, sont correctement définies, avant chaque appel MQGET .

Par exemple, une application qui souhaite *réacheminer* les messages physiques qu'elle reçoit, indépendamment du fait que ces messages se trouvent dans des groupes ou des segments de messages logiques, ne doit *pas* spécifier MQGMO_LOGICAL_ORDER. Dans un réseau complexe avec plusieurs chemins entre les gestionnaires de files d'attente d'envoi et de réception, les messages physiques peuvent arriver dans le désordre. En spécifiant ni MQGMO_LOGICAL_ORDER, ni le MQPMO_LOGICAL_ORDER correspondant sur l'appel MQPUT, l'application de réacheminement peut extraire et réacheminer chaque message physique dès qu'il arrive, sans avoir à attendre le message suivant dans l'ordre logique d'arrivée.

Vous pouvez spécifier MQGMO_LOGICAL_ORDER avec l'une des autres options MQGMO_* et avec diverses options MQMO_* dans des circonstances appropriées (voir ci-dessus).

- Sous z/OS, cette option est prise en charge pour les files d'attente privées et partagées, mais la file d'attente doit avoir le type d'index MQIT_GROUP_ID. Pour les files d'attente partagées, l'objet CFSTRUCT que la file d'attente mappe doit être au niveau CFLEVEL (3) ou CFLEVEL (4).
- Sous AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ MQI connectés à ces systèmes, cette option est prise en charge pour toutes les files d'attente locales.

MQGMO_COMPLETE_MSG
Seul un message logique complet peut être renvoyé par l'appel MQGET . Si le message logique est segmenté, le gestionnaire de files d'attente réassemble les segments et renvoie le message logique complet à l'application. Le fait que le message logique ait été segmenté n'est pas évident pour l'application qui l'extrait.

Remarque : Il s'agit de la seule option qui permet au gestionnaire de files d'attente de réassembler des segments de message. S'ils ne sont pas spécifiés, les segments sont renvoyés individuellement à l'application s'ils sont présents dans la file d'attente (et répondent aux autres critères de sélection spécifiés dans l'appel MQGET). Les applications qui ne souhaitent pas recevoir de segments individuels doivent toujours spécifier MQGMO_COMPLETE_MSG.

Pour utiliser cette option, l'application doit fournir une mémoire tampon suffisamment grande pour contenir le message complet, ou spécifier l'option MQGMO_ACCEPT_TRUNCATED_MSG .

Si la file d'attente contient des messages segmentés dont certains des segments sont manquants (peut-être parce qu'ils ont été retardés dans le réseau et qu'ils ne sont pas encore arrivés), la spécification de MQGMO_COMPLETE_MSG empêche l'extraction de segments appartenant à des messages logiques incomplets. Toutefois, ces segments de message contribuent toujours à la valeur de l'attribut de file d'attente *CurrentQDepth* ; cela signifie qu'il peut n'y avoir aucun message logique récupérable, même si *CurrentQDepth* est supérieur à zéro.

Pour les messages *persistants* , le gestionnaire de files d'attente peut réassembler les segments uniquement dans une unité de travail:

- Si l'appel MQGET est effectué dans une unité de travail définie par l'utilisateur, cette unité de travail est utilisée. Si l'appel échoue lors du processus de réassemblage, le gestionnaire de files d'attente rétablit dans la file d'attente tous les segments qui ont été supprimés lors du réassemblage. Toutefois, l'échec n'empêche pas la validation de l'unité de travail.
- Si l'appel est effectué en dehors d'une unité de travail définie par l'utilisateur et qu'il n'existe aucune unité de travail définie par l'utilisateur, le gestionnaire de files d'attente crée une unité de travail pour la durée de l'appel. Si l'appel aboutit, le gestionnaire de files d'attente valide automatiquement l'unité d'oeuvre (l'application n'a pas besoin de le faire). Si l'appel échoue, le gestionnaire de files d'attente annule l'unité d'oeuvre.
- Si l'appel fonctionne en dehors d'une unité d'oeuvre définie par l'utilisateur, mais qu'une unité d'oeuvre définie par l'utilisateur existe, le gestionnaire de files d'attente ne peut pas être

réassemblé. Si le message ne nécessite pas de réassemblage, l'appel peut tout de même aboutir. Mais si le message nécessite un réassemblage, l'appel échoue avec le code anomalie MQRC_UOW_NOT_AVAILABLE.

Pour les messages *non persistants* , le gestionnaire de files d'attente ne nécessite pas qu'une unité de travail soit disponible pour effectuer le réassemblage.

Chaque message physique qui est un segment possède son propre descripteur de message. Pour les segments constituant un message logique unique, la plupart des zones du descripteur de message sont identiques pour tous les segments du message logique ; en règle générale, seules les zones *MsgId*, *Offset*et *MsgFlags* diffèrent entre les segments du message logique. Toutefois, si un segment est placé dans une file d'attente de rebut au niveau d'un gestionnaire de files d'attente intermédiaire, le gestionnaire DLQ extrait le message en spécifiant l'option MQGMO_CONVERT , ce qui peut entraîner la modification du jeu de caractères ou du codage du segment. Si le gestionnaire de la file d'attente des messages non livrés envoie le segment en cours de route, le segment peut avoir un jeu de caractères ou un codage différent des autres segments du message logique lorsque le segment arrive sur le gestionnaire de files d'attente de destination.

Un message logique composé de segments dans lesquels les zones *CodedCharSetId* et *Encoding* diffèrent ne peut pas être réassemblé par le gestionnaire de files d'attente en un seul message logique. A la place, le gestionnaire de files d'attente réassemble et renvoie les premiers segments consécutifs au début du message logique ayant les mêmes identificateurs et codages de jeu de caractères, et l'appel MQGET se termine avec le code achèvement MQCC_WARNING et le code anomalie MQRC_INCONSISTENT_CCSIDS ou MQRC_INCONSISTENT_ENCODINGS, selon le cas. Cela se produit que MQGMO_CONVERT soit spécifié ou non. Pour extraire les segments restants, l'application doit réémettre l'appel MQGET sans l'option MQGMO_COMPLETE_MSG , en extrayant les segments un par un. MQGMO_LOGICAL_ORDER peut être utilisé pour extraire les segments restants dans l'ordre.

Une application qui insère des segments peut également définir d'autres zones dans le descripteur de message avec des valeurs qui diffèrent entre les segments. Toutefois, cela ne présente aucun avantage si l'application de réception utilise MQGMO_COMPLETE_MSG pour extraire le message logique. Lorsque le gestionnaire de files d'attente réassemble un message logique, il renvoie dans le descripteur de message les valeurs du descripteur de message pour le *premier* segment ; la seule exception est la zone *MsgFlags* , que le gestionnaire de files d'attente définit pour indiquer que le message réassemblé est le seul segment.

Si MQGMO_COMPLETE_MSG est spécifié pour un message de rapport, le gestionnaire de files d'attente effectue un traitement spécial. Le gestionnaire de files d'attente vérifie la file d'attente pour voir si tous les messages de rapport de ce type relatifs aux différents segments du message logique sont présents dans la file d'attente. Si tel est le cas, ils peuvent être extraits sous la forme d'un message unique en spécifiant MQGMO_COMPLETE_MSG. Pour que cela soit possible, les messages de rapport doivent être générés par un gestionnaire de files d'attente ou un agent MCA prenant en charge la segmentation, ou l'application d'origine doit demander au moins 100 octets de données de message (c'est-à-dire que les options MQRO_*_WITH_DATA ou MQRO_*_WITH_FULL_DATA appropriées doivent être spécifiées). Si la quantité de données d'application est inférieure à la quantité totale pour un segment, les octets manquants sont remplacés par des valeurs nulles dans le message de rapport renvoyé.

Si MQGMO_COMPLETE_MSG est spécifié avec MQGMO_MSG_UNDER_CURSOR ou MQGMO_BROWSE_MSG_UNDER_CURSOR, le curseur de navigation doit être positionné sur un message dont la zone *Offset* dans MQMD a la valeur 0. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC_INVALID_MSG_UNDER_CURSOR.

MQGMO_COMPLETE_MSG implique MQGMO_ALL_SEGMENTS_AVAILABLE, qui n'a donc pas besoin d'être spécifié.

MQGMO_COMPLETE_MSG peut être spécifié avec l'une des autres options MQGMO_* à l'exception de MQGMO_SYNCPOINT_IF_PERSISTENTet avec l'une des options MQMO_* à l'exception de MQMO_MATCH_OFFSET.

- Sous z/OS, cette option est prise en charge pour les files d'attente privées et partagées, mais la file d'attente doit avoir un type d'index MQIT_GROUP_ID. Pour les files d'attente partagées, l'objet CFSTRUCT que la mappe de files d'attente doit être au niveau CFLEVEL (3) ou CFLEVEL (4).
- Sous AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ MQI connectés à ces systèmes, cette option est prise en charge pour toutes les files d'attente locales.

MQGMO_ALL_MSGS_AVAILABLE
Les messages d'un groupe peuvent être extraits uniquement lorsque *tous* les messages du groupe sont disponibles. Si la file d'attente contient des groupes de messages dont certains des messages sont manquants (peut-être parce qu'ils ont été retardés sur le réseau et qu'ils ne sont pas encore arrivés), la spécification de MQGMO_ALL_MSGS_AVAILABLE empêche l'extraction des messages appartenant à des groupes incomplets. Toutefois, ces messages contribuent toujours à la valeur de l'attribut de file d'attente *CurrentQDepth* ; cela signifie qu'il peut n'y avoir aucun groupe de messages pouvant être extrait, même si *CurrentQDepth* est supérieur à zéro. Si aucun autre message n'est récupérable, le code raison MQRC_NO_MSG_AVAILABLE est renvoyé après l'expiration de l'intervalle d'attente indiqué (le cas échéant).

Le traitement de MQGMO_ALL_MSGS_AVAILABLE varie selon que MQGMO_LOGICAL_ORDER est également spécifié ou non:

- Si les deux options sont spécifiées, MQGMO_ALL_MSGS_AVAILABLE a un effet *uniquement* lorsqu'il n'y a pas de groupe en cours ou de message logique. S'il existe *un* groupe en cours ou un message logique, MQGMO_ALL_MSGS_AVAILABLE est ignoré. Cela signifie que MQGMO_ALL_MSGS_AVAILABLE peut rester en fonction lors du traitement des messages dans l'ordre logique.
- Si MQGMO_ALL_MSGS_AVAILABLE est spécifié sans MQGMO_LOGICAL_ORDER, MQGMO_ALL_MSGS_AVAILABLE *toujours* a un effet. Cela signifie que l'option doit être désactivée une fois que le premier message du groupe a été supprimé de la file d'attente, afin de pouvoir supprimer les messages restants du groupe.

L'exécution réussie d'un appel MQGET spécifiant MQGMO_ALL_MSGS_AVAILABLE signifie qu'au moment de l'émission de l'appel MQGET , tous les messages du groupe se trouvaient dans la file d'attente. Toutefois, sachez que d'autres applications peuvent toujours supprimer des messages du groupe (le groupe n'est pas verrouillé pour l'application qui extrait le premier message du groupe).

Si vous omettez cette option, les messages appartenant à des groupes peuvent être extraits même lorsque le groupe est incomplet.

MQGMO_ALL_MSGS_AVAILABLE implique MQGMO_ALL_SEGMENTS_AVAILABLE, qui n'a donc pas besoin d'être spécifié.

MQGMO_ALL_MSGS_AVAILABLE peut être spécifié avec l'une des autres options MQGMO_* et avec l'une des options MQMO_* .

- Sous z/OS, cette option est prise en charge pour les files d'attente privées et partagées, mais la file d'attente doit avoir un type d'index MQIT_GROUP_ID. Pour les files d'attente partagées, l'objet CFSTRUCT que la mappe de files d'attente doit être au niveau CFLEVEL (3) ou CFLEVEL (4).
- Sous AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ MQI connectés à ces systèmes, cette option est prise en charge pour toutes les files d'attente locales.

MQGMO_ALL_SEGMENTS_AVAILABLE
Les segments d'un message logique peuvent être extraits uniquement lorsque *tous* les segments du message logique sont disponibles. Si la file d'attente contient des messages segmentés dont certains des segments sont manquants (peut-être parce qu'ils ont été retardés dans le réseau et qu'ils ne sont pas encore arrivés), la spécification de MQGMO_ALL_SEGMENTS_AVAILABLE empêche l'extraction de segments appartenant à des messages logiques incomplets. Toutefois, ces segments contribuent toujours à la valeur de l'attribut de file d'attente *CurrentQDepth* ; cela signifie qu'il peut n'y avoir aucun message logique récupérable, même si *CurrentQDepth* est supérieur à zéro. Si aucun autre message n'est récupérable, le code raison MQRC_NO_MSG_AVAILABLE est renvoyé après l'expiration de l'intervalle d'attente indiqué (le cas échéant).

Le traitement de MQGMO_ALL_SEGMENTS_AVAILABLE varie selon que MQGMO_LOGICAL_ORDER est également spécifié ou non:

- Si les deux options sont spécifiées, MQGMO_ALL_SEGMENTS_AVAILABLE a un effet *uniquement* lorsqu'il n'y a pas de message logique en cours. S'il existe *un* message logique en cours, MQGMO_ALL_SEGMENTS_AVAILABLE est ignoré. Cela signifie que MQGMO_ALL_SEGMENTS_AVAILABLE peut rester en fonction lors du traitement des messages dans l'ordre logique.
- Si MQGMO_ALL_SEGMENTS_AVAILABLE est spécifié sans MQGMO_LOGICAL_ORDER, MQGMO_ALL_SEGMENTS_AVAILABLE *toujours* a un effet. Cela signifie que l'option doit être désactivée une fois que le premier segment du message logique a été supprimé de la file d'attente, afin de pouvoir supprimer les segments restants du message logique.

Si cette option n'est pas spécifiée, les segments de message peuvent être extraits même lorsque le message logique est incomplet.

Alors que MQGMO_COMPLETE_MSG et MQGMO_ALL_SEGMENTS_AVAILABLE exigent que tous les segments soient disponibles avant que l'un d'eux puisse être extrait, le premier renvoie le message complet, tandis que le second permet d'extraire les segments un par un.

Si MQGMO_ALL_SEGMENTS_AVAILABLE est spécifié pour un message de rapport, le gestionnaire de files d'attente vérifie la file d'attente pour voir s'il existe au moins un message de rapport pour chacun des segments qui constituent le message logique complet. Si tel est le cas, la condition MQGMO_ALL_SEGMENTS_AVAILABLE est satisfaite. Toutefois, le gestionnaire de files d'attente ne vérifie pas le *type* des messages de rapport présents, de sorte qu'il peut y avoir un mélange de types de rapport dans les messages de rapport relatifs aux segments du message logique. Par conséquent, la réussite de MQGMO_ALL_SEGMENTS_AVAILABLE n'implique pas que MQGMO_COMPLETE_MSG réussira. S'il existe une combinaison de types de rapport pour les segments d'un message logique particulier, ces messages de rapport doivent être extraits un par un.

Vous pouvez spécifier MQGMO_ALL_SEGMENTS_AVAILABLE avec l'une des autres options MQGMO_* et avec l'une des options MQMO_* .

- Sous z/OS, cette option est prise en charge pour les files d'attente privées et partagées, mais la file d'attente doit avoir un type d'index MQIT_GROUP_ID. Pour les files d'attente partagées, l'objet CFSTRUCT que la mappe de files d'attente doit être au niveau CFLEVEL (3) ou CFLEVEL (4).
- Sous AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ MQI connectés à ces systèmes, cette option est prise en charge pour toutes les files d'attente locales.

Options de propriétés : Les options suivantes sont liées aux propriétés du message :

MQGMO_PROPERTIES_AS_Q_DEF

Les propriétés du message, à l'exception de celles contenues dans le descripteur de message (ou l'extension), doivent être représentées comme défini par l'attribut de file d'attente *PropertyControl* . Si un *MsgHandle* est fourni, cette option est ignorée et les propriétés du message sont disponibles via *MsgHandle*, sauf si la valeur de l'attribut de file d'attente *PropertyControl* est MQPROP_FORCE_MQRFH2.

Il s'agit de l'action par défaut si aucune option de propriétés n'est spécifiée.

MQGMO_PROPERTIES_IN_HANDLE

Les propriétés du message doivent être mises à disposition via le *MsgHandle*. Si aucun descripteur de message n'est fourni, l'appel échoue avec le code anomalie MQRC_HMSG_ERROR.

Remarque : Si le message est lu ultérieurement par une application qui ne crée pas de descripteur de message, le gestionnaire de files d'attente place les propriétés de message dans une structure MQRFH2 . Vous pouvez constater que la présence d'un en-tête MQRFH2 inattendu perturbe le comportement d'une application existante.

MQGMO_NO_PROPERTIES

Aucune propriété du message, à l'exception de celles contenues dans le descripteur de message (ou l'extension), ne sera extraite. Si un *MsgHandle* est fourni, il est ignoré.

MQGMO_PROPERTIES_FORCE_MQRFH2

Les propriétés du message, à l'exception de celles contenues dans le descripteur de message (ou l'extension), doivent être représentées à l'aide d'en-têtes MQRFH2 . Cela permet d'assurer la compatibilité avec une version antérieure pour les applications qui s'attendent à extraire des propriétés mais qui ne peuvent pas être modifiées pour utiliser des descripteurs de message. Si un *MsgHandle* est fourni, il est ignoré.

MQGMO_PROPERTIES_COMPATIBILITY

Si le message contient une propriété avec le préfixe **"mcd."**, **"jms."**, **"usr."**ou **"mqext."**, toutes les propriétés de message sont distribuées à l'application dans un en-tête MQRFH2 . Sinon, toutes les propriétés du message, à l'exception de celles du descripteur de message (ou extension), sont supprimées et ne sont plus accessibles à l'application.

Option par défaut: Si aucune des options décrites n'est requise, l'option suivante peut être utilisée:

MQGMO_NONE

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut. MQGMO_NONE aide la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

La valeur initiale de la zone *Options* est MQGMO_NO_WAIT plus MQGMO_PROPERTIES_AS_Q_DEF.

Reserved1 (MQCHAR)

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc. Cette zone est ignorée si *Version* est inférieur à MQGMO_VERSION_2.

Reserved2 (MQLONG)

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc. Cette zone est ignorée si *Version* est inférieur à **MQGMO_VERSION_4**.

ResolvedQName (MQCHAR48)

Il s'agit d'une zone de sortie que le gestionnaire de files d'attente définit sur le nom local de la file d'attente à partir de laquelle le message a été extrait, comme défini pour le gestionnaire de files d'attente local. Ce nom est différent du nom utilisé pour ouvrir la file d'attente si:

- Une file d'attente alias a été ouverte (auquel cas, le nom de la file d'attente locale dans laquelle l'alias a été résolu est renvoyé), ou
- Une file d'attente modèle a été ouverte (auquel cas, le nom de la file d'attente locale dynamique est renvoyé).

La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

ReturnedLength (MQLONG)

Il s'agit d'une zone de sortie que le gestionnaire de files d'attente définit sur la longueur en octets des données de message renvoyées par l'appel MQGET dans le paramètre *Buffer* . Si le gestionnaire de files d'attente ne prend pas en charge cette fonction, *ReturnedLength* est défini sur la valeur MQRL_UNDEFINED.

Lorsque des messages sont convertis entre des codages ou des jeux de caractères, les données de message peuvent parfois changer de taille. Au retour de l'appel MQGET:

- Si *ReturnedLength* n'est *pas* MQRL_UNDEFINED, le nombre d'octets de données de message renvoyés est donné par *ReturnedLength*.
- Si *ReturnedLength* a la valeur MQRL_UNDEFINED, le nombre d'octets de données de message renvoyés est généralement indiqué par la plus petite des valeurs *BufferLength* et *DataLength*, mais peut être *inférieur à* si l'appel MQGET se termine avec le code anomalie MQRC_TRUNCATED_MSG_ACCEPTED. Dans ce cas, les octets non significatifs du paramètre *Buffer* sont définis sur des valeurs nulles.

La valeur spéciale suivante est définie:

MQRL_UNDEFINED
Longueur des données renvoyées non définie.

Sous z/OS, la valeur renvoyée pour la zone *ReturnedLength* est toujours MQRL_UNDEFINED.

La valeur initiale de cette zone est MQRL_UNDEFINED. Cette zone est ignorée si *Version* est inférieur à MQGMO_VERSION_3.

Segmentation (MQCHAR)

Il s'agit d'un indicateur qui indique si une segmentation supplémentaire est autorisée pour le message extrait. Elle a l'une des valeurs suivantes :

MQSEG_INHIBÉ
Segmentation non autorisée.

MQSEG_AUTORISÉ_
Segmentation autorisée.

Sous z/OS, le gestionnaire de files d'attente définit toujours cette zone sur MQSEG_INHIBÉE.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est MQSEG_INHIBÉE. Cette zone est ignorée si *Version* est inférieur à MQGMO_VERSION_2.

SegmentStatus (MQCHAR)

Il s'agit d'un indicateur qui indique si le message extrait est un segment d'un message logique. Elle a l'une des valeurs suivantes :

MQSS_NON_SEGMENT
Le message n'est pas un segment.

SEGMENT_MQSS_SEGMENT
Le message est un segment, mais n'est pas le dernier segment du message logique.

SEGMENT_LAST_MQSS_
Le message est le dernier segment du message logique.

Il s'agit également de la valeur renvoyée si le message logique se compose d'un seul segment.

Sous z/OS, le gestionnaire de files d'attente définit toujours cette zone sur MQSS_NOT_A_SEGMENT.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est MQSS_NOT_A_SEGMENT. Cette zone est ignorée si *Version* est inférieur à MQGMO_VERSION_2.

Signal1 (MQLONG)

Il s'agit d'une zone d'entrée qui est utilisée uniquement avec l'option MQGMO_SET_SIGNAL ; elle identifie un signal à distribuer lorsqu'un message est disponible.

Remarque : Le type de données et l'utilisation de cette zone sont déterminés par l'environnement ; pour cette raison, les applications que vous souhaitez porter entre différents environnements ne doivent pas utiliser de signaux.

- Sous z/OS, cette zone doit contenir l'adresse d'un bloc de contrôle d'événement (ECB). L'ECB doit être effacé par l'application avant que l'appel MQGET ne soit émis. La mémoire contenant le ECB ne doit pas être libérée tant que la file d'attente n'est pas fermée. La commande ECB est envoyée par le gestionnaire de files d'attente avec l'un des codes d'achèvement de signal décrits. Ces codes

achèvement sont définis dans les bits 2 à 31 de l'ECB, la zone définie dans la macro de mappage z/OS IHAECB pour un code achèvement utilisateur.

- Dans tous les autres environnements, il s'agit d'une zone réservée ; sa valeur n'est pas significative.

Les codes achèvement du signal sont les suivants:

MQEC_MSG_ARRIVÉ
Un message approprié est arrivé dans la file d'attente. Ce message n'a pas été réservé à l'appelant ; une deuxième demande MQGET doit être émise, mais une autre application peut extraire le message avant que la deuxième demande ne soit effectuée.

MQEC_INTERVALLE_ATTENTE_EXPIRÉE
Le *WaitInterval* spécifié a expiré sans qu'un message approprié n'arrive.

MQEC_ATTENTE_ANNULE
L'attente a été annulée pour une raison indéterminée (telle que l'arrêt du gestionnaire de files d'attente ou la désactivation de la file d'attente). Emettez à nouveau la demande si vous souhaitez un diagnostic plus approfondi.

MQEC_Q_MGR_QUIESCING
L'attente a été annulée car le gestionnaire de files d'attente est passé à l'état de mise au repos (MQGMO_FAIL_IF_QUIESCING a été spécifié sur l'appel MQGET).

MQEC_CONNEXION_MISE au repos
L'attente a été annulée car la connexion est entrée dans l'état de mise au repos (MQGMO_FAIL_IF_QUIESCING a été spécifié dans l'appel MQGET).

La valeur initiale de cette zone est déterminée par l'environnement:

- Sous z/OS, la valeur initiale est le pointeur null.
- Dans tous les autres environnements, la valeur initiale est `0`.

Signal2 (MQLONG)

Il s'agit d'une zone d'entrée utilisée uniquement avec l'option MQGMO_SET_SIGNAL. Il s'agit d'une zone réservée ; sa valeur n'est pas significative.

La valeur initiale de cette zone est `0`.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure. La valeur doit être:

ID_STRUC_MQGMO_
Identificateur de la structure des options get-message.

Pour le langage de programmation C, la constante MQGMO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQGMO_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQGMO_STRUC_ID.

Version (MQLONG)
La version est le numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

MQGMO_VERSION_1
Structure des options get-message Version-1 .

Cette version est prise en charge dans tous les environnements.

MQGMO_VERSION_2
Structure des options d'obtention de message Version-2 .

Cette version est prise en charge dans tous les environnements.

MQGMO_VERSION_3
Structure des options get-message Version-3 .

Cette version est prise en charge dans tous les environnements.

MQGMO_VERSION_4
Structure des options get-message Version-4 .

Cette version est prise en charge dans tous les environnements.

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

MQGMO_CURRENT_VERSION
Version actuelle de la structure des options get-message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQGMO_VERSION_1.

WaitInterval (MQLONG)

Il s'agit de la durée approximative, exprimée en millisecondes, pendant laquelle l'appel MQGET attend l'arrivée d'un message approprié (c'est-à-dire un message satisfaisant aux critères de sélection spécifiés dans le paramètre *MsgDesc* de l'appel MQGET.

.

Important : Il n'y a pas d'attente, ni de délai, si un message approprié est disponible immédiatement.

Voir la zone *MsgId* décrite dans «MQMD-Descripteur de message», à la page 394 pour plus de détails). Si aucun message approprié n'est arrivé après ce délai, l'appel se termine avec MQCC_FAILED et le code anomalie MQRC_NO_MSG_AVAILABLE.

Sous z/OS, la période pendant laquelle l'appel MQGET attend réellement est affectée par les considérations relatives au chargement du système et à la planification des travaux et peut varier entre la valeur spécifiée pour *WaitInterval* et une valeur supérieure d'environ 250 millisecondes à *WaitInterval*.

WaitInterval est utilisé avec l'option MQGMO_WAIT ou MQGMO_SET_SIGNAL. Elle est ignorée si aucune de ces deux options n'est spécifiée. Si l'une de ces valeurs est indiquée, *WaitInterval* doit être supérieur ou égal à zéro, ou la valeur spéciale suivante:

MQWI_ILLIMITÉ
Intervalle d'attente illimité.

La valeur initiale de cette zone est 0.

Valeurs initiales et déclarations de langue pour MQGMO

Tableau 508. Valeurs initiales des zones dans MQGMO pour MQGMO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQGMO_	'GMO¬'
Version	MQGMO_VERSION_1	1
Options	MQGMO_NO_WAIT	0
WaitInterval	Aucun	0
Signal1	Aucun	Pointeur null sur z/OS; 0 dans le cas contraire
Signal2	Aucun	0
ResolvedQName	Aucun	Chaîne nulle ou blancs

Tableau 508. Valeurs initiales des zones dans MQGMO pour MQGMO (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
MatchOptions	MQMO_MATCH_MSG_ID + MQMO_MATCH_CORREL_ID	3
GroupStatus	MQGS_NOT_IN_GROUPE	'¬'
SegmentStatus	MQSS_NON_SEGMENT	'¬'
Segmentation	MQSEG_INHIBÉ	'¬'
Reserved1	Aucun	'¬'
MsgToken	MQMTOK_NONE	Valeurs NULL
ReturnedLength	MQRL_UNDEFINED	-1
Reserved2	Aucun	'¬'
MsgHandle	MQHM_NONE	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQGMO_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQGMO MyGMO = {MQGMO_DEFAULT};
```

Déclaration C

```
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 */
};
```

- Sous z/OS, la zone *Signal1* est déclarée comme PMQLONG.

Déclaration COBOL

```
**   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.
```

- Sous z/OS, la zone *Signal1* est déclarée comme POINTER.

Déclaration PL/I

```
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 */
```

- Sous z/OS, la zone *Signal1* est déclarée comme pointer.

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-en-tête d'informations IMS

Le tableau suivant récapitule les zones de la structure.

Tableau 509. Zones dans MQIIH

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
StrucLength	Longueur de la structure MQIIH	StrucLength
Encoding	Réservé	Codage
CodedCharSetId	Réservé	CodedCharSetId

Tableau 509. Zones dans MQIIH (suite)		
Zone	**Description**	**Topic**
Format	Nom de format MQ des données qui suivent MQIIH	Format
Flags	Indicateurs	Indicateurs
LTermOverride	Substitution de terminal logique	LTermOverride
MFSMapName	Nom de mappe des services de structuration de messages	MFSMapName
ReplyToFormat	Nom de format MQ du message de réponse	FormatReplyTo
Authenticator	Mot de passe ou mot de passe™ RACF	Authentificateur
TranInstanceId	Identificateur d'instance de transaction	IDTranInstance
TranState	Etat de la transaction	TranState
CommitMode	Mode de validation	CommitMode
SecurityScope	Etendue de la sécurité	SecurityScope
Reserved	Réservé	Reserved

Présentation de MQIIH

Disponibilité: Tous les systèmes WebSphere MQ et les clients WebSphere MQ .

Objectif: La structure MQIIH décrit les informations qui doivent être présentes au début d'un message envoyé au pont IMS via WebSphere MQ for z/OS.

Nom de format: MQFMT_IMS.

Jeu de caractères et codage: des conditions spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQIIH et les données de message d'application:

- Les applications qui se connectent au gestionnaire de files d'attente qui possède la file d'attente de pont IMS doivent fournir une structure MQIIH qui se trouve dans le jeu de caractères et le codage du gestionnaire de files d'attente. En effet, la conversion de données de la structure MQIIH n'est pas effectuée dans ce cas.
- Les applications qui se connectent à d'autres gestionnaires de files d'attente peuvent fournir une structure MQIIH qui se trouve dans l'un des jeux de caractères et codages pris en charge ; l'agent de canal de message de réception connecté au gestionnaire de files d'attente qui possède la file d'attente de pont IMS convertit le MQIIH.
- Les données de message d'application qui suivent la structure MQIIH doivent être dans le même jeu de caractères et le même codage que la structure MQIIH. N'utilisez pas les zones *CodedCharSetId* et *Encoding* de la structure MQIIH pour spécifier le jeu de caractères et le codage des données de message d'application.

  Vous devez fournir un exit de conversion de données pour convertir les données de message d'application si les données ne sont pas l'un des formats intégrés pris en charge par le gestionnaire de files d'attente.

Zones pour MQIIH

La structure MQIIH contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

Authentificateur (MQCHAR8)

Il s'agit du mot de passe RACF ou PassTicket. Il est facultatif ; s'il est spécifié, il est utilisé avec l'ID utilisateur dans le contexte de sécurité MQMD pour générer un UTOKEN envoyé à IMS afin de fournir un

contexte de sécurité. S'il n'est pas spécifié, l'ID utilisateur est utilisé sans vérification. Cela dépend des paramètres des commutateurs RACF , qui peuvent nécessiter la présence d'un authentificateur.

Cette valeur est ignorée si le premier octet est vide ou null. La valeur spéciale suivante peut être utilisée:

MQIAUT_AUCUN
Pas d'authentification.

Pour le langage de programmation C, la constante MQIAUT_NONE_ARRAY est également définie ; elle a la même valeur que MQIAUT_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ_AUTHENTICATOR_LENGTH. La valeur initiale de cette zone est MQIAUT_NONE.

CodedCharSetId (MQLONG)

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

L'ID de jeu de caractères pour les structures prises en charge qui suivent une structure MQIIH est identique à celui de la structure MQIIH elle-même et provient de tout en-tête MQ précédent.

CommitMode (MQCHAR)

Il s'agit du mode de validation IMS . Pour plus d'informations sur les modes de validation IMS , voir *OTMA Reference* . Il doit s'agir de l'une des valeurs suivantes :

MQICM_COMMIT_THEN_SEND
Validez puis envoyez.

Ce mode implique une double mise en file d'attente de la sortie, mais des temps d'occupation de région plus courts. Les transactions rapides et conversationnelles ne peuvent pas s'exécuter avec ce mode.

MQICM_SEND_THEN_COMMIT
Envoyer puis valider.

Toute transaction IMS lancée suite à une validation mpde de MQICM_SEND_THEN_COMMIT s'exécute en mode RESPONSE, quelle que soit la façon dont la transaction est définie dans la définition du système IMS (paramètre MSGTYPE dans la macro TRANSACT). Cela s'applique également aux transactions initiées au moyen d'un commutateur de transaction.

La valeur initiale de cette zone est MQICM_COMMIT_THEN_SEND.

Codage (MQLONG)

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est 0.

Le codage des structures prises en charge qui suivent une structure MQIIH est identique à celui de la structure MQIIH elle-même et provient de tout en-tête MQ précédent.

Indicateurs (MQLONG)

La valeur des indicateurs doit être:

MQIIH_AUCUN
Aucun indicateur.

MQIIH_PASS_EXPIRATION
Le message de réponse contient:

- Les mêmes options de rapport d'expiration que le message de demande
- Délai d'expiration restant à partir du message de demande sans ajustement effectué pour le temps de traitement du pont

Si cette valeur n'est pas définie, le délai d'expiration est défini sur *unlimited*.

MQIIH_REPLY_FORMAT_NONE
Définit MQIIH.Format de la réponse à MQFMT_NONE.

MQIIH_IGNORE_PURG
Définit l'indicateur TMAMIPRG dans le préfixe OTMA, qui demande à OTMA d'ignorer les appels PURG sur le PCB TP pour les transactions CM0 .

MQIIH_CM0_REQUEST_RESPONSE
Pour les transactions en mode de validation 0 (CM0), cet indicateur définit l'indicateur TMAMHRSP dans le préfixe OTMA. La définition de cet indicateur demande à OTMA/IMS de générer un message DFS2082 RESPONSE MODE TRANSACTION TERMINATED WITHOUT REPLY lorsque le programme d'application IMS d'origine ne répond pas au bloc de commande d'E-S ni ne bascule vers une autre transaction.

La valeur initiale de cette zone est MQIIH_NONE.

Format (MQCHAR8)

Indique le nom de format MQ des données qui suivent la structure MQIIH.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La longueur de cette zone est indiquée par MQ_FORMAT_LENGTH. La valeur initiale de cette zone est MQFMT_NONE.

LTermOverride (MQCHAR8)

Remplacement du terminal logique, placé dans la zone IO PCB. Il est facultatif ; s'il n'est pas spécifié, le nom TPIPE est utilisé. Elle est ignorée si le premier octet est vide ou null.

La longueur de cette zone est indiquée par MQ_LTERM_OVERRIDE_LENGTH. La valeur initiale de cette zone est de 8 caractères blancs.

MFSMapName (MQCHAR8)

Nom de mappe des services de format de message, placé dans la zone IO PCB. Il est facultatif. En entrée, il représente le MID, en sortie, il représente le MOD. Il est ignoré si le premier octet est vide ou nul.

La longueur de cette zone est indiquée par MQ_MFS_MAP_NAME_LENGTH. La valeur initiale de cette zone est de 8 caractères blancs.

Format ReplyTo(MQCHAR8)

Il s'agit du nom de format MQ du message de réponse envoyé en réponse au message en cours. La longueur de cette zone est indiquée par MQ_FORMAT_LENGTH. La valeur initiale de cette zone est MQFMT_NONE.

Pour convertir les données du message de réponse à l'aide de MQGMO_CONVERT, spécifiez MQIIH.replyToFormat= MQFMT_STRING ou MQIIH.replyToFormat= MQFMT_IMS_VAR_STRING. Pour plus d'informations sur l'utilisation de ces zones, voir «Format (MQCHAR8)», à la page 410.

Si la valeur par défaut (MQIIH.replyToFormat= MQFMT_NONE) est utilisée sur le message de demande et que le message de réponse est extrait à l'aide de MQGMO_CONVERT, aucune conversion de données n'est effectuée.

Réservé (MQCHAR)

Il s'agit d'une zone réservée ; elle doit être vide.

SecurityScope (MQCHAR)

Indique le traitement de sécurité IMS requis. Les valeurs suivantes sont définies :

CONTROLE MQISS_CHECK
Vérifier la portée de la sécurité: un ACEE est généré dans la région de contrôle, mais pas dans la région dépendante.

MQISS_FULL
Portée de sécurité complète: un ACEE mis en cache est généré dans la région de contrôle et un ACEE non mis en cache est généré dans la région dépendante. Si vous utilisez MQISS_FULL, vérifiez que l'ID utilisateur pour lequel ACEE est généré a accès aux ressources utilisées dans la région dépendante.

Si ni MQISS_CHECK ni MQISS_FULL n'est spécifié pour cette zone, MQISS_CHECK est utilisé par défaut.

La valeur initiale de cette zone est MQISS_CHECK.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure. La valeur doit être:

ID MQIIH_STRUC_ID
Identificateur de la structure d'en-tête d'informations IMS .

Pour le langage de programmation C, la constante MQIIH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQIIH_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQIIH_STRUC_ID.

StrucLength (MQLONG)

Longueur de la structure MQIIH. La valeur doit être:

MQIIH_LENGTH_1
Longueur de la structure d'en-tête d'informations IMS .

La valeur initiale de cette zone est MQIIH_LENGTH_1.

ID TranInstance(MQBYTE16)

Il s'agit de l'identificateur de l'instance de transaction. Cette zone est utilisée par les messages de sortie d' IMS. Par conséquent, elle est ignorée lors de la première entrée. Si vous définissez *TranState* sur MQITS_IN_CONVERSATION, cette valeur doit être fournie dans l'entrée suivante, ainsi que dans toutes les entrées suivantes, pour permettre à IMS de corréler les messages à la conversation correcte. Vous pouvez utiliser la valeur spéciale suivante:

MQITII_AUCUN
Aucun identificateur d'instance de transaction.

Pour le langage de programmation C, la constante MQITII_NONE_ARRAY est également définie ; elle a la même valeur que MQITII_NONE, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

La longueur de cette zone est indiquée par MQ_TRAN_INSTANCE_ID_LENGTH. La valeur initiale de cette zone est MQITII_NONE.

TranState (MQCHAR)

Indique l'état de la conversation IMS . Elle est ignorée lors de la première entrée car aucune conversation n'existe. Sur les entrées suivantes, il indique si une conversation est active ou non. En sortie, il est défini par IMS. Il doit s'agir de l'une des valeurs suivantes :

MQITS_DANS_CONVERSATION
En conversation.

MQITS_NON_CONVERSATION
Pas en conversation.

MQITS_ARCHITECTURE
Renvoie les données d'état de transaction sous forme structurée.

Cette valeur est utilisée uniquement avec la commande IMS `/DISPLAY TRAN` . Elle renvoie les données d'état de transaction dans la forme structurée IMS au lieu de la forme alphanumérique.

La valeur initiale de cette zone est MQITS_NOT_IN_CONVERSATION.

Version (MQLONG)

Il s'agit du numéro de version de la structure. La valeur doit être:

MQIIH_VERSION_1
Numéro de version de la structure d'en-tête d'informations IMS .

La constante suivante indique le numéro de version de la version en cours:

MQIIH_VERSION_CURRENT_
Version actuelle de la structure d'en-tête d'informations IMS .

La valeur initiale de cette zone est MQIIH_VERSION_1.

Valeurs initiales et déclarations de langue pour MQIIH

Tableau 510. Valeurs initiales des zones dans MQIIH pour MQIIH

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID MQIIH_STRUC_ID	'IIH?'
Version	MQIIH_VERSION_1	1
StrucLength	MQIIH_LENGTH_1	84
Encoding	Aucun	0
CodedCharSetId	Aucun	0
Format	MQFMT_AUCUN	Espaces vides
Flags	MQIIH_AUCUN	0
LTermOverride	Aucun	Espaces vides
MFSMapName	Aucun	Espaces vides
ReplyToFormat	MQFMT_AUCUN	Espaces vides
Authenticator	MQIAUT_AUCUN	Espaces vides
TranInstanceId	MQITII_AUCUN	Valeurs NULL
TranState	MQITS_NON_CONVERSATION	'?'
CommitMode	MQICM_COMMIT_THEN_SEND	'0'
SecurityScope	CONTROLE MQISS_CHECK	'C'
Reserved	Aucun	'?'

Remarques :

1. Le symbole? représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQIIH_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQIIH MyIIH = {MQIIH_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Options de propriété de message d'interrogation

Le tableau suivant récapitule les zones de la structure. Structure MQIMPO-Options de propriété de message d'interrogation

Tableau 511. Zones dans MQIMPO

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options contrôlant l'action de MQINQMP	Options
RequestedEncoding	Codage dans lequel la propriété inquired doit être convertie	RequestedEncoding
RequestedCCSID	Jeu de caractères de la propriété inquired	RequestedCCSID
ReturnedEncoding	Codage de la valeur renvoyée	ReturnedEncoding
ReturnedCCSID	Jeu de caractères de la valeur renvoyée	ReturnedCCSID
Reserved1	Réservé, zone	ReturnedCCSID

Tableau 511. Zones dans MQIMPO (suite)		
Zone	**Description**	**Topic**
ReturnedName	Nom de la propriété interrogée	ReturnedName
TypeString	Représentation sous forme de chaîne du type de données de la propriété	TypeString

Présentation de MQIMPO

Structure des options d'interrogation des propriétés de message.

Disponibilité: Tous les systèmes WebSphere MQ et les clients WebSphere MQ .

Objectif: La structure MQIMPO permet aux applications de spécifier des options qui contrôlent la façon dont les propriétés des messages sont recherchées. La structure est un paramètre d'entrée dans l'appel MQINQMP.

Jeu de caractères et codage: les données de MQIMPO doivent être dans le jeu de caractères de l'application et le codage de l'application (MQENC_NATIVE).

Zones pour MQIMPO

Structure des options d'interrogation des propriétés de message-zones

La structure MQIMPO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

Options (MQLONG)
Structure des options d'interrogation des propriétés de message-Zone Options

Les options suivantes contrôlent l'action de MQINQMP. Vous pouvez spécifier une ou plusieurs de ces options et, si vous en avez besoin, les valeurs peuvent être:

- Ajouté ensemble (ne pas ajouter la même constante plus d'une fois), ou
- combinées par le biais de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations par bit).

Les combinaisons d'options non valides sont notées ; toutes les autres combinaisons sont valides.

Options de données de valeur: Les options suivantes concernent le traitement des données de valeur lorsque la propriété est extraite du message.

MQIMPO_CONVERT_VALUE
Cette option demande que la valeur de la propriété soit convertie pour être conforme aux valeurs *RequestedCCSID* et *RequestedEncoding* spécifiées avant que l'appel MQINQMP ne renvoie la valeur de la propriété dans la zone *Value* .

- Si la conversion aboutit, les zones *ReturnedCCSID* et *ReturnedEncoding* sont définies sur les mêmes valeurs que *RequestedCCSID* et *RequestedEncoding* en cas de retour de l'appel MQINQMP.
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur, la valeur de la propriété est renvoyée non convertie.

  Si la propriété est une chaîne, les zones *ReturnedCCSID* et *ReturnedEncoding* sont définies sur le jeu de caractères et le codage de la chaîne non convertie.

  Le code achèvement est MQCC_WARNING dans ce cas, avec le code anomalie MQRC_PROP_VALUE_NOT_CONVERTED. Le curseur de propriété est avancé vers la propriété renvoyée.

Si la valeur de la propriété se développe lors de la conversion et dépasse la taille du paramètre *Value* , la valeur est renvoyée non convertie, avec le code achèvement MQCC_FAILED ; le code anomalie est défini sur MQRC_PROPERTY_VALEUR_TOO_BIG.

Le paramètre *DataLength* de l'appel MQINQMP renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la mémoire tampon requise pour prendre en charge la valeur de propriété convertie. Le curseur de propriété est inchangé.

Cette option demande également que:

- Si le nom de la propriété contient un caractère générique, et
- La zone *ReturnedName* est initialisée avec une adresse ou un décalage pour le nom renvoyé,

le nom renvoyé est converti pour être conforme aux valeurs *RequestedCCSID* et *RequestedEncoding* .

- Si la conversion aboutit, la zone *VSCCSID* de *ReturnedName* et le codage du nom renvoyé sont définis sur la valeur d'entrée *RequestedCCSID* et *RequestedEncoding* .
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur ni avertissement, le nom renvoyé n'est pas converti. Le code achèvement est MQCC_WARNING dans ce cas, avec le code anomalie MQRC_PROP_NAME_NOT_CONVERTIS.

  Le curseur de propriété est avancé vers la propriété renvoyée. MQRC_PROP_VALEUR_NOT_CONVERTED est renvoyé si la valeur et le nom ne sont pas convertis.

Si le nom renvoyé se développe lors de la conversion et dépasse la taille de la zone *VSBufsize* de la *RequestedName*, la chaîne renvoyée reste non convertie, avec le code achèvement MQCC_FAILED et le code anomalie défini sur MQRC_PROPERTY_NAME_TOO_BIG.

La zone *VSLength* de la structure MQCHARV renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la mémoire tampon requise pour accueillir la valeur de propriété convertie. Le curseur de propriété est inchangé.

MQIMPO_TYPE_CONVERT

Cette option demande que la valeur de la propriété soit convertie de son type de données en cours en type de données spécifié dans le paramètre *Type* de l'appel MQINQMP.

- Si la conversion aboutit, le paramètre *Type* est inchangé lors du renvoi de l'appel MQINQMP.
- Si la conversion échoue, mais que l'appel MQINQMP se termine sans erreur, l'appel échoue avec la raison MQRC_PROP_CONV_NOT_SUPPORTED. Le curseur de propriété est inchangé.

Si la conversion du type de données entraîne l'extension de la valeur lors de la conversion et que la valeur convertie dépasse la taille du paramètre *Value* , la valeur est renvoyée non convertie avec le code achèvement MQCC_FAILED et le code anomalie est défini sur MQRC_PROPERTY_VALEUR_TOO_BIG.

Le paramètre *DataLength* de l'appel MQINQMP renvoie la longueur à laquelle la valeur de propriété aurait été convertie, afin de permettre à l'application de déterminer la taille de la mémoire tampon requise pour prendre en charge la valeur de propriété convertie. Le curseur de propriété est inchangé.

Si la valeur du paramètre *Type* de l'appel MQINQMP n'est pas valide, l'appel échoue avec la raison MQRC_PROPERTY_TYPE_ERROR.

Si la conversion de type de données demandée n'est pas prise en charge, l'appel échoue avec la raison MQRC_PROP_CONV_NOT_SUPPORTED. Les conversions de type de données suivantes sont prises en charge:

Type de données de propriété	Types de données cible pris en charge
MQTYPE_BOOLÉEN	MQTYPE_STRING, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_BYTE_STRING	MQTYPE_CHAINE
MQTYPE_INT8	MQTYPE_STRING, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_INT16	MQTYPE_STRING, MQTYPE_INT32, MQTYPE_INT64

Type de données de propriété	Types de données cible pris en charge
MQTYPE_INT32	MQTYPE_STRING, MQTYPE_INT64
MQTYPE_INT64	MQTYPE_CHAINE
MQTYPE_FLOAT32	MQTYPE_STRING, MQTYPE_FLOAT64
MQTYPE_FLOAT64	MQTYPE_CHAINE
MQTYPE_CHAINE	MQTYPE_BOOLEAN, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64, MQTYPE_FLOAT32, MQTYPE_FLOAT64
MQTYPE_NULL	Aucun

Les règles générales régissant les conversions prises en charge sont les suivantes:

- Les valeurs de propriété numérique peuvent être converties d'un type de données à un autre, à condition qu'aucune donnée ne soit perdue lors de la conversion.

  Par exemple, la valeur d'une propriété avec le type de données MQTYPE_INT32 peut être convertie en valeur avec le type de données MQTYPE_INT64, mais ne peut pas être convertie en valeur avec le type de données MQTYPE_INT16.
- Une valeur de propriété de tout type de données peut être convertie en chaîne.
- Une valeur de propriété de type chaîne peut être convertie en tout autre type de données, du moment que la chaîne est correctement formatée pour la conversion. Si une application tente de convertir une valeur de propriété de chaîne qui n'est pas formatée correctement, WebSphere MQ renvoie le code anomalie MQRC_PROP_NUMBER_FORMAT_ERROR.
- Si une application tente une conversion qui n'est pas prise en charge, WebSphere MQ renvoie le code anomalie MQRC_PROP_CONV_NOT_SUPPORTED.

Les règles spécifiques pour la conversion d'une valeur de propriété d'un type de données à un autre sont les suivantes:

- Lors de la conversion d'une valeur de propriété MQTYPE_BOOLEAN en chaîne, la valeur TRUE est convertie en chaîne "TRUE" et la valeur false est convertie en chaîne "FALSE".
- Lors de la conversion d'une valeur de propriété MQTYPE_BOOLEAN en un type de données numérique, la valeur TRUE est convertie en un et la valeur FALSE est convertie en zéro.
- Lors de la conversion d'une valeur de propriété de chaîne en valeur MQTYPE_BOOLEAN, la chaîne "TRUE", ou "1", est convertie en TRUE, et la chaîne "FALSE", ou "0", est convertie en FALSE.

  Notez que les termes "TRUE" et "FALSE" ne sont pas sensibles à la casse.

  Toute autre chaîne ne peut pas être convertie ; WebSphere MQ renvoie le code anomalie MQRC_PROP_NUMBER_FORMAT_ERROR.
- Lors de la conversion d'une valeur de propriété de chaîne en valeur avec le type de données MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32 ou MQTYPE_INT64, la chaîne doit avoir le format suivant:

  ```
  [blanks][sign]digits
  ```

  Les significations des composants de la chaîne sont les suivantes:

  blanks
  Caractères blancs de début facultatifs

  sign
  Caractère facultatif de signe plus (+) ou moins (-).

  digits
  Séquence contiguë de caractères numériques (0-9). Au moins un chiffre doit être présent.

Après la séquence de caractères numériques, la chaîne peut contenir d'autres caractères autres que des chiffres, mais la conversion s'arrête au premier caractère de ce type. La chaîne est supposée représenter un entier décimal.

WebSphere MQ renvoie le code anomalie MQRC_PROP_NUMBER_FORMAT_ERROR si la chaîne n'est pas correctement formatée.

- Lors de la conversion d'une valeur de propriété de chaîne en valeur avec le type de données MQTYPE_FLOAT32 ou MQTYPE_FLOAT64, la chaîne doit avoir le format suivant:

```
[blanks][sign]digits[.digits][e_char[e_sign]e_digits]
```

Les significations des composants de la chaîne sont les suivantes:

blanks
Caractères blancs de début facultatifs

sign
Caractère facultatif de signe plus (+) ou moins (-).

digits
Séquence contiguë de caractères numériques (0-9). Au moins un chiffre doit être présent.

e_char
Un caractère exposant, qui est soit "E", soit "e".

e_sign
Signe plus (+) ou signe moins (-) facultatif pour l'exposant.

e_digits
Séquence contiguë de caractères numériques (0-9) pour l'exposant. Au moins un chiffre doit être présent si la chaîne contient un caractère exposant.

Après la séquence de chiffres, la chaîne peut contenir d'autres caractères autres que des chiffres, mais la conversion s'arrête au premier caractère de ce type. La chaîne est supposée représenter un nombre décimal en virgule flottante avec un exposant puissance de 10.

WebSphere MQ renvoie le code anomalie MQRC_PROP_NUMBER_FORMAT_ERROR si la chaîne n'est pas correctement formatée.

- Lors de la conversion d'une valeur de propriété numérique en chaîne, la valeur est convertie en représentation de chaîne de la valeur sous forme de nombre décimal, et non en chaîne contenant le caractère ASCII pour cette valeur. Par exemple, l'entier 65 est converti en chaîne "65", et non en chaîne "A".
- Lors de la conversion d'une valeur de propriété de chaîne d'octets en chaîne, chaque octet est converti en deux caractères hexadécimaux représentant l'octet. Par exemple, le tableau d'octets {0xF1, 0x12, 0x00, 0xFF} est converti en chaîne "F11200FF".

LONGUEUR_REQUÊTE_MQIMPO_LONG
Interrogez le type et la longueur de la valeur de propriété. La longueur est renvoyée dans le paramètre *DataLength* de l'appel MQINQMP. La valeur de la propriété n'est pas renvoyée.

Si une mémoire tampon *ReturnedName* est spécifiée, la zone *VSLength* de la structure MQCHARV est renseignée avec la longueur du nom de propriété. Le nom de propriété n'est pas renvoyé.

Options d'itération: Les options suivantes concernent l'itération sur les propriétés, à l'aide d'un nom avec un caractère générique

MQIMPO_INQ_FIRST
Renseignez-vous sur la première propriété qui correspond au nom spécifié. Après cet appel, un curseur est établi sur la propriété renvoyée.

Il s'agit de la valeur par défaut.

L'option MQIMPO_INQ_PROP_UNDER_CURSOR peut ensuite être utilisée avec un appel MQINQMP, si nécessaire, pour demander à nouveau la même propriété.

Notez qu'il n'y a qu'un seul curseur de propriété ; par conséquent, si le nom de propriété, spécifié dans l'appel MQINQMP, change le curseur est réinitialisé.

Cette option n'est pas valide avec l'une des options suivantes:

MQIMPO_INQ_NEXT
MQIMPO_INQ_PROP_UNDER_CURSOR

MQIMPO_INQ_NEXT
Demande sur la propriété suivante qui correspond au nom spécifié, en poursuivant la recherche à partir du curseur de propriété. Le curseur est avancé à la propriété qui est renvoyée.

S'il s'agit du premier appel MQINQMP pour le nom spécifié, la première propriété correspondant au nom spécifié est renvoyée.

L'option MQIMPO_INQ_PROP_UNDER_CURSOR peut ensuite être utilisée avec un appel MQINQMP, si nécessaire, pour demander à nouveau la même propriété.

Si la propriété sous le curseur a été supprimée, MQINQMP renvoie la propriété correspondante suivante après celle qui a été supprimée.

Si une propriété qui correspond au caractère générique est ajoutée alors qu'une itération est en cours, elle peut ou non être renvoyée lors de la fin de l'itération. La propriété est renvoyée une fois que l'itération redémarre à l'aide de MQIMPO_INQ_FIRST.

Une propriété correspondant au caractère générique qui a été supprimé alors que l'itération était en cours n'est pas renvoyée après sa suppression.

Cette option n'est pas valide avec l'une des options suivantes:

MQIMPO_INQ_FIRST
MQIMPO_INQ_PROP_UNDER_CURSOR

MQIMPO_INQ_PROP_UNDER_CURSOR
Extrayez la valeur de la propriété pointée par le curseur de propriété. La propriété pointée par le curseur de propriété est celle qui a été interrogée pour la dernière fois, à l'aide de l'option MQIMPO_INQ_FIRST ou MQIMPO_INQ_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé, lorsque le descripteur de message est spécifié dans la zone *MsgHandle* du MQGMO sur un appel MQGET, ou lorsque le descripteur de message est spécifié dans les zones *OriginalMsgHandle* ou *NewMsgHandle* de la structure MQPMO sur un appel MQPUT.

Si cette option est utilisée lorsque le curseur de propriété n'a pas encore été établi ou si la propriété pointée par le curseur de propriété a été supprimée, l'appel échoue avec le code achèvement MQCC_FAILED et la raison MQRC_PROPERTY_NOT_AVAILABLE.

Cette option n'est pas valide avec l'une des options suivantes:

MQIMPO_INQ_FIRST
MQIMPO_INQ_NEXT

Si aucune des options décrites précédemment n'est requise, l'option suivante peut être utilisée:

MQIMPO_NONE
Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

MQIMPO_NONE aide la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQIMPO_INQ_FIRST.

RequestedCCSID (MQLONG)
Structure des options d'interrogation des propriétés de message-Zone RequestedCCSID

Jeu de caractères dans lequel la valeur de la propriété recherchée doit être convertie si la valeur est une chaîne de caractères. Il s'agit également du jeu de caractères dans lequel le *ReturnedName* doit être converti lorsque MQIMPO_CONVERT_VALUE ou MQIMPO_CONVERT_TYPE est spécifié.

La valeur initiale de cette zone est MQCCSI_APPL.

RequestedEncoding (MQLONG)
Interroger la structure des options de propriété de message-Zone RequestedEncoding

Il s'agit du codage dans lequel la valeur de propriété interrogée doit être convertie lorsque MQIMPO_CONVERT_VALUE ou MQIMPO_CONVERT_TYPE est spécifié.

La valeur initiale de cette zone est MQENC_NATIVE.

Reserved1 (MQCHAR)

Il s'agit d'une zone réservée. La valeur initiale de cette zone est un caractère blanc (zone de 4 octets).

ReturnedCCSID (MQLONG)
Interroger la structure des options de propriété de message-Zone ReturnedCCSID

En sortie, il s'agit du jeu de caractères de la valeur renvoyée si le paramètre *Type* de l'appel MQINQMP est MQTYPE_STRING.

Si l'option MQIMPO_CONVERT_VALUE est spécifiée et que la conversion a abouti, la zone *ReturnedCCSID* , en retour, est identique à la valeur transmise.

La valeur initiale de cette zone est zéro.

ReturnedEncoding (MQLONG)
Consulter la structure des options de propriété de message- ReturnedEncoding

En sortie, il s'agit du codage de la valeur renvoyée.

Si l'option MQIMPO_CONVERT_VALUE est spécifiée et que la conversion a abouti, la zone *ReturnedEncoding* , en retour, est identique à la valeur transmise.

La valeur initiale de cette zone est MQENC_NATIVE.

ReturnedName (MQCHARV)
Interroger la structure des options de propriété de message-Zone ReturnedName

Nom réel de la propriété interrogée.

En entrée, une mémoire tampon de chaîne peut être transmise à l'aide de la zone *VSPtr* ou *VSOffset* de la structure MQCHARV . La longueur de la mémoire tampon de chaîne est spécifiée à l'aide de la zone *VSBufsize* de la structure MQCHARV.

Lors du retour de l'appel MQINQMP, la mémoire tampon de la chaîne est terminée avec le nom de la propriété qui a été interrogée, à condition que la mémoire tampon de la chaîne soit suffisamment longue pour contenir complètement le nom. La zone *VSLength* de la structure MQCHARV est renseignée avec la longueur du nom de propriété. La zone *VSCCSID* de la structure MQCHARV est renseignée pour indiquer le jeu de caractères du nom renvoyé, que la conversion du nom ait échoué ou non.

Il s'agit d'une zone d'entrée/sortie. La valeur initiale de cette zone est MQCHARV_DEFAULT.

StrucId (MQCHAR4)
Interroger la structure des options de propriété de message-Zone StrucId

Il s'agit de l'identificateur de structure. La valeur doit être:

ID_STRUC_MQIMPO_
Identificateur de la structure des options d'interrogation des propriétés de message.

Pour le langage de programmation C, la constante MQIMPO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQIMPO_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQIMPO_STRUC_ID.

TypeString (MQCHAR8)
Structure des options d'interrogation des propriétés de message-Zone TypeString

Représentation sous forme de chaîne du type de données de la propriété.

Si la propriété a été spécifiée dans un en-tête MQRFH2 et que l'attribut MQRFH2 `dt` n'est pas reconnu, cette zone peut être utilisée pour déterminer le type de données de la propriété. *TypeString* est renvoyé dans le jeu de caractères codés 1208 (UTF-8) et correspond aux huit premiers octets de la valeur de l'attribut `dt` de la propriété dont la reconnaissance a échoué.

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est la chaîne nulle dans le langage de programmation C et 8 caractères blancs dans les autres langages de programmation.

Version (MQLONG)
Consulter la structure des options de propriété de message-Zone Version

Il s'agit du numéro de version de la structure. La valeur doit être:

MQIMPO_VERSION_1
Numéro de version de la structure des options de propriété de message d'interrogation.

La constante suivante indique le numéro de version de la version en cours:

MQIMPO_CURRENT_VERSION
Version actuelle de la structure des options de propriété de message d'interrogation.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQIMPO_VERSION_1.

Valeurs initiales et déclarations de langage pour MQIMPO
Structure des options d'interrogation des propriétés de message-Valeurs initiales

Tableau 512. Valeurs initiales des zones dans MQIPMO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQIMPO_	'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_VALEUR PAR DEFAUT	
TypeString	Chaîne nulle ou blancs	

Tableau 512. Valeurs initiales des zones dans MQIPMO (suite)

Nom de zone	Nom de la constante	Valeur de la constante

Remarques :

1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
2. Dans le langage de programmation C, la variable macroMQIMPO_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQIMPO MyIMPO = {MQIMPO_DEFAULT};
```

Déclaration C
Consulter la structure des options de propriété de message-Déclaration de langage C

```
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 */
};
```

Déclaration COBOL
Structure des options d'interrogation des propriétés de message-Déclaration de langage COBOL

```
**   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.
```

Déclaration PL/I
Interroger la structure des options de propriété de message-Déclaration de langage PL/I

```
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 */
```

Déclaration High Level Assembler
Structure des options d'interrogation des propriétés de message-Déclaration du langage assembleur

```
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-Descripteur de message

Le tableau suivant récapitule les zones de la structure.

Tableau 513. Zones dans MQMD

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Report	Options des messages de rapport	Rapport
MsgType	Type de message	MsgType
Expiry	Durée de vie des messages	MQMD-Zone d'expiration
Feedback	Commentaires en retour ou code anomalie	MQMD-Zone de commentaire en retour

Tableau 513. Zones dans MQMD (suite)		
Zone	**Description**	**Topic**
Encoding	Codage numérique des données de message	Codage
CodedCharSetId	Identificateur de jeu de caractères des données de message	CodedCharSetId
Format	Nom de format des données de message	Format
Priority	Priorité de message	Priorité
Persistence	Persistance des messages	Persistance
MsgId	Identificateur de message	MQMD-Zone MsgId
CorrelId	Identificateur de corrélation	CorrelId
BackoutCount	Nombre d'annulations	BackoutCount
ReplyToQ	Nom de la file d'attente de réponses	ReplyToQ
ReplyToQMgr	Nom du gestionnaire de files d'attente de réponses	ReplyToQMgr
UserIdentifier	Identificateur utilisateur	UserIdentifier
AccountingToken	Jeton de comptabilité	AccountingToken
ApplIdentityData	Données d'application relatives à l'identité	ApplIdentityData
PutApplType	Type de l'application ayant placé le message en file d'attente	PutApplType
PutApplName	Nom de l'application ayant placé le message en file d'attente	PutApplName
PutDate	Date à laquelle le message a été placé en file d'attente	PutDate
PutTime	Heure à laquelle le message a été placé en file d'attente	PutTime
ApplOriginData	Données d'application relatives à l'origine	ApplOriginData
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQMD_VERSION_2.		
GroupId	Identificateur de groupe	GroupId
MsgSeqNumber	Numéro de séquence du message logique dans le groupe	MsgSeqNumber
Offset	Décalage des données du message physique à partir du début du message logique	offset
MsgFlags	Indicateurs de message	MQMD - Zone MsgFlags
OriginalLength	Longueur du message d'origine	OriginalLength

Présentation de MQMD

Disponibilité: Tous les systèmes WebSphere MQ , plus WebSphere MQ clients MQI connectés à ces systèmes.

Objectif: La structure MQMD contient les informations de contrôle qui accompagnent les données d'application lorsqu'un message circule entre les applications d'envoi et de réception. La structure est un paramètre d'entrée/sortie sur les appels MQGET, MQPUT et MQPUT1 .

Version: la version actuelle de MQMD est MQMD_VERSION_2. Les applications destinées à être portables entre plusieurs environnements doivent garantir que la version requise de MQMD est prise en charge dans tous les environnements concernés. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQMD prise en charge par l'environnement, mais avec la valeur initiale de la zone *Version* définie sur MQMD_VERSION_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone *Version* sur le numéro de version de la version requise.

Une déclaration pour la structure version-1 est disponible avec le nom MQMD1.

Jeu de caractères et codage: les données dans MQMD doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local ; ces données sont fournies par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client WebSphere MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Si les gestionnaires de files d'attente d'envoi et de réception utilisent des jeux de caractères ou des codages différents, les données de MQMD sont converties automatiquement. Il n'est pas nécessaire que l'application convertisse le MQMD.

Utilisation de différentes versions de MQMD: Un MQMD version-2 équivaut à utiliser un MQMD version-1 et à préfixer les données de message avec une structure MQMDE. Toutefois, si toutes les zones de la structure MQMDE ont leurs valeurs par défaut, MQMDE peut être omis. Une instance version-1 de MQMD plus MQMDE est utilisée comme décrit ci-après:

- Dans les appels MQPUT et MQPUT1 , si l'application fournit un MQMD version-1 , l'application peut éventuellement préfixer les données de message avec un MQMDE, en définissant la zone *Format* de MQMD sur MQFMT_MD_EXTENSION pour indiquer qu'un MQMDE est présent. Si l'application ne fournit pas de MQMDE, le gestionnaire de files d'attente utilise les valeurs par défaut pour les zones du MQMDE.

  Remarque : Plusieurs des zones qui existent dans le MQMD version-2 mais pas dans le MQMD version-1 sont des zones d'entrée/sortie sur les appels MQPUT et MQPUT1 . Toutefois, le gestionnaire de files d'attente ne renvoie *aucune* valeur dans les zones équivalentes de MQMDE dans la sortie des appels MQPUT et MQPUT1 ; si l'application requiert ces valeurs de sortie, elle doit utiliser un MQMD version-2 .
- Dans l'appel MQGET, si l'application fournit un MQMD version-1 , le gestionnaire de files d'attente préfixe le message renvoyé avec un MQMDE, mais uniquement si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut. La zone *Format* de MQMD aura la valeur MQFMT_MD_EXTENSION pour indiquer qu'un MQMDE est présent.

Les valeurs par défaut utilisées par le gestionnaire de files d'attente pour les zones de MQMDE sont identiques aux valeurs initiales de ces zones, indiquées dans le Tableau 518, à la page 451.

Lorsqu'un message se trouve dans une file d'attente de transmission, certaines zones de MQMD sont définies sur des valeurs particulières ; pour plus de détails, voir «MQXQH-en-tête de file d'attente de transmission», à la page 601 .

Contexte de message: Certaines zones de MQMD contiennent le contexte de message. Il existe deux types de contexte de message: *contexte d'identité* et *contexte d'origine*. Généralement :

- Le contexte d'identité est lié à l'application qui a *initialement* inséré le message
- Le contexte d'origine est lié à l'application qui a *le plus récemment* inséré le message.

Ces deux applications peuvent être la même application, mais elles peuvent également être des applications différentes (par exemple, lorsqu'un message est transmis d'une application à une autre).

Bien que le contexte d'identité et d'origine aient généralement la signification décrite, le contenu des deux types de zones de contexte dans MQMD dépend des options MQPMO_ * _CONTEXT qui sont spécifiées lors de l'insertion du message. Par conséquent, le contexte d'identité ne se rapporte pas nécessairement à l'application qui a initialement inséré le message, et le contexte d'origine ne se

rapporte pas nécessairement à l'application qui a inséré le message le plus récemment ; il dépend de la conception de la suite d'applications.

L'agent MCA ne modifie jamais le contexte de message. Les agents MCA qui reçoivent des messages des gestionnaires de files d'attente éloignées utilisent l'option de contexte MQPMO_SET_ALL_CONTEXT dans l'appel MQPUT ou MQPUT1 . Cela permet à l'agent MCA récepteur de conserver exactement le contexte de message qui a été transmis avec le message à partir de l'agent MCA émetteur. Toutefois, il en résulte que le contexte d'origine ne se rapporte à aucun des agents MCA qui ont envoyé et reçu le message. Le contexte d'origine fait référence à une application antérieure qui a inséré le message. Si toutes les applications intermédiaires ont passé le contexte de message, le contexte d'origine fait référence à l'application d'origine elle-même.

Dans les descriptions, les zones de contexte sont décrites comme si elles étaient utilisées comme décrit précédemment. Pour plus d'informations sur le contexte de message, voir Contexte de message.

Zones pour MQMD

La structure MQMD contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

Tableau 514. Zones dans MQMD

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Report	Options des messages de rapport	Rapport
MsgType	Type de message	MsgType
Expiry	Durée de vie des messages	MQMD-Zone d'expiration
Feedback	Commentaires en retour ou code anomalie	MQMD-Zone de commentaire en retour
Encoding	Codage numérique des données de message	Codage
CodedCharSetId	Identificateur de jeu de caractères des données de message	CodedCharSetId
Format	Nom de format des données de message	Format
Priority	Priorité de message	Priorité
Persistence	Persistance des messages	Persistance
MsgId	Identificateur de message	MQMD-Zone MsgId
CorrelId	Identificateur de corrélation	CorrelId
BackoutCount	Nombre d'annulations	BackoutCount
ReplyToQ	Nom de la file d'attente de réponses	ReplyToQ
ReplyToQMgr	Nom du gestionnaire de files d'attente de réponses	ReplyToQMgr
UserIdentifier	Identificateur utilisateur	UserIdentifier
AccountingToken	Jeton de comptabilité	AccountingToken
ApplIdentityData	Données d'application relatives à l'identité	ApplIdentityData
PutApplType	Type de l'application ayant placé le message en file d'attente	PutApplType

Tableau 514. Zones dans MQMD (suite)		
Zone	**Description**	**Topic**
PutApplName	Nom de l'application ayant placé le message en file d'attente	PutApplName
PutDate	Date à laquelle le message a été placé en file d'attente	PutDate
PutTime	Heure à laquelle le message a été placé en file d'attente	PutTime
ApplOriginData	Données d'application relatives à l'origine	ApplOriginData
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQMD_VERSION_2.		
GroupId	Identificateur de groupe	GroupId
MsgSeqNumber	Numéro de séquence du message logique dans le groupe	MsgSeqNumber
Offset	Décalage des données du message physique à partir du début du message logique	offset
MsgFlags	Indicateurs de message	MQMD - Zone MsgFlags
OriginalLength	Longueur du message d'origine	OriginalLength

AccountingToken (MQBYTE32)

Il s'agit du jeton de comptabilité, qui fait partie du **contexte d'identité** du message. Pour plus d'informations sur le contexte de message, voir «Présentation de MQMD», à la page 395; voir aussi Contexte de message.

AccountingToken permet à une application de facturer de manière appropriée le travail effectué à la suite du message. Le gestionnaire de files d'attente traite ces informations comme une chaîne de bits et ne vérifie pas leur contenu.

Le gestionnaire de files d'attente génère ces informations comme suit:

- Le premier octet de la zone est défini sur la longueur des informations comptables présentes dans les octets qui suivent ; cette longueur est comprise entre zéro et 30, et est stockée dans le premier octet sous la forme d'un entier binaire.
- Le second octet et les octets suivants (tels que spécifiés par la zone de longueur) sont définis sur les informations de comptabilité appropriées à l'environnement.
  - Sous z/OS , les informations de comptabilité sont définies sur:
    - Pour le lot z/OS , les informations de comptabilité de la carte de travail JES ou d'une instruction JES ACCT dans la carte EXEC (les séparateurs de virgules sont remplacés par X'FF'). Ces informations sont tronquées, si nécessaire, à 31 octets.
    - Pour TSO, numéro de compte de l'utilisateur.
    - Pour CICS, l'identificateur d'unité de travail (UEPUOWDS) de l'unité logique 6.2 (26 octets).
    - Pour IMS, nom de bloc de spécification de programme à 8 caractères concaténé avec le jeton de récupération IMS à 16 caractères.
  - Sous IBM i, les informations de comptabilité sont définies sur le code de comptabilité du travail.
  - Sur les systèmes UNIX , les informations de comptabilité sont définies sur l'ID utilisateur numérique, en caractères ASCII.
  - Sous Windows, les informations de comptabilité sont définies sur un identificateur de sécurité (SID) Windows dans un format compressé. Le SID identifie de manière unique l'identificateur d'utilisateur stocké dans la zone *UserIdentifier* . Lorsque le SID est stocké dans la zone *AccountingToken* ,

l'autorité d'identification de 6 octets (située dans le troisième octet et les suivants du SID) est omise. Par exemple, si le SID Windows a une longueur de 28 octets, 22 octets d'informations SID sont stockés dans la zone *AccountingToken* .

- Le dernier octet (octet 32) de la zone de comptabilité est défini sur le type de jeton de comptabilité (dans ce cas, MQACTT_NT_SECURITY_ID, x'0b'):

  MQACTT_CICS_LUOW_ID
  Identificateur CICS LUOW.

  MQACTT_NT_SECURITY_ID
  Identificateur de sécurité Windows .

  MQACTT_OS400_ACCOUNT_TOKEN
  Jeton de comptabilité IBM i .

  MQACTT_UNIX_NUMERIC_ID
  Identificateur numérique des systèmes UNIX .

  MQACTT_UTILISATEUR
  Jeton de comptabilité défini par l'utilisateur.

  MQACTT_INCONNU
  Type de jeton de comptabilité inconnu.

  Le type de jeton de comptabilité est défini sur une valeur explicite uniquement dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus les clients WebSphere MQ MQI connectés à ces systèmes. Dans d'autres environnements, le type de jeton de comptabilité est défini sur la valeur MQACTT_UNKNOWN. Dans ces environnements, utilisez la zone *PutApplType* pour déduire le type de jeton de comptabilité reçu.

- Tous les autres octets sont définis sur zéro binaire.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO_SET_IDENTITY_CONTEXT ou MQPMO_SET_ALL_CONTEXT est spécifié dans le paramètre *PutMsgOpts* . Si ni MQPMO_SET_IDENTITY_CONTEXT ni MQPMO_SET_ALL_CONTEXT n'est spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement. Pour plus d'informations sur le contexte de message, voir Contexte de message .

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le *AccountingToken* qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de *AccountingToken* qui est conservée avec le message (voir la description de MQPMO_RETAIN dans«Options MQPMO (MQLONG)», à la page 485 pour plus de détails sur les publications conservées) mais qui n'est pas utilisée comme valeur *AccountingToken* lorsque le message est envoyé en tant que publication aux abonnés car elle fournit une valeur pour remplacer *AccountingToken* dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement zéro binaire.

Il s'agit d'une zone de sortie pour l'appel MQGET.

Cette zone n'est soumise à aucune conversion basée sur le jeu de caractères du gestionnaire de files d'attente ; elle est traitée comme une chaîne de bits et non comme une chaîne de caractères.

Le gestionnaire de files d'attente ne fait rien avec les informations de cette zone. L'application doit interpréter les informations si elle souhaite les utiliser à des fins comptables.

Vous pouvez utiliser la valeur spéciale suivante pour la zone *AccountingToken* :

MQACT_AUCUN
Aucun jeton de comptabilité n'est spécifié.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQACT_NONE_ARRAY est également définie ; elle a la même valeur que MQACT_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ_ACCOUNTING_TOKEN_LENGTH. La valeur initiale de cette zone est MQACT_NONE.

Données ApplIdentity(MQCHAR32)

Il fait partie du **contexte d'identité** du message. Pour plus d'informations sur le contexte de message, voir «Présentation de MQMD», à la page 395 et Contexte de message .

ApplIdentityData est une information définie par la suite d'applications et peut être utilisée pour fournir des informations supplémentaires sur le message ou son émetteur. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Lorsque le gestionnaire de files d'attente génère ces informations, elles sont entièrement vides.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO_SET_IDENTITY_CONTEXT ou MQPMO_SET_ALL_CONTEXT est spécifié dans le paramètre *PutMsgOpts* . Si un caractère null est présent, les caractères null et les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si ni MQPMO_SET_IDENTITY_CONTEXT ni MQPMO_SET_ALL_CONTEXT n'est spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement. Pour plus d'informations sur le contexte de message, voir Contexte de message.

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le *ApplIdentityData* qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de *ApplIdentityData* qui est conservée avec le message (voir la description de MQPMO_RETAIN pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que *ApplIdentityData* lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer *ApplIdentityData* dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ_APPL_IDENTITY_DATA_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 32 caractères blancs dans les autres langages de programmation.

Données ApplOrigin(MQCHAR4)

Il fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir «Présentation de MQMD», à la page 395 et Contexte de message .

ApplOriginData est une information définie par la suite d'applications qui peut être utilisée pour fournir des informations supplémentaires sur l'origine du message. Par exemple, elle peut être définie par des applications s'exécutant avec des droits utilisateur appropriés pour indiquer si les données d'identité sont dignes de confiance.

Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Lorsque le gestionnaire de files d'attente génère ces informations, elles sont entièrement vides.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO_SET_ALL_CONTEXT est spécifié dans le paramètre *PutMsgOpts* . Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le gestionnaire de files d'attente convertit le caractère null et les caractères suivants en blancs. Si MQPMO_SET_ALL_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ_APPL_ORIGIN_DATA_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 4 caractères blancs dans les autres langages de programmation.

Lorsque le message est publié, bien que *ApplOriginData* soit défini, il est vide dans l'abonnement qu'il reçoit.

BackoutCount (MQLONG)

Il s'agit du nombre de fois où le message a été précédemment renvoyé par l'appel MQGET dans le cadre d'une unité de travail, puis annulé. Il aide l'application à détecter les erreurs de traitement basées sur le contenu des messages. Le nombre exclut les appels MQGET qui spécifient l'une des options MQGMO_BROWSE_ *.

La précision de ce nombre est affectée par l'attribut de file d'attente *HardenGetBackout* ; voir «Attributs des files d'attente», à la page 824.

Sous z/OS, la valeur 255 signifie que le message a été annulé 255 fois ou plus ; la valeur renvoyée n'est jamais supérieure à 255.

Il s'agit d'une zone de sortie pour l'appel MQGET. Il est ignoré pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est 0.

CodedCharSetId (MQLONG)
Cette zone indique l'identificateur de jeu de caractères des données de type caractère dans le corps du message.

Remarque : Les données de type caractère dans MQMD et les autres structures de données MQ qui sont des paramètres sur les appels doivent figurer dans le jeu de caractères du gestionnaire de files d'attente. Il est défini par l'attribut *CodedCharSetId* du gestionnaire de files d'attente ; pour plus de détails sur cet attribut, voir «Attributs du gestionnaire de files d'attente», à la page 787 .

Si cette zone est définie sur MQCCSI_Q_MGR lors de l'appel de MQGET avec MQGMO_CONVERT dans les options, le comportement est différent entre les applications client et serveur. Pour les applications serveur, la page de codes utilisée pour la conversion de caractères est la *CodedCharSetId* du gestionnaire de files d'attente ; pour les applications client, la page de codes utilisée pour la conversion de caractères est la page de codes de l'environnement local en cours.

Pour les applications client, MQCCSI_Q_MGR est renseigné en fonction de l'environnement local du client et non de celui du gestionnaire de files d'attente. L'exception à cette règle est lorsque vous placez un message dans une file d'attente IMS Bridge ; ce qui est renvoyé, dans la zone *CodedCharSetId* de MQMD, est le CCSID du gestionnaire de files d'attente.

Vous ne devez pas utiliser la valeur spéciale suivante:

MQCCSI_APPL
Il en résulte une valeur incorrecte dans la zone *CodedCharSetId* du MQMD et un code retour de MQRC_SOURCE_CCSID_ERROR (ou MQRC_FORMAT_ERROR pour z/OS) lorsque le message est reçu à l'aide de l'appel MQGET avec l'option MQGMO_CONVERT.

Vous pouvez utiliser les valeurs spéciales suivantes:

MQCCSI_Q_DIR
Les données de type caractère du message se trouvent dans le jeu de caractères du gestionnaire de files d'attente.

Dans les appels MQPUT et MQPUT1 , le gestionnaire de files d'attente modifie cette valeur dans le MQMD qui est envoyé avec le message avec l'identificateur de jeu de caractères true du gestionnaire de files d'attente. Par conséquent, la valeur MQCCSI_Q_MGR n'est jamais renvoyée par l'appel MQGET.

MQCCSI_DEFAULT
Le *CodedCharSetId* des données de la zone *String* est défini par la zone *CodedCharSetId* dans la structure d'en-tête qui précède la structure MQCFH, ou par la zone *CodedCharSetId* dans le MQMD si le MQCFH est au début du message.

MQCCSI_HÉRITER
Les données de type caractère du message se trouvent dans le même jeu de caractères que cette structure ; il s'agit du jeu de caractères du gestionnaire de files d'attente. (Pour MQMD uniquement, MQCCSI_INHERIT a la même signification que MQCCSI_Q_MGR).

Le gestionnaire de files d'attente remplace cette valeur dans le MQMD envoyé avec le message par l'identificateur de jeu de caractères réel de MQMD. Si aucune erreur ne se produit, la valeur MQCCSI_INHERIT n'est pas renvoyée par l'appel MQGET.

N'utilisez pas MQCCSI_INHERIT si la valeur de la zone *PutApplType* dans MQMD est MQAT_BROKER.

MQCCSI_IMBRIQUÉ
Les données de type caractère du message se trouvent dans un jeu de caractères avec l'identificateur contenu dans les données du message lui-même. Il peut y avoir n'importe quel nombre d'identificateurs de jeu de caractères imbriqués dans les données de message, s'appliquant à différentes parties des données. Cette valeur doit être utilisée pour les messages PCF (au format MQFMT_ADMIN, MQFMT_EVENT ou MQFMT_PCF) qui contiennent des données dans un mélange de jeux de caractères. Chaque structure MQCFST, MQCFSL et MQCFSF contenue dans le message PCF doit avoir un identificateur de jeu de caractères explicite spécifié et non MQCCSI_DEFAULT.

Si un message au format MQFMT_EMBEDDED_PCF doit contenir des données dans une combinaison de jeux de caractères, n'utilisez pas MQCCSI_EMBEDDED. A la place, définissez MQEPH_CCSID_EMBEDDED dans la zone Flags de la structure MQEPH. Cela revient à définir MQCCSI_EMBEDDED dans la structure précédente. Chaque structure MQCFST, MQCFSL et MQCFSF contenue dans le message PCF doit alors avoir un identificateur de jeu de caractères explicite spécifié et non MQCCSI_DEFAULT. Pour plus d'informations sur la structure MQEPH, voir «MQEPH-en-tête PCF imbriqué», à la page 340.

Indiquez cette valeur uniquement sur les appels MQPUT et MQPUT1 . S'il est spécifié dans l'appel MQGET, il empêche la conversion du message.

Dans les appels MQPUT et MQPUT1 , le gestionnaire de files d'attente modifie les valeurs MQCCSI_Q_MGR et MQCCSI_INHERIT dans le MQMD envoyé avec le message comme décrit ci-dessus, mais ne modifie pas le MQMD spécifié dans l'appel MQPUT ou MQPUT1 . Aucune autre vérification n'est effectuée sur la valeur spécifiée.

Les applications qui extraient des messages doivent comparer cette zone à la valeur attendue par l'application ; si les valeurs diffèrent, l'application peut avoir besoin de convertir des données de type caractères dans le message.

Si vous spécifiez l'option MQGMO_CONVERT dans l'appel MQGET, cette zone est une zone d'entrée-sortie. La valeur indiquée par l'application est l'identificateur de jeu de caractères codés vers lequel convertir les données de message si nécessaire. Si la conversion aboutit ou n'est pas nécessaire, la valeur reste inchangée (sauf que la valeur MQCCSI_Q_MGR ou MQCCSI_INHERIT est convertie en valeur réelle). Si la conversion échoue, la valeur après l'appel MQGET représente l'identificateur de jeu de caractères codés du message non converti renvoyé à l'application.

Sinon, il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQCCSI_Q_MGR.

CorrelId (MQBYTE24)
La zone CorrelId est une propriété de l'en-tête de message qui peut être utilisée pour identifier un message ou un groupe de messages spécifique.

Il s'agit d'une chaîne d'octets que l'application peut utiliser pour relier un message à un autre ou pour relier le message à un autre travail effectué par l'application. L'identificateur de corrélation est une propriété permanente du message et est conservé lors des redémarrages du gestionnaire de files d'attente. Etant donné que l'identificateur de corrélation est une chaîne d'octets et non une chaîne de caractères, l'identificateur de corrélation n'est *pas* converti entre les jeux de caractères lorsque le message est transmis d'un gestionnaire de files d'attente à un autre.

Pour les appels MQPUT et MQPUT1 , l'application peut spécifier n'importe quelle valeur. Le gestionnaire de files d'attente transmet cette valeur avec le message et la distribue à l'application qui émet la demande d'obtention du message.

Si l'application spécifie MQPMO_NEW_CORREL_ID, le gestionnaire de files d'attente génère un identificateur de corrélation unique qui est envoyé avec le message et renvoyé à l'application émettrice dans la sortie de l'appel MQPUT ou MQPUT1 .

Un identificateur de corrélation généré par le gestionnaire de files d'attente se compose d'un identificateur de produit de 3 octets (AMQ ou CSQ en ASCII ou EBCDIC), suivi d'un octet réservé et d'une implémentation spécifique du produit d'une chaîne unique. Dans WebSphere MQ , cette chaîne d'implémentation spécifique au produit contient les 12 premiers caractères du nom du gestionnaire de files d'attente et une valeur dérivée de l'horloge système. Tous les gestionnaires de files d'attente qui

peuvent intercommuniquer doivent donc avoir des noms qui diffèrent dans les 12 premiers caractères pour s'assurer que les identificateurs de message sont uniques. La possibilité de générer une chaîne unique dépend également du fait que l'horloge système n'est pas modifiée en amont. Pour éviter qu'un identificateur de message généré par le gestionnaire de files d'attente ne soit dupliqué par l'application, celle-ci doit éviter de générer des identificateurs dont les caractères initiaux sont compris entre A et I en ASCII ou EBCDIC (X'41'à X'49'et X'C1'à X'C9'). Toutefois, l'application n'est pas empêchée de générer des identificateurs avec des caractères initiaux dans ces plages.

Cet identificateur de corrélation généré est conservé avec le message s'il est conservé et il est utilisé comme identificateur de corrélation lorsque le message est envoyé en tant que publication aux abonnés qui spécifient MQCI_NONE dans la zone SubCorrelde l'ID MQSD transmis à l'appel MQSUB. Voir Options MQPMO pour plus de détails sur les publications conservées.

Lorsque le gestionnaire de files d'attente ou un agent MCA génère un message de rapport, il définit la zone *CorrelId* de la manière spécifiée par la zone *Report* du message d'origine, MQRO_COPY_MSG_ID_TO_CORREL_ID ou MQRO_PASS_CORREL_ID. Les applications qui génèrent des messages de rapport doivent également effectuer cette opération.

Pour l'appel MQGET, *CorrelId* est l'une des cinq zones qui peuvent être utilisées pour sélectionner un message particulier à extraire de la file d'attente. Voir la description de la zone *MsgId* pour plus de détails sur la manière de spécifier des valeurs pour cette zone.

La spécification de MQCI_NONE comme identificateur de corrélation a le même effet que la *non* spécification de MQMO_MATCH_CORREL_ID, c'est-à-dire que *tout* identificateur de corrélation correspond.

Si l'option MQGMO_MSG_UNDER_CURSOR est spécifiée dans le paramètre *GetMsgOpts* de l'appel MQGET, cette zone est ignorée.

En cas de retour d'un appel MQGET, la zone *CorrelId* est définie sur l'identificateur de corrélation du message renvoyé (le cas échéant).

Les valeurs spéciales suivantes peuvent être utilisées:

MQCI_NONE
Aucun identificateur de corrélation n'est indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQCI_NONE_ARRAY est également définie ; elle a la même valeur que MQCI_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQCI_NOUVEAU_SESSION
Le message est le début d'une nouvelle session.

Cette valeur est reconnue par le pont CICS comme indiquant le début d'une nouvelle session, c'est-à-dire le début d'une nouvelle séquence de messages.

Pour le langage de programmation C, la constante MQCI_NEW_SESSION_ARRAY est également définie ; elle a la même valeur que MQCI_NEW_SESSION, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Pour l'appel MQGET, il s'agit d'une zone d'entrée/sortie. Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée si MQPMO_NEW_CORREL_ID n'est *pas* spécifié et d'une zone de sortie si MQPMO_NEW_CORREL_ID *est* spécifié. La longueur de cette zone est indiquée par MQ_CORREL_ID_LENGTH. La valeur initiale de cette zone est MQCI_NONE.

Remarque :

Vous ne pouvez pas transmettre l'identificateur de corrélation d'une publication dans une hiérarchie. Cette zone est utilisée par le gestionnaire de files d'attente.

Codage (MQLONG)

Indique le codage numérique des données numériques dans le message ; il ne s'applique pas aux données numériques dans la structure MQMD elle-même. Le codage numérique définit la représentation utilisée pour les entiers binaires, les entiers décimaux condensés et les nombres à virgule flottante.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que la zone est valide. La valeur spéciale suivante est définie:

MQENC_NATIVE
Le codage est la valeur par défaut du langage de programmation et de la machine sur lesquels l'application s'exécute.

Remarque : La valeur de cette constante dépend du langage de programmation et de l'environnement. Pour cette raison, les applications doivent être compilées à l'aide des fichiers d'en-tête, de macro, de COPY ou d'INCLUDE appropriés à l'environnement dans lequel l'application sera exécutée.

Les applications qui placent des messages spécifient généralement MQENC_NATIVE. Les applications qui extraient des messages doivent comparer cette zone à la valeur MQENC_NATIVE; si les valeurs diffèrent, l'application peut avoir besoin de convertir des données numériques dans le message. Utilisez l'option MQGMO_CONVERT pour demander au gestionnaire de files d'attente de convertir le message dans le cadre du traitement de l'appel MQGET. Pour plus d'informations sur la façon dont la zone *Encoding* est construite, voir «Codages de machine», à la page 891 .

Si vous spécifiez l'option MQGMO_CONVERT dans l'appel MQGET, cette zone est une zone d'entrée-sortie. La valeur spécifiée par l'application est le codage vers lequel convertir les données de message si nécessaire. Si la conversion aboutit ou n'est pas nécessaire, la valeur reste inchangée. Si la conversion échoue, la valeur après l'appel MQGET représente le codage du message non converti renvoyé à l'application.

Dans d'autres cas, il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQENC_NATIVE.

Expiration (MQLONG)

Il s'agit d'une période de temps exprimée en dixièmes de seconde, définie par l'application qui insère le message. Le message peut être supprimé s'il n'a pas été supprimé de la file d'attente de destination avant l'expiration de ce délai.

La valeur est décrémentée pour refléter le temps passé par le message dans la file d'attente de destination, ainsi que dans les files d'attente de transmission intermédiaires si l'insertion est dans une file d'attente éloignée. Il peut également être décrémenté par les agents de canal de message pour refléter les temps de transmission, si ceux-ci sont significatifs. De même, une application qui transmet ce message à une autre file d'attente peut décrémenter la valeur si nécessaire, si elle a conservé le message pendant un temps significatif. Toutefois, le délai d'expiration est traité comme approximatif et la valeur n'a pas besoin d'être décrémentée pour refléter de petits intervalles de temps.

Lorsque le message est extrait par une application à l'aide de l'appel MQGET, la zone *Expiry* représente la durée d'expiration d'origine qui reste.

Une fois le délai d'expiration d'un message écoulé, il peut être supprimé par le gestionnaire de files d'attente. Le message est supprimé lorsqu'un appel MQGET de consultation ou de non-consultation se produit et qu'il aurait renvoyé le message s'il n'était pas déjà arrivé à expiration. Par exemple, un appel MQGET de non-consultation avec la zone *MatchOptions* de MQGMO définie sur MQMO_NONE en lecture à partir d'une file d'attente commandée FIFO supprime tous les messages arrivés à expiration jusqu'au premier message non arrivé à expiration. Avec une file d'attente ordonnée de priorité, le même appel va supprimer les messages arrivés à expiration de priorité supérieure et les messages de priorité égale qui sont arrivés dans la file d'attente avant le premier message non arrivé à expiration.

Un message arrivé à expiration n'est jamais renvoyé à une application (par un appel MQGET de recherche ou de non-recherche). Par conséquent, la valeur de la zone *Expiry* du descripteur de message après un appel MQGET réussi est supérieure à zéro ou la valeur spéciale MQEI_UNLIMITED.

Si un message est inséré dans une file d'attente éloignée, il peut expirer (et être supprimé) alors qu'il se trouve dans une file d'attente de transmission intermédiaire, avant que le message n'atteigne la file d'attente de destination.

Un rapport est généré lorsqu'un message arrivé à expiration est supprimé, si le message a spécifié l'une des options de rapport MQRO_EXPIRATION_ *. Si aucune de ces options n'est spécifiée, aucun rapport de ce type n'est généré ; le message est supposé ne plus être pertinent après cette période (peut-être parce qu'un message ultérieur l'a remplacé).

Pour un message inséré dans un point de synchronisation, l'intervalle d'expiration commence au moment où le message est inséré et non au moment où le point de synchronisation est validé. Il est possible que l'intervalle d'expiration se soit écoulé avant que le point de synchronisation ne soit validé. Dans ce cas, le message sera supprimé à un moment donné après l'opération de validation et le message ne sera pas renvoyé à une application en réponse à une opération MQGET.

Tout autre programme qui supprime des messages en fonction de l'heure d'expiration doit également envoyer un message de rapport approprié si un tel message a été demandé.

Remarque :

1. Si un message est inséré avec une heure *Expiry* égale à zéro ou un nombre supérieur à 999 999 999, l'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC_EXPIRY_ERROR; aucun message de rapport n'est généré dans ce cas.
2. Etant donné qu'un message dont le délai d'expiration est écoulé peut ne pas être supprimé avant une date ultérieure, il se peut que des messages d'une file d'attente aient dépassé leur délai d'expiration et qu'ils ne puissent donc pas être extraits. Ces messages sont néanmoins pris en compte dans le nombre de messages dans la file d'attente à toutes fins, y compris le déclenchement de la profondeur.
3. Un rapport d'expiration est généré, s'il est demandé, lorsque le message est supprimé et non lorsqu'il devient éligible à la suppression.
4. La suppression d'un message arrivé à expiration et la génération d'un rapport d'expiration, le cas échéant, ne font jamais partie de l'unité d'oeuvre de l'application, même si la suppression du message a été planifiée suite à un appel MQGET effectué dans une unité d'oeuvre.
5. Si un message presque arrivé à expiration est extrait par un appel MQGET au sein d'une unité de travail et que l'unité de travail est ensuite annulée, le message peut être supprimé avant de pouvoir être à nouveau extrait.
6. Si un message presque arrivé à expiration est verrouillé par un appel MQGET avec MQGMO_LOCK, le message peut être supprimé avant de pouvoir être extrait par un appel MQGET avec MQGMO_MSG_UNDER_CURSOR; le code anomalie MQRC_NO_MSG_UNDER_CURSOR est renvoyé sur cet appel MQGET suivant si cela se produit.
7. Lorsqu'un message de demande dont le délai d'expiration est supérieur à zéro est extrait, l'application peut effectuer l'une des actions suivantes lorsqu'elle envoie le message de réponse:
   - Copiez le délai d'expiration restant entre le message de demande et le message de réponse.
   - Définissez l'heure d'expiration dans le message de réponse sur une valeur explicite supérieure à zéro.
   - Définissez le délai d'expiration dans le message de réponse sur MQEI_UNLIMITED.

   L'action à effectuer dépend de la conception de l'application. Toutefois, l'action par défaut pour placer des messages dans une file d'attente de messages non livrés doit être de conserver le délai d'expiration restant du message et de continuer à le décrémenter.
8. Les messages de déclenchement sont toujours générés avec MQEI_UNLIMITED.
9. Un message (généralement dans une file d'attente de transmission) dont le nom *Format* est MQFMT_XMIT_Q_HEADER comporte un deuxième descripteur de message dans MQXQH. Il est donc associé à deux zones *Expiry* . Dans ce cas, il convient de noter les points supplémentaires suivants:
   - Lorsqu'une application place un message dans une file d'attente éloignée, le gestionnaire de files d'attente place initialement le message dans une file d'attente de transmission locale et préfixe les

données de message d'application avec une structure MQXQH. Le gestionnaire de files d'attente définit les valeurs des deux zones *Expiry* de sorte qu'elles soient identiques à celles spécifiées par l'application.

Si une application insère un message directement dans une file d'attente de transmission locale, les données de message doivent déjà commencer par une structure MQXQH et le nom de format doit être MQFMT_XMIT_Q_HEADER. Dans ce cas, l'application n'a pas besoin de définir les valeurs de ces deux zones *Expiry* pour qu'elles soient identiques. (Le gestionnaire de files d'attente vérifie que la zone *Expiry* dans MQXQH contient une valeur valide et que les données de message sont suffisamment longues pour l'inclure). Pour une application qui peut écrire directement dans la file d'attente de transmission, l'application doit créer un en-tête de file d'attente de transmission avec le descripteur de message intégré. Toutefois, si la valeur d'expiration du descripteur de message écrit dans la file d'attente de transmission est incohérente avec la valeur du descripteur de message intégré, une erreur d'expiration est rejetée.

- Lorsqu'un message portant le nom *Format* MQFMT_XMIT_Q_HEADER est extrait d'une file d'attente (qu'il s'agisse d'une file d'attente normale ou de transmission), le gestionnaire de files d'attente décrémente *ces deux* zones *Expiry* avec le temps d'attente de la file d'attente. Aucune erreur n'est générée si les données de message ne sont pas suffisamment longues pour inclure la zone *Expiry* dans le MQXQH.
- Le gestionnaire de files d'attente utilise la zone *Expiry* dans le descripteur de message distinct (c'est-à-dire, pas celui du descripteur de message intégré dans la structure MQXQH) pour tester si le message peut être supprimé.
- Si les valeurs initiales des deux zones *Expiry* sont différentes, l'heure *Expiry* dans le descripteur de message distinct lors de l'extraction du message peut être supérieure à zéro (de sorte que le message ne peut pas être supprimé), alors que l'heure indiquée dans la zone *Expiry* de MQXQH s'est écoulée. Dans ce cas, la zone *Expiry* de MQXQH est définie sur zéro.

10. Le délai d'expiration d'un message de réponse renvoyé par le pont IMS est illimité sauf si MQIIH_PASS_EXPIRATION est défini dans la zone Indicateurs de MQIIH. Pour plus d'informations, voir Indicateurs .

La valeur spéciale suivante est reconnue:

MQEI_UNLIMITED
Le message a un délai d'expiration illimité.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQEI_UNLIMITED.

Commentaires en retour (MQLONG)
La zone Feedback est utilisée avec un message de type MQMT_REPORT pour indiquer la nature du rapport et n'est significative qu'avec ce type de message.

La zone peut contenir l'une des valeurs MQFB_ * ou l'une des valeurs MQRC_ *. Les codes retour sont regroupés comme suit:

MQFB_AUCUN
Aucun commentaire n'a été fourni.

MQFB_SYSTEM_FIRST
Valeur la plus faible pour les commentaires générés par le système.

MQFB_SYSTEM_LAST
Valeur la plus élevée pour les commentaires générés par le système.

La plage des codes retour générés par le système MQFB_SYSTEM_FIRST à MQFB_SYSTEM_LAST inclut les codes retour généraux répertoriés dans cette rubrique (MQFB_ *), ainsi que les codes raison (MQRC_ *) qui peuvent se produire lorsque le message ne peut pas être inséré dans la file d'attente de destination.

MQFB_APPL_FIRST
Valeur la plus faible pour les commentaires générés par l'application.

MQFB_APPL_LAST
Valeur la plus élevée pour les commentaires générés par l'application.

Les applications qui génèrent des messages de rapport ne doivent pas utiliser de codes retour dans la plage système (autre que MQFB_QUIT), sauf si elles souhaitent simuler des messages de rapport générés par le gestionnaire de files d'attente ou l'agent MCA.

Dans les appels MQPUT ou MQPUT1 , la valeur spécifiée doit être MQFB_NONE ou être comprise dans la plage système ou la plage d'applications. Cette case est cochée quelle que soit la valeur de *MsgType*.

Codes retour généraux:

MQFB_COA
Confirmation de l'arrivée dans la file d'attente de destination (voir MQRO_COA).

MQFB_COD
Confirmation de la livraison à l'application réceptrice (voir MQRO_COD).

EXPIRATION_MQFB
Le message a été supprimé car il n'avait pas été supprimé de la file d'attente de destination avant que son délai d'expiration ne soit écoulé.

MQFB_PAN
Notification d'action positive (voir MQRO_PAN).

MQFB_NAN
Notification d'action négative (voir MQRO_NAN).

MQFB_QUIT
Arrêtez l'application.

Cela peut être utilisé par un programme de planification de la charge de travail pour contrôler le nombre d'instances d'un programme d'application en cours d'exécution. L'envoi d'un message MQMT_REPORT avec ce code retour à une instance du programme d'application indique à cette instance qu'elle doit arrêter le traitement. Toutefois, le respect de cette convention est du ressort de l'application ; elle n'est pas appliquée par le gestionnaire de files d'attente.

Codes de retour d'informations du canal:

MQFB_CHANNEL_COMPLETED
Un canal s'est arrêté normalement.

MQFB_CHANNEL_FAIL
Un canal s'est arrêté de manière anormale et passe à l'état ARRETE.

MQFB_CHANNEL_FAIL_RETRY
Un canal s'est arrêté de manière anormale et passe à l'état RETRY.

IMS-Codes retour

Ces codes sont utilisés lorsqu'un code de détection IMS-OTMA inattendu est reçu. Le code de détection ou, lorsque le code de détection est 0x1A , le code raison associé à ce code de détection, est indiqué dans le *commentaire en retour*.

1. Pour les codes *Feedback* compris entre MQFB_IMS_FIRST (300) et MQFB_IMS_LAST (399), un code de détection autre que 0x1A a été reçu. Le *code de détection* est donné par l'expression (*Feedback* - MQFB_IMS_FIRST+1)
2. Pour les codes *Feedback* dans la plage MQFB_IMS_NACK_1A_REASON_FIRST (600) à MQFB_IMS_NACK_1A_REASON_LAST (855), un code de détection 0x1A a été reçu. Le *code anomalie* associé au code d'analyse est donné par l'expression (*Feedback* - MQFB_IMS_NACK_1A_REASON_FIRST)

La signification des codes de détection IMS-OTMA et des codes anomalie correspondants est décrite dans le document *Open Transaction Manager Access Guide and Reference*.

Les codes retour suivants peuvent être générés par le pont IMS :

MQFB_DATA_LENGTH_ZERO
La longueur d'un segment était égale à zéro dans les données d'application du message.

MQFB_DONNÉES_LONGUEUR_NÉGATIVE
Une longueur de segment était négative dans les données d'application du message.

MQFB_DATA_LENGTH_TOO_BIG
Une longueur de segment était trop grande dans les données d'application du message.

MQFB_BUFFER_OVERFLOW
La valeur de l'une des zones de longueur entraînerait le dépassement de la mémoire tampon de messages par les données.

MQFB_LENGTH_OFF_BY_ONE
La valeur de l'une des zones de longueur était trop courte de 1 octet.

MQFB_IIH_ERREUR
La zone *Format* dans MQMD indique MQFMT_IMS, mais le message ne commence pas par une structure MQIIH valide.

MQFB_NOT_AUTHORIZED_FOR_IMS
L'ID utilisateur contenu dans le descripteur de message MQMD ou le mot de passe contenu dans la zone *Authenticator* de la structure MQIIH n'a pas pu être validé par le pont IMS . Par conséquent, le message n'a pas été transmis à IMS.

MQFB_IMS_ERREUR
Une erreur inattendue a été renvoyée par IMS. Consultez le journal des erreurs WebSphere MQ sur le système sur lequel réside le pont IMS pour plus d'informations sur l'erreur.

MQFB_IMS_FIRST
Lorsque le code de détection IMS-OTMA n'est pas 0x1A, les codes retour générés par IMSsont compris entre MQFB_IMS_FIRST (300) et MQFB_IMS_LAST (399). Le code de détection IMS-OTMA lui-même est *Feedback* moins MQFB_IMS_ERROR.

MQFB_IMS_LAST
Valeur la plus élevée pour les commentaires générés par IMSlorsque le code de détection n'est pas 0x1A.

MQFB_IMS_NACK_1A_REASON_FIRST
Lorsque le code de détection est 0x1A, les codes retour générés par IMSsont compris entre MQFB_IMS_NACK_1A_REASON_FIRST (600) et MQFB_IMS_NACK_1A_REASON_LAST (855).

MQFB_IMS_NACK_1A_REASON_LAST
Valeur la plus élevée pour les commentaires générés par IMSlorsque le code de détection est 0x1A

Codes retour CICS-bridge: les codes retour suivants peuvent être générés par le pont CICS :

MQFB_CICS_APPL_ABENDED
Le programme d'application indiqué dans le message s'est arrêté de manière anormale. Ce code retour apparaît uniquement dans la zone *Reason* de la structure MQDLH.

MQFB_CICS_APPL_NOT_STARTED
La commande EXEC CICS LINK pour le programme d'application spécifié dans le message a échoué. Ce code retour apparaît uniquement dans la zone *Reason* de la structure MQDLH.

MQFB_CICS_BRIDGE_FAILURE
Le pont CICS s'est arrêté de manière anormale sans avoir terminé le traitement normal des erreurs.

MQFB_CICS_CCSID_ERREUR
Identificateur de jeu de caractères incorrect.

MQFB_CICS_CIH_ERREUR
Structure d'en-tête d'informations CICS manquante ou non valide.

MQFB_CICS_COMMAREA_ERREUR
Longueur de la zone de communication CICS non valide.

MQFB_CICS_CORREL_ID_ERREUR
Identificateur de corrélation incorrect.

MQFB_CICS_DLQ_ERREUR
La tâche de pont CICS n'a pas pu copier une réponse à cette demande dans la file d'attente de rebut. La demande a été annulée.

MQFB_CICS_ENCODING_ERROR
Codage non valide.

MQFB_CICS_INTERNAL_ERROR
Le pont CICS a rencontré une erreur inattendue.

Ce code retour apparaît uniquement dans la zone *Reason* de la structure MQDLH.

MQFB_CICS_NOT_AUTHORIZED
ID utilisateur non autorisé ou mot de passe incorrect.

Ce code retour apparaît uniquement dans la zone *Reason* de la structure MQDLH.

MQFB_CICS_UOW_BACKED_OUT
L'unité de travail a été annulée pour l'une des raisons suivantes:

- Une erreur a été détectée lors du traitement d'une autre demande dans la même unité d'oeuvre.
- Une fin anormale CICS s'est produite alors que l'unité de travail était en cours.

MQFB_ERREUR UOW_CICS_
Zone de contrôle d'unité de travail *UOWControl* incorrecte.

Codes de retour de message de routage de trace:

ACTIVITE MQFB_MQ
Utilisé avec le format MQFMT_EMBEDDED_PCF pour autoriser l'option des données utilisateur suivant les rapports d'activité.

ACTIVITÉS MQFB_MAX_ACTIVITÉS
Renvoyé lorsque le message de trace-route est supprimé car le nombre d'activités dans lesquelles le message a été impliqué dépasse la limite du nombre maximal d'activités.

MQFB_NON_RÉACHEMINEMENT
Renvoyé lorsque le message de trace-route est supprimé car il est sur le point d'être envoyé à un gestionnaire de files d'attente éloignées qui ne prend pas en charge les messages de trace-route.

MQFB_NOT_DELIVERED
Renvoyé lorsque le message de trace-route est supprimé car il est sur le point d'être placé dans une file d'attente locale.

MQFB_UNSUPPORT_FORWARDING
Renvoyé lorsque le message de trace-route est supprimé car une valeur du paramètre de réacheminement n'est pas reconnue et se trouve dans le masque de bits rejeté.

MQFB_UNSUPPORTED_DISTRIBUTION
Renvoyé lorsque le message de trace-route est supprimé car une valeur du paramètre de distribution n'est pas reconnue et se trouve dans le masque de bits rejeté.

Codes anomalieWebSphere MQ: pour les messages de rapport d'exception, *Feedback* contient un code anomalie WebSphere MQ . Les codes anomalie possibles sont les suivants:

MQRC_PUT_INHIBÉ
(2051, X'803') Appels d'insertion bloqués pour la file d'attente.

MQRC_Q_FULL
(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_Q_ESPACE_NON_DISPONIBLE
(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

MQRC_MSG_TOO_BIG_FOR_Q
(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

Pour obtenir la liste complète des codes anomalie, voir:

- Pour WebSphere MQ for z/OS, voir Codes anomalie d'API.
- Pour toutes les autres plateformes, voir Codes achèvement et raison d'API.

.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQFB_NONE.

Format (MQCHAR8)

Il s'agit d'un nom que l'expéditeur du message utilise pour indiquer au destinataire la nature des données du message. Tous les caractères du jeu de caractères du gestionnaire de files d'attente peuvent être spécifiés pour le nom, mais vous devez limiter le nom à ce qui suit:

- Majuscules de A à Z
- Chiffres de 0 à 9

Si d'autres caractères sont utilisés, il se peut qu'il ne soit pas possible de convertir le nom entre les jeux de caractères des gestionnaires de files d'attente d'envoi et de réception.

Mettez le nom à blanc en fonction de la longueur de la zone ou utilisez un caractère indéfini pour terminer le nom avant la fin de la zone. La valeur indéfinie et les caractères suivants sont traités comme des blancs. N'indiquez pas de nom avec des espaces de début ou des blancs imbriqués. Pour l'appel MQGET, le gestionnaire de files d'attente renvoie le nom complété par des blancs à la longueur de la zone.

Le gestionnaire de files d'attente ne vérifie pas que le nom est conforme aux recommandations décrites ci-dessus.

Les noms commençant par MQ en majuscules, en minuscules et en casse mixte ont des significations qui sont définies par le gestionnaire de files d'attente ; n'utilisez pas de noms commençant par ces lettres pour vos propres formats.Les formats intégrés du gestionnaire de files d'attente sont les suivants:

MQFMT_AUCUN
La nature des données n'est pas définie: les données ne peuvent pas être converties lorsque le message est extrait d'une file d'attente à l'aide de l'option MQGMO_CONVERT.

Si vous spécifiez MQGMO_CONVERT dans l'appel MQGET et que le jeu de caractères ou le codage des données du message diffère de celui spécifié dans le paramètre *MsgDesc* , le message est renvoyé avec les codes achèvement et anomalie suivants (en supposant qu'il n'y ait pas d'autres erreurs):

- Code achèvement MQCC_WARNING et code anomalie MQRC_FORMAT_ERROR si les données MQFMT_NONE se trouve au début du message.
- Code achèvement MQCC_OK et code anomalie MQRC_NONE si les données MQFMT_NONE sont à la fin du message (c'est-à-dire précédées d'une ou de plusieurs structures d'en-tête MQ). Les structures d'en-tête MQ sont converties au jeu de caractères et au codage demandés dans ce cas.

Pour le langage de programmation C, la constante MQFMT_NONE_ARRAY est également définie ; elle a la même valeur que MQFMT_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_ADMIN
Le message est un message de demande ou de réponse du serveur de commandes au format PCF (Programmable Command Format). Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET. Pour plus d'informations sur l'utilisation des messages au format de commande programmable, voir Utilisation des formats de commande programmables .

Pour le langage de programmation C, la constante MQFMT_ADMIN_ARRAY est également définie ; elle a la même valeur que MQFMT_ADMIN, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

MQFMT_CICS
Les données de message commencent par l'en-tête d'informations CICS MQCIH, suivi des données d'application. Le nom de format des données d'application est donné par la zone *Format* dans la structure MQCIH.

Sous z/OS, spécifiez l'option MQGMO_CONVERT dans l'appel MQGET pour convertir les messages au format MQFMT_CICS.

Pour le langage de programmation C, la constante MQFMT_CICS_ARRAY est également définie ; elle a la même valeur que MQFMT_CICS, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_COMMAND_1
Le message est un message de réponse du serveur de commandes MQSC contenant le nombre d'objets, le code achèvement et le code anomalie. Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT_COMMAND_1_ARRAY est également définie ; elle a la même valeur que MQFMT_COMMAND_1, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_COMMAND_2
Le message est un message de réponse du serveur de commandes MQSC contenant des informations sur les objets demandés. Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT_COMMAND_2_ARRAY est également définie ; elle a la même valeur que MQFMT_COMMAND_2, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_DEAD_LETTER_HEADER
Les données de message commencent par l'en-tête de rebut MQDLH. Les données du message d'origine suivent immédiatement la structure MQDLH. Le nom de format des données de message d'origine est donné par la zone *Format* dans la structure MQDLH ; pour plus de détails sur cette structure, voir «En-tête MQDLH-Dead-letter», à la page 327 . Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Les rapports COA et COD ne sont pas générés pour les messages dont le *Format* est MQFMT_DEAD_LETTER_HEADER.

Pour le langage de programmation C, la constante MQFMT_DEAD_LETTER_HEADER_ARRAY est également définie ; elle a la même valeur que MQFMT_DEAD_LETTER_HEADER, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

MQFMT_DIST_HEADER
Les données de message commencent par l'en-tête de liste de distribution MQDH ; cela inclut les tableaux des enregistrements MQOR et MQPMR. L'en-tête de liste de distribution peut être suivi de données supplémentaires. Le format des données supplémentaires (le cas échéant) est fourni par la zone *Format* dans la structure MQDH ; voir «En-tête MQDH-Distribution», à la page 320 pour plus de détails sur cette structure. Les messages au format MQFMT_DIST_HEADER peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Ce format est pris en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus WebSphere MQ clients MQI connectés à ces systèmes.

Pour le langage de programmation C, la constante MQFMT_DIST_HEADER_ARRAY est également définie ; elle a la même valeur que MQFMT_DIST_HEADER, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_EMBEDDED_PCF
Format d'un message de suivi, à condition que la valeur de la commande PCF soit définie sur MQCMD_TRACE_ROUTE. L'utilisation de ce format permet d'envoyer des données utilisateur avec

le message de trace, à condition que leurs applications puissent s'adapter aux paramètres PCF précédents.

L'en-tête PCF **doit** être le premier en-tête, sinon le message ne sera pas traité comme un message de trace. Cela signifie que le message ne peut pas faire partie d'un groupe et que les messages de trace-route ne peuvent pas être segmentés. Si un message de trace-route est envoyé dans un groupe, le message est rejeté avec le code anomalie MQRC_MSG_NOT_ALLOWED_IN_GROUP.

Notez que MQFMT_ADMIN peut également être utilisé pour le format d'un message de routage de trace, mais dans ce cas, aucune donnée utilisateur ne peut être envoyée avec le message de routage de trace.

MQFMT_EVENT
Le message est un message d'événement MQ qui signale un événement qui s'est produit. Les messages d'événement ont la même structure que les commandes programmables ; voir Messages de commande PCF pour plus d'informations sur cette structure et Surveillance des événements pour plus d'informations sur les événements.

Les messages d'événement Version-1 peuvent être convertis dans tous les environnements si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET. Les messages d'événement Version-2 peuvent être convertis uniquement sur z/OS.

Pour le langage de programmation C, la constante MQFMT_EVENT_ARRAY est également définie ; elle a la même valeur que MQFMT_EVENT, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_IMS
Les données de message commencent par l'en-tête d'information IMS MQIIH, qui est suivi des données d'application. Le nom de format des données d'application est donné par la zone *Format* dans la structure MQIIH.

Pour plus de détails sur la façon dont la structure MQIIH est gérée lors de l'utilisation de MQGET avec MQGMO_CONVERT, voir «Format (MQCHAR8)», à la page 381 et «Format ReplyTo(MQCHAR8)», à la page 381.

Pour le langage de programmation C, la constante MQFMT_IMS_ARRAY est également définie ; elle a la même valeur que MQFMT_IMS, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_IMS_VAR_STRING
Le message est une chaîne de variable IMS , qui est une chaîne au format `llzzccc`, où:

ll
est une zone de longueur de 2 octets spécifiant la longueur totale de l'élément de chaîne de variable IMS . Cette longueur est égale à la longueur de `ll` (2 octets), plus la longueur de `zz` (2 octets), plus la longueur de la chaîne de caractères elle-même. `ll` est un entier binaire à 2 octets dans le codage spécifié par la zone *Encoding* .

zz
est une zone de 2 octets contenant des indicateurs significatifs pour IMS. `zz` est une chaîne d'octets composée de deux zones MQBYTE et est transmise sans modification de l'expéditeur au destinataire (c'est-à-dire que `zz` n'est soumis à aucune conversion).

ccc
est une chaîne de caractères de longueur variable contenant `ll-4` caractères. `ccc` est dans le jeu de caractères spécifié par la zone *CodedCharSetId* .

Sous z/OS, les données de message peuvent être constituées d'une séquence de chaînes de variables IMS qui sont regroupées, chaque chaîne étant de la forme `llzzccc`. Aucun octet ne doit être ignoré entre les chaînes de variables IMS successives. Cela signifie que si la première chaîne a une longueur impaire, la deuxième chaîne sera mal alignée, c'est-à-dire qu'elle ne commencera pas sur une limite qui est un multiple de deux. Faites attention lorsque vous construisez de telles chaînes sur des machines qui nécessitent l'alignement de types de données élémentaires.

Utilisez l'option MQGMO_CONVERT dans l'appel MQGET pour convertir les messages dont le format est MQFMT_IMS_VAR_STRING.

Pour le langage de programmation C, la constante MQFMT_IMS_VAR_STRING_ARRAY est également définie ; elle a la même valeur que MQFMT_IMS_VAR_STRING, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_MD_EXTENSION
Les données de message commencent par l'extension de descripteur de message MQMDE et sont éventuellement suivies par d'autres données (généralement les données de message d'application). Le nom de format, le jeu de caractères et le codage des données qui suivent MQMDE sont fournis par les zones *Format*, *CodedCharSetId*et *Encoding* dans MQMDE. Pour plus de détails sur cette structure, voir «MQMDE-Extension de descripteur de message», à la page 447 . Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT_MD_EXTENSION_ARRAY est également définie ; elle a la même valeur que MQFMT_MD_EXTENSION, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

MQFMT_PCF
Le message est un message défini par l'utilisateur qui est conforme à la structure d'un message PCF (Programmable Command Format). Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET. Pour plus d'informations sur l'utilisation des messages au format de commande programmable, voir Utilisation des formats de commande programmables .

Pour le langage de programmation C, la constante MQFMT_PCF_ARRAY est également définie ; elle a la même valeur que MQFMT_PCF, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_REF_MSG_HEADER
Les données de message commencent par l'en-tête de message de référence MQRMH et sont éventuellement suivies d'autres données. Le nom de format, le jeu de caractères et le codage des données sont donnés par les zones *Format*, *CodedCharSetId*et *Encoding* dans MQRMH. Pour plus de détails sur cette structure, voir «MQRMH-En-tête de message de référence», à la page 528 . Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Ce format est pris en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus WebSphere MQ clients MQI connectés à ces systèmes.

Pour le langage de programmation C, la constante MQFMT_REF_MSG_HEADER_ARRAY est également définie ; elle a la même valeur que MQFMT_REF_MSG_HEADER, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_RF_HEADER
Les données de message commencent par les règles et l'en-tête de formatage MQRFH, et sont éventuellement suivies par d'autres données. Le nom de format, le jeu de caractères et le codage des données (le cas échéant) sont fournis par les zones *Format*, *CodedCharSetId*et *Encoding* dans le MQRFH. Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT_RF_HEADER_ARRAY est également définie ; elle a la même valeur que MQFMT_RF_HEADER, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_RF_HEADER_2
Les données de message commencent par les règles version-2 et l'en-tête de formatage MQRFH2, et sont éventuellement suivies par d'autres données. Le nom de format, le jeu de caractères et le codage des données facultatives (le cas échéant) sont fournis par les zones *Format*, *CodedCharSetId*et *Encoding* dans MQRFH2. Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT_RF_HEADER_2_ARRAY est également définie ; elle a la même valeur que MQFMT_RF_HEADER_2, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

MQFMT_STRING
Les données de message d'application peuvent être soit une chaîne SBCS (jeu de caractères codé sur un octet), soit une chaîne DBCS (jeu de caractères codé sur deux octets). Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT_STRING_ARRAY est également définie ; elle a la même valeur que MQFMT_STRING, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_TRIGGER
Le message est un message de déclenchement, décrit par la structure MQTM. Pour plus de détails sur cette structure, voir «MQTM-Message de déclenchement», à la page 581 . Les messages de ce format peuvent être convertis si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET.

Pour le langage de programmation C, la constante MQFMT_TRIGGER_ARRAY est également définie ; elle a la même valeur que MQFMT_TRIGGER, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

MQFMT_WORK_INFO_HEADER
Les données de message commencent par l'en-tête d'informations de travail MQWIH, qui est suivi des données d'application. Le nom de format des données d'application est donné par la zone *Format* dans la structure MQWIH.

Sous z/OS, spécifiez l'option MQGMO_CONVERT dans l'appel MQGET pour convertir les *données utilisateur* dans les messages au format MQFMT_WORK_INFO_HEADER. Toutefois, la structure MQWIH elle-même est toujours renvoyée dans le jeu de caractères et le codage du gestionnaire de files d'attente (c'est-à-dire que la structure MQWIH est convertie, que l'option MQGMO_CONVERT soit spécifiée ou non).

Pour le langage de programmation C, la constante MQFMT_WORK_INFO_HEADER_ARRAY est également définie ; elle a la même valeur que MQFMT_WORK_INFO_HEADER, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

MQFMT_XMIT_Q_HEADER
Les données de message commencent par l'en-tête de file d'attente de transmission MQXQH. Les données du message d'origine suivent immédiatement la structure MQXQH. Le nom de format des données de message d'origine est donné par la zone *Format* de la structure MQMD, qui fait partie de l'en-tête de file d'attente de transmission MQXQH. Pour plus de détails sur cette structure, voir «MQXQH-en-tête de file d'attente de transmission», à la page 601 .

Les rapports COA et COD ne sont pas générés pour les messages dont le *Format* est MQFMT_XMIT_Q_HEADER.

Pour le langage de programmation C, la constante MQFMT_XMIT_Q_HEADER_ARRAY est également définie ; elle a la même valeur que MQFMT_XMIT_Q_HEADER, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La longueur de cette zone est indiquée par MQ_FORMAT_LENGTH. La valeur initiale de cette zone est MQFMT_NONE.

GroupId (MQBYTE24)

Il s'agit d'une chaîne d'octets utilisée pour identifier le groupe de messages particulier ou le message logique auquel appartient le message physique. *GroupId* est également utilisé si la segmentation est autorisée pour le message. Dans tous ces cas, *GroupId* a une valeur non nulle et un ou plusieurs des indicateurs suivants sont définis dans la zone *MsgFlags* :

- GROUPE MQMF_MSG_IN_GROUP
- MQMF_LAST_MSG_IN_GROUP
- SEGMENT_MQMF
- SEGMENT_DERNIER_MQMF_SEGMENT

- MQMF_SEGMENTATION_ALLOWED

Si aucun de ces indicateurs n'est défini, *GroupId* a la valeur null spéciale MQGI_NONE.

L'application n'a pas besoin de définir cette zone dans l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, MQPMO_LOGICAL_ORDER est spécifié.
- Dans l'appel MQGET, MQMO_MATCH_GROUP_ID n'est *pas* spécifié.

Il s'agit des méthodes recommandées pour utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application requiert davantage de contrôle ou si l'appel est MQPUT1, l'application doit s'assurer que *GroupId* est défini sur une valeur appropriée.

Les groupes de messages et les segments ne peuvent être traités correctement que si l'identificateur de groupe est unique. Pour cette raison, les applications *ne doivent pas générer leurs propres identificateurs de groupe*; à la place, les applications doivent effectuer l'une des opérations suivantes:

- Si MQPMO_LOGICAL_ORDER est spécifié, le gestionnaire de files d'attente génère automatiquement un identificateur de groupe unique pour le premier message du groupe ou du segment du message logique et utilise cet identificateur de groupe pour les messages restants du groupe ou des segments du message logique, de sorte que l'application n'a pas besoin d'effectuer d'action spéciale. Il s'agit de la procédure recommandée.
- Si MQPMO_LOGICAL_ORDER n'est *pas* spécifié, l'application doit demander au gestionnaire de files d'attente de générer l'identificateur de groupe, en définissant *GroupId* sur MQGI_NONE lors du premier appel MQPUT ou MQPUT1 pour un message dans le groupe ou le segment du message logique. L'identificateur de groupe renvoyé par le gestionnaire de files d'attente en sortie de cet appel doit ensuite être utilisé pour les messages restants du groupe ou des segments du message logique. Si un groupe de messages contient des messages segmentés, le même identificateur de groupe doit être utilisé pour tous les segments et messages du groupe.

  Lorsque MQPMO_LOGICAL_ORDER n'est pas spécifié, les messages des groupes et les segments de messages logiques peuvent être placés dans n'importe quel ordre (par exemple, dans l'ordre inverse), mais l'identificateur de groupe doit être alloué par le *premier* appel MQPUT ou MQPUT1 émis pour l'un de ces messages.

Lors de l'entrée dans les appels MQPUT et MQPUT1 , le gestionnaire de files d'attente utilise la valeur décrite dans Ordre physique dans une file d'attente. Dans la sortie des appels MQPUT et MQPUT1 , le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message si l'objet ouvert est une file d'attente unique et non une liste de distribution, mais la laisse inchangée si l'objet ouvert est une liste de distribution. Dans ce dernier cas, si l'application doit connaître les identificateurs de groupe générés, elle doit fournir des enregistrements MQPMR contenant la zone *GroupId* .

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur décrite dans Tableau 506, à la page 365. Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur spéciale suivante est définie:

MQGI_AUCUN
Aucun identificateur de groupe n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone. Il s'agit de la valeur utilisée pour les messages qui ne sont pas dans des groupes, qui ne sont pas des segments de messages logiques et pour lesquels la segmentation n'est pas autorisée.

Pour le langage de programmation C, la constante MQGI_NONE_ARRAY est également définie ; elle a la même valeur que MQGI_NONE, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

La longueur de cette zone est indiquée par MQ_GROUP_ID_LENGTH. La valeur initiale de cette zone est MQGI_NONE. Cette zone est ignorée si *Version* est inférieur à MQMD_VERSION_2.

MsgFlags (MQLONG)
Les MsgFlags sont des indicateurs qui spécifient les attributs du message ou qui contrôlent son traitement.

Les indicateurs de message (MsgFlags) se répartissent dans les catégories suivantes:

- Indicateurs de segmentation
- Indicateurs de statut

Indicateurs de segmentation: Lorsqu'un message est trop volumineux pour une file d'attente, une tentative d'insertion du message dans la file d'attente échoue généralement. La segmentation est une technique par laquelle le gestionnaire de files d'attente ou l'application fractionne le message en éléments plus petits appelés segments et place chaque segment dans la file d'attente sous la forme d'un message physique distinct. L'application qui extrait le message peut soit extraire les segments un par un, soit demander au gestionnaire de files d'attente de réassembler les segments en un seul message renvoyé par l'appel MQGET. Ce dernier est obtenu en spécifiant l'option MQGMO_COMPLETE_MSG dans l'appel MQGET et en fournissant une mémoire tampon suffisamment grande pour contenir le message complet. (Voir «Options MQGMO-Get-message», à la page 345 pour plus de détails sur l'option MQGMO_COMPLETE_MSG.) Un message peut être segmenté au niveau du gestionnaire de files d'attente d'envoi, au niveau d'un gestionnaire de files d'attente intermédiaire ou au niveau du gestionnaire de files d'attente de destination.

Vous pouvez spécifier l'une des options suivantes pour contrôler la segmentation d'un message:

MQMF_SEGMENTATION_INHIBÉE
Cette option empêche le message d'être divisé en segments par le gestionnaire de files d'attente. Si elle est spécifiée pour un message qui est déjà un segment, cette option empêche le segment d'être divisé en segments plus petits.

La valeur de cet indicateur est zéro binaire. Il s'agit de l'option par défaut.

MQMF_SEGMENTATION_ALLOWED
Cette option permet au message d'être divisé en segments par le gestionnaire de files d'attente. Si elle est spécifiée pour un message qui est déjà un segment, cette option permet de le scinde en segments plus petits. MQMF_SEGMENTATION_ALLOWED peut être défini sans que MQMF_SEGMENT ou MQMF_LAST_SEGMENT ne soit défini.

- Sous z/OS, le gestionnaire de files d'attente ne prend pas en charge la segmentation des messages. Si un message est trop volumineux pour la file d'attente, l'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC_MSG_TOO_BIG_FOR_Q. Toutefois, l'option MQMF_SEGMENTATION_ALLOWED peut toujours être spécifiée et permet de segmenter le message au niveau d'un gestionnaire de files d'attente éloignées.

Lorsque le gestionnaire de files d'attente segmente un message, il active l'indicateur MQMF_SEGMENT dans la copie du MQMD envoyé avec chaque segment, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1 . Pour le dernier segment du message logique, le gestionnaire de files d'attente active également l'indicateur MQMF_LAST_SEGMENT dans le MQMD envoyé avec le segment.

Remarque : Faites attention lors de l'insertion de messages avec MQMF_SEGMENTATION_ALLOWED mais sans MQPMO_LOGICAL_ORDER. Si le message est:

- N'est pas un segment, et
- Pas dans un groupe, et
- Non réacheminé,

l'application doit réinitialiser la zone *GroupId* sur MQGI_NONE avant *chaque appel* MQPUT ou MQPUT1 , afin que le gestionnaire de files d'attente puisse générer un identificateur de groupe unique pour chaque message. Si tel n'est pas le cas, les messages non liés peuvent avoir le même identificateur de groupe, ce qui peut entraîner un traitement incorrect par la suite. Pour plus d'informations sur la réinitialisation de la zone *GroupId* , voir les descriptions de la zone *GroupId* et de l'option MQPMO_LOGICAL_ORDER.

Le gestionnaire de files d'attente divise les messages en segments selon les besoins afin que les segments (plus les données d'en-tête requises) tiennent dans la file d'attente. Toutefois, il existe

une limite inférieure pour la taille d'un segment généré par le gestionnaire de files d'attente et seul le dernier segment créé à partir d'un message peut être inférieur à cette limite (la limite inférieure pour la taille d'un segment généré par une application est d'un octet). Les segments générés par le gestionnaire de files d'attente peuvent être de longueur inégale. Le gestionnaire de files d'attente traite le message comme suit:

- Les formats définis par l'utilisateur sont divisés sur des limites multiples de 16 octets ; le gestionnaire de files d'attente ne génère pas de segments inférieurs à 16 octets (autres que le dernier segment).
- Les formats intégrés autres que MQFMT_STRING sont fractionnés aux points appropriés à la nature des données présentes. Toutefois, le gestionnaire de files d'attente ne fractionne jamais un message au milieu d'une structure d'en-tête WebSphere MQ . Cela signifie qu'un segment contenant une structure d'en-tête MQ unique ne peut pas être fractionné davantage par le gestionnaire de files d'attente et que, par conséquent, la taille de segment minimale possible pour ce message est supérieure à 16 octets.

  Le deuxième segment ou un segment ultérieur généré par le gestionnaire de files d'attente commence par l'un des segments suivants:

  – Une structure d'en-tête MQ
  – Début des données de message d'application
  – Une partie de l'utilisation des données de message d'application
- MQFMT_STRING est divisé sans tenir compte de la nature des données présentes (SBCS, DBCS ou SBCS/DBCS mixte). Lorsque la chaîne est de type DBCS ou SBCS/DBCS mixte, il se peut que des segments ne puissent pas être convertis d'un jeu de caractères à un autre. Le gestionnaire de files d'attente ne fractionne jamais les messages MQFMT_STRING en segments de moins de 16 octets (autres que le dernier segment).
- Le gestionnaire de files d'attente définit les zones *Format*, *CodedCharSetId*et *Encoding* dans le MQMD de chaque segment pour décrire correctement les données présentes au *début* du segment ; le nom de format est soit le nom d'un format intégré, soit le nom d'un format défini par l'utilisateur.
- La zone *Report* dans le MQMD des segments dont le *Offset* est supérieur à zéro est modifiée. Pour chaque type de rapport, si l'option de rapport est MQRO_ * _WITH_DATA, mais que le segment ne peut pas contenir les 100 premiers octets de données utilisateur (c'est-à-dire, les données qui suivent les structures d'en-tête WebSphere MQ éventuellement présentes), l'option de rapport est remplacée par MQRO_ *.

Le gestionnaire de files d'attente respecte les règles ci-dessus, mais fractionne les messages de manière imprévisible ; ne faites pas d'hypothèses sur l'endroit où un message est fractionné.

Pour les messages *persistants* , le gestionnaire de files d'attente peut effectuer une segmentation uniquement dans une unité d'oeuvre:

- Si l'appel MQPUT ou MQPUT1 s'exécute dans une unité de travail définie par l'utilisateur, cette unité de travail est utilisée. Si l'appel échoue lors du processus de segmentation, le gestionnaire de files d'attente supprime tous les segments placés dans la file d'attente suite à l'échec de l'appel. Toutefois, l'échec n'empêche pas la validation de l'unité de travail.
- Si l'appel est effectué en dehors d'une unité de travail définie par l'utilisateur et qu'il n'existe aucune unité de travail définie par l'utilisateur, le gestionnaire de files d'attente crée une unité de travail uniquement pour la durée de l'appel. Si l'appel aboutit, le gestionnaire de files d'attente valide automatiquement l'unité de travail. Si l'appel échoue, le gestionnaire de files d'attente annule l'unité d'oeuvre.
- Si l'appel est exécuté en dehors d'une unité de travail définie par l'utilisateur, mais qu'une unité de travail définie par l'utilisateur existe, le gestionnaire de files d'attente ne peut pas effectuer de segmentation. Si le message ne nécessite pas de segmentation, l'appel peut tout de même aboutir. Mais si le message requiert une segmentation, l'appel échoue avec le code anomalie MQRC_UOW_NOT_AVAILABLE.

Pour les messages *non persistants* , le gestionnaire de files d'attente ne requiert pas qu'une unité d'oeuvre soit disponible pour effectuer la segmentation.

Soyez particulièrement prudent lorsque vous convertissez des données dans des messages qui peuvent être segmentés:

- Si l'application de réception convertit des données dans l'appel MQGET et spécifie l'option MQGMO_COMPLETE_MSG, l'exit de conversion de données reçoit le message complet de l'exit à convertir et le fait que le message a été segmenté est visible pour l'exit.
- Si l'application de réception extrait un segment à la fois, l'exit de conversion de données est appelé pour convertir un segment à la fois. L'exit doit donc convertir les données dans un segment indépendamment des données dans n'importe lequel des autres segments.

  Si la nature des données dans le message est telle que la segmentation arbitraire des données sur des limites de 16 octets peut entraîner des segments qui ne peuvent pas être convertis par l'exit, ou si le format est MQFMT_STRING et que le jeu de caractères est DBCS ou SBCS/DBCS mixte, l'application émettrice doit créer et placer les segments, en spécifiant MQMF_SEGMENTATION_INHIBÉE pour supprimer la segmentation supplémentaire. De cette manière, l'application émettrice peut s'assurer que chaque segment contient suffisamment d'informations pour permettre à l'exit de conversion de données de convertir le segment avec succès.
- Si la conversion de l'émetteur est indiquée pour un agent MCA, ce dernier convertit uniquement les messages qui ne sont pas des segments de messages logiques ; l'agent MCA ne tente jamais de convertir les messages qui sont des segments.

Cet indicateur est un indicateur d'entrée sur les appels MQPUT et MQPUT1 et un indicateur de sortie sur l'appel MQGET. Lors de ce dernier appel, le gestionnaire de files d'attente renvoie également la valeur de l'indicateur à la zone *Segmentation* dans MQGMO.

La valeur initiale de cet indicateur est MQMF_SEGMENTATION_INHIBÉE.

Indicateurs de statut: indicateurs qui indiquent si le message physique appartient à un groupe de messages, s'il s'agit d'un segment d'un message logique, les deux ou aucun des deux. Un ou plusieurs des éléments suivants peuvent être spécifiés dans l'appel MQPUT ou MQPUT1 ou renvoyés par l'appel MQGET:

GROUPE MQMF_MSG_IN_GROUP
Le message est membre d'un groupe.

MQMF_LAST_MSG_IN_GROUP
Le message est le dernier message logique d'un groupe.

Si cet indicateur est défini, le gestionnaire de files d'attente active MQMF_MSG_IN_GROUP dans la copie de MQMD envoyée avec le message, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1 .

Un groupe ne peut comporter qu'un seul message logique. Si tel est le cas, MQMF_LAST_MSG_IN_GROUP est défini, mais la zone *MsgSeqNumber* a la valeur 1.

SEGMENT_MQMF
Le message est un segment d'un message logique.

Lorsque MQMF_SEGMENT est spécifié sans MQMF_LAST_SEGMENT, la longueur des données de message d'application dans le segment (*à l'exclusion* des longueurs des structures d'en-tête WebSphere MQ pouvant être présentes) doit être au moins une. Si la longueur est égale à zéro, l'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC_SEGMENT_LENGTH_ZERO.

Sous z/OS, cette option n'est pas prise en charge si le message est placé dans une file d'attente dont le type d'index est MQIT_GROUP_ID.

SEGMENT_DERNIER_MQMF_SEGMENT
Le message est le dernier segment d'un message logique.

Si cet indicateur est défini, le gestionnaire de files d'attente active MQMF_SEGMENT dans la copie de MQMD envoyée avec le message, mais ne modifie pas les paramètres de ces indicateurs dans le MQMD fourni par l'application sur l'appel MQPUT ou MQPUT1 .

Un message logique ne peut se composer que d'un seul segment. Si tel est le cas, MQMF_LAST_SEGMENT est défini, mais la zone *Offset* a la valeur zéro.

Lorsque MQMF_LAST_SEGMENT est spécifié, la longueur des données de message d'application dans le segment (*à l'exclusion* des longueurs des structures d'en-tête éventuellement présentes) peut être égale à zéro.

Sous z/OS, cette option n'est pas prise en charge si le message est placé dans une file d'attente dont le type d'index est MQIT_GROUP_ID.

L'application doit s'assurer que ces indicateurs sont correctement définis lors de l'insertion de messages. Si MQPMO_LOGICAL_ORDER est spécifié ou a été spécifié dans l'appel MQPUT précédent pour l'identificateur de file d'attente, les paramètres des indicateurs doivent être cohérents avec les informations de groupe et de segment conservées par le gestionnaire de files d'attente pour l'identificateur de file d'attente. Les conditions suivantes s'appliquent aux appels MQPUT *successifs* pour le descripteur de file d'attente lorsque MQPMO_LOGICAL_ORDER est spécifié:

- S'il n'y a pas de groupe ou de message logique en cours, tous ces indicateurs (et leurs combinaisons) sont valides.
- Une fois que MQMF_MSG_IN_GROUP a été spécifié, il doit rester en fonction jusqu'à ce que MQMF_LAST_MSG_IN_GROUP soit spécifié. L'appel échoue avec le code anomalie MQRC_INCOMPLETE_GROUP si cette condition n'est pas satisfaite.
- Une fois que MQMF_SEGMENT a été spécifié, il doit rester en fonction jusqu'à ce que MQMF_LAST_SEGMENT soit spécifié. L'appel échoue avec le code anomalie MQRC_INCOMPLETE_MSG si cette condition n'est pas satisfaite.
- Une fois que MQMF_SEGMENT a été spécifié sans MQMF_MSG_IN_GROUP, MQMF_MSG_IN_GROUP doit rester *désactivé* jusqu'à ce que MQMF_LAST_SEGMENT ait été spécifié. L'appel échoue avec le code anomalie MQRC_INCOMPLETE_MSG si cette condition n'est pas satisfaite.

Ordre physique dans une file d'attente affiche les combinaisons valides des indicateurs et les valeurs utilisées pour les différentes zones.

Ces indicateurs sont des indicateurs d'entrée sur les appels MQPUT et MQPUT1 et des indicateurs de sortie sur l'appel MQGET. Lors de ce dernier appel, le gestionnaire de files d'attente répercute également les valeurs des indicateurs dans les zones *GroupStatus* et *SegmentStatus* de MQGMO.

Vous ne pouvez pas utiliser de messages groupés ou segmentés avec la fonction de publication / abonnement.

Indicateurs par défaut: Vous pouvez indiquer ce qui suit pour indiquer que le message possède des attributs par défaut:

MQMF_AUCUN
Aucun indicateur de message (attributs de message par défaut).

Cela inhibe la segmentation et indique que le message ne fait pas partie d'un groupe et n'est pas un segment d'un message logique. MQMF_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cet indicateur soit utilisé avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

La zone *MsgFlags* est partitionnée en sous-zones ; pour plus de détails, voir «Options de rapport et indicateurs de message», à la page 894.

La valeur initiale de cette zone est MQMF_NONE. Cette zone est ignorée si *Version* est inférieur à MQMD_VERSION_2.

MsgId (MQBYTE24)

Il s'agit d'une chaîne d'octets utilisée pour distinguer un message d'un autre. En règle générale, deux messages ne doivent pas avoir le même identificateur de message, bien que cela ne soit pas interdit par

le gestionnaire de files d'attente. L'identificateur de message est une propriété permanente du message et est conservé lors des redémarrages du gestionnaire de files d'attente. Etant donné que l'identificateur de message est une chaîne d'octets et non une chaîne de caractères, l'identificateur de message n'est *pas* converti entre les jeux de caractères lorsque le message passe d'un gestionnaire de files d'attente à un autre.

Pour les appels MQPUT et MQPUT1 , si MQMI_NONE ou MQPMO_NEW_MSG_ID est spécifié par l'application, le gestionnaire de files d'attente génère un identificateur de message unique[2]lorsque le message est inséré et le place dans le descripteur de message envoyé avec le message. Le gestionnaire de files d'attente renvoie également cet identificateur de message dans le descripteur de message appartenant à l'application émettrice. L'application peut utiliser cette valeur pour enregistrer des informations sur des messages particuliers et pour répondre à des requêtes provenant d'autres parties de l'application.

Si le message est inséré dans une rubrique, le gestionnaire de files d'attente génère des identificateurs de message uniques pour chaque message publié. Si MQPMO_NEW_MSG_ID est spécifié par l'application, le gestionnaire de files d'attente génère un identificateur de message unique à renvoyer en sortie. Si MQMI_NONE est spécifié par l'application, la valeur de la zone *MsgId* dans le MQMD reste inchangée lors du retour de l'appel.

Pour plus d'informations sur les publications conservées, voir la description de MQPMO_RETAIN dans «Options MQPMO (MQLONG)», à la page 485 .

Si le message est inséré dans une liste de distribution, le gestionnaire de files d'attente génère des identificateurs de message uniques si nécessaire, mais la valeur de la zone *MsgId* dans MQMD est inchangée lors du retour de l'appel, même si MQMI_NONE ou MQPMO_NEW_MSG_ID a été spécifié. Si l'application a besoin de connaître les identificateurs de message générés par le gestionnaire de files d'attente, elle doit fournir des enregistrements MQPMR contenant la zone *MsgId* .

L'application émettrice peut également spécifier une valeur pour l'identificateur de message autre que MQMI_NON; cela empêche le gestionnaire de files d'attente de générer un identificateur de message unique. Une application qui transfère un message peut l'utiliser pour propager l'identificateur du message d'origine.

Le gestionnaire de files d'attente n'utilise pas cette zone sauf pour:

- Générer une valeur unique si demandée, comme décrit ci-dessus
- Distribuer la valeur à l'application qui émet la demande d'obtention du message
- Copiez la valeur dans la zone *CorrelId* de tout message de rapport généré à propos de ce message (en fonction des options *Report*)

Lorsque le gestionnaire de files d'attente ou un agent MCA génère un message de rapport, il définit la zone *MsgId* de la manière spécifiée par la zone *Report* du message d'origine, à savoir MQRO_NEW_MSG_ID ou MQRO_PASS_MSG_ID. Les applications qui génèrent des messages de rapport doivent également effectuer cette opération.

Pour l'appel MQGET, *MsgId* est l'une des cinq zones pouvant être utilisées pour extraire un message particulier de la file d'attente. Normalement, l'appel MQGET renvoie le message suivant dans la file

---

[2] Un *MsgId* généré par le gestionnaire de files d'attente se compose d'un identificateur de produit de 4 octets (AMQ ¬ ou CSQ ¬ en ASCII ou EBCDIC, où ¬ représente un caractère blanc), suivi d'une implémentation spécifique au produit d'une chaîne unique. Dans WebSphere MQ , il contient les 12 premiers caractères du nom du gestionnaire de files d'attente, ainsi qu'une valeur dérivée de l'horloge système. Tous les gestionnaires de files d'attente qui peuvent intercommuniquer doivent donc avoir des noms qui diffèrent dans les 12 premiers caractères, afin de s'assurer que les identificateurs de message sont uniques. La possibilité de générer une chaîne unique dépend également du fait que l'horloge système n'est pas modifiée en amont. Pour éviter qu'un identificateur de message généré par le gestionnaire de files d'attente ne soit dupliqué par l'application, celle-ci doit éviter de générer des identificateurs dont les caractères initiaux sont compris entre A et I en ASCII ou EBCDIC (X'41'à X'49'et X'C1'à X'C9'). Toutefois, l'application n'est pas empêchée de générer des identificateurs avec des caractères initiaux dans ces plages.

d'attente, mais un message particulier peut être obtenu en spécifiant un ou plusieurs des cinq critères de sélection, dans n'importe quelle combinaison ; ces zones sont les suivantes:

- *MsgId*
- *CorrelId*
- *GroupId*
- *MsgSeqNumber*
- *Offset*

L'application définit une ou plusieurs de ces zones sur les valeurs requises, puis définit les options de correspondance MQMO_ * correspondantes dans la zone *MatchOptions* de MQGMO pour utiliser ces zones comme critères de sélection. Seuls les messages dont les valeurs sont indiquées dans ces zones peuvent être extraits. La valeur par défaut de la zone *MatchOptions* (si elle n'est pas modifiée par l'application) correspond à la fois à l'identificateur de message et à l'identificateur de corrélation.

Sous z/OS, les critères de sélection que vous pouvez utiliser sont limités par le type d'index utilisé pour la file d'attente. Pour plus de détails, voir l'attribut de file d'attente *IndexType* .

Normalement, le message renvoyé est le *premier* message de la file d'attente qui répond aux critères de sélection. Mais si MQGMO_BROWSE_NEXT est spécifié, le message renvoyé est le message *suivant* qui répond aux critères de sélection ; l'analyse de ce message commence par le message *suivant* la position actuelle du curseur.

Remarque : La file d'attente est analysée séquentiellement à la recherche d'un message qui répond aux critères de sélection. Par conséquent, les temps d'extraction sont plus lents que si aucun critère de sélection n'est spécifié, en particulier si de nombreux messages doivent être analysés avant qu'un message approprié ne soit trouvé. Les exceptions sont les suivantes:

- un appel MQGET par *CorrelId* sur des plateformes réparties 64 bits où l'index *CorrelId* élimine la nécessité d'effectuer une analyse séquentielle réelle.
- un appel MQGET par *IndexType* sur z/OS.

Dans ces deux cas, les performances d'extraction sont améliorées.

Pour plus d'informations sur l'utilisation des critères de sélection dans diverses situations, voir Tableau 506, à la page 365 .

La spécification de MQMI_NONE comme identificateur de message a le même effet que la *non* spécification de MQMO_MATCH_MSG_ID, c'est-à-dire que *tout* identificateur de message correspond.

Cette zone est ignorée si l'option MQGMO_MSG_UNDER_CURSOR est spécifiée dans le paramètre *GetMsgOpts* de l'appel MQGET.

En cas de retour d'un appel MQGET, la zone *MsgId* est définie sur l'identificateur du message renvoyé (le cas échéant).

La valeur spéciale suivante peut être utilisée:

MQMI_AUCUN
Aucun identificateur de message n'est indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQMI_NONE_ARRAY est également définie ; elle a la même valeur que MQMI_NONE, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit d'une zone d'entrée-sortie pour les appels MQGET, MQPUT et MQPUT1 . La longueur de cette zone est indiquée par MQ_MSG_ID_LENGTH. La valeur initiale de cette zone est MQMI_NONE.

MsgSeqNuméro (MQLONG)
Il s'agit du numéro de séquence d'un message logique au sein d'un groupe.

Les numéros de séquence commencent à 1 et augmentent de 1 pour chaque nouveau message logique du groupe, jusqu'à un maximum de 999 999 999. Un message physique ne se trouvant pas dans un groupe possède le numéro de séquence 1.

L'application n'a pas besoin de définir cette zone dans l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, MQPMO_LOGICAL_ORDER est spécifié.
- Dans l'appel MQGET, MQMO_MATCH_MSG_SEQ_NUMBER n'est *pas* spécifié.

Il s'agit des méthodes recommandées pour utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application requiert davantage de contrôle ou si l'appel est MQPUT1, l'application doit s'assurer que *MsgSeqNumber* est défini sur une valeur appropriée.

Lors de l'entrée dans les appels MQPUT et MQPUT1 , le gestionnaire de files d'attente utilise la valeur décrite dans Ordre physique dans une file d'attente. Dans la sortie des appels MQPUT et MQPUT1 , le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message.

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur indiquée dans Tableau 506, à la page 365. Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur initiale de cette zone est 1. Cette zone est ignorée si *Version* est inférieur à MQMD_VERSION_2.

MsgType (MQLONG)

Indique le type du message. Les types de message sont regroupés comme suit:

MQMT_SYSTEM_FIRST
Valeur la plus basse pour les types de message définis par le système.

MQMT_SYSTEM_LAST
Valeur la plus élevée pour les types de message définis par le système.

Les valeurs suivantes sont actuellement définies dans la plage système:

MQMT_DATAGRAM
Le message ne requiert pas de réponse.

MQMT_REQUEST
Le message requiert une réponse.

Indiquez le nom de la file d'attente à laquelle envoyer la réponse dans la zone *ReplyToQ* . La zone *Report* indique comment définir les valeurs *MsgId* et *CorrelId* de la réponse.

MQMT_REPLY
Le message correspond à la réponse à un message de demande antérieur (MQMT_REQUEST). Le message doit être envoyé à la file d'attente indiquée par la zone *ReplyToQ* du message de demande. Utilisez la zone *Report* de la demande pour contrôler comment définir les valeurs *MsgId* et *CorrelId* de la réponse.

Remarque : Le gestionnaire de files d'attente n'applique pas la relation de demande / réponse ; il s'agit d'une responsabilité d'application.

MQMT_REPORT
Le message signale une occurrence attendue ou inattendue, généralement liée à un autre message (par exemple, un message de demande contenant des données non valides a été reçu). Envoyez le message à la file d'attente indiquée par la zone *ReplyToQ* du descripteur de message du message d'origine. Définissez les zones *Feedback* pour indiquer la nature du rapport. Utilisez la zone *Report* du message d'origine pour contrôler comment définir les valeurs *MsgId* et *CorrelId* du message de rapport.

Les messages de rapport générés par le gestionnaire de files d'attente ou l'agent MCA sont toujours envoyés à la file d'attente *ReplyToQ* , avec les zones *Feedback* et *CorrelId* définies comme indiqué ci-dessus.

Les valeurs définies par l'application peuvent également être utilisées. Ils doivent être compris dans la plage suivante:

MQMT_APPL_FIRST
Valeur la plus faible pour les types de message définis par l'application.

MQMT_APPL_LAST
Valeur la plus élevée pour les types de message définis par l'application.

Pour les appels MQPUT et MQPUT1 , la valeur *MsgType* doit être comprise dans la plage définie par le système ou dans la plage définie par l'application ; si ce n'est pas le cas, l'appel échoue avec le code anomalie MQRC_MSG_TYPE_ERROR.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQMT_DATAGRAM.

Décalage (MQLONG)

Il s'agit du décalage en octets des données du message physique à partir du début du message logique dont font partie les données. Ces données sont appelées un *segment*. Le décalage est compris entre 0 et 999 999 999. Un message physique qui n'est pas un segment d'un message logique a un décalage de zéro.

L'application n'a pas besoin de définir cette zone dans l'appel MQPUT ou MQGET si:

- Dans l'appel MQPUT, MQPMO_LOGICAL_ORDER est spécifié.
- Dans l'appel MQGET, MQMO_MATCH_OFFSET n'est *pas* spécifié.

Il s'agit des méthodes recommandées pour utiliser ces appels pour les messages qui ne sont pas des messages de rapport. Toutefois, si l'application ne respecte pas ces conditions ou si l'appel est MQPUT1, l'application doit s'assurer que *Offset* est défini sur une valeur appropriée.

Lors de l'entrée dans les appels MQPUT et MQPUT1 , le gestionnaire de files d'attente utilise la valeur décrite dans Ordre physique dans une file d'attente. Dans la sortie des appels MQPUT et MQPUT1 , le gestionnaire de files d'attente définit cette zone sur la valeur qui a été envoyée avec le message.

Pour un message de rapport signalant un segment d'un message logique, la zone *OriginalLength* (à condition qu'elle ne soit pas MQOL_UNDEFINED) est utilisée pour mettre à jour le décalage dans les informations de segment conservées par le gestionnaire de files d'attente.

En entrée de l'appel MQGET, le gestionnaire de files d'attente utilise la valeur indiquée dans Tableau 506, à la page 365. Dans la sortie de l'appel MQGET, le gestionnaire de files d'attente définit cette zone sur la valeur du message extrait.

La valeur initiale de cette zone est zéro. Cette zone est ignorée si *Version* est inférieur à MQMD_VERSION_2.

OriginalLength (MQLONG)

Cette zone est pertinente uniquement pour les messages de rapport qui sont des segments. Il indique la longueur du segment de message auquel se rapporte le message de rapport ; il ne précise pas la longueur du message logique dont fait partie le segment, ni la longueur des données du message de rapport.

Remarque : Lors de la génération d'un message de rapport pour un message qui est un segment, le gestionnaire de files d'attente et l'agent de canal de message sont copiés dans le MQMD pour le message de rapport dans les zones *GroupId*, *MsgSeqNumber*, *Offset*et *MsgFlags*du message d'origine. Par conséquent, le message de rapport est également un segment. Les applications qui génèrent des messages de rapport doivent faire de même et définir correctement la zone *OriginalLength* .

La valeur spéciale suivante est définie:

MQOL_UNDEFINI
Longueur d'origine du message non définie.

OriginalLength est une zone d'entrée dans les appels MQPUT et MQPUT1 , mais la valeur fournie par l'application n'est acceptée que dans des circonstances particulières:

- Si le message en cours d'insertion est un segment et est également un message de rapport, le gestionnaire de files d'attente accepte la valeur spécifiée. La valeur doit être:
  - Supérieur à zéro si le segment n'est pas le dernier segment
  - Pas moins de zéro si le segment est le dernier segment

– Pas moins que la longueur des données présentes dans le message

Si ces conditions ne sont pas satisfaites, l'appel échoue avec le code anomalie MQRC_ORIGINAL_LENGTH_ERROR.

- Si le message en cours d'insertion est un segment mais pas un message de rapport, le gestionnaire de files d'attente ignore la zone et utilise à la place la longueur des données de message d'application.
- Dans tous les autres cas, le gestionnaire de files d'attente ignore la zone et utilise la valeur MQOL_UNDEFINED à la place.

Il s'agit d'une zone de sortie de l'appel MQGET.

La valeur initiale de cette zone est MQOL_UNDEFINED. Cette zone est ignorée si *Version* est inférieur à MQMD_VERSION_2.

Persistance (MQLONG)

Indique si le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Pour les appels MQPUT et MQPUT1 , la valeur doit être l'une des suivantes:

MQPER_PERSISTANT
Le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Une fois que le message a été inséré et que l'unité de travail dans laquelle il a été inséré a été validée (si le message est inséré dans une unité de travail), le message est conservé dans la mémoire secondaire. Il y reste jusqu'à ce que le message soit supprimé de la file d'attente et que l'unité de travail dans laquelle il a été obtenu ait été validée (si le message est extrait dans le cadre d'une unité de travail).

Lorsqu'un message persistant est envoyé à une file d'attente éloignée, un mécanisme de stockage et de transfert conserve le message sur chaque gestionnaire de files d'attente le long de la route menant à la destination, jusqu'à ce que le message soit connu comme étant arrivé sur le gestionnaire de files d'attente suivant.

Les messages persistants ne peuvent pas être placés sur:

- Files d'attente dynamiques temporaires
- Files d'attente partagées qui sont mappées à un objet CFSTRUCT au niveau CFLEVEL (2) ou à un niveau inférieur, ou où l'objet CFSTRUCT est défini en tant que RECOVER (NO).

Les messages persistants peuvent être placés dans des files d'attente dynamiques permanentes et dans des files d'attente prédéfinies.

MQPER_NON_PERSISTENT
Le message ne survit généralement pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente. Cela s'applique même si une copie intacte du message est détectée sur la mémoire secondaire lors du redémarrage du gestionnaire de files d'attente.

Dans le cas des files d'attente NPMCLASS (HIGH), les messages non persistants survivent à un arrêt et à un redémarrage normaux du gestionnaire de files d'attente.

Dans le cas de files d'attente partagées, les messages non persistants survivent aux redémarrages du gestionnaire de files d'attente dans le groupe de partage de files d'attente, mais ne survivent pas aux échecs de l'unité de couplage utilisée pour stocker les messages dans les files d'attente partagées.

MQPER_PERSISTENCE_AS_Q_DEF

- Si la file d'attente est une file d'attente de cluster, la persistance du message est extraite de l'attribut *DefPersistence* défini sur le gestionnaire de files d'attente *destination* qui possède l'instance particulière de la file d'attente dans laquelle le message est placé. En général, toutes les instances d'une file d'attente de cluster ont la même valeur pour l'attribut *DefPersistence* , bien que cela ne soit pas obligatoire.

  La valeur de *DefPersistence* est copiée dans la zone *Persistence* lorsque le message est placé dans la file d'attente de destination. Si *DefPersistence* est modifié ultérieurement, les messages qui ont déjà été placés dans la file d'attente ne sont pas affectés.

- Si la file d'attente n'est pas une file d'attente de cluster, la persistance du message est extraite de l'attribut *DefPersistence* défini sur le gestionnaire de files d'attente *local* , même si le gestionnaire de files d'attente de destination est distant.

  S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la persistance par défaut est extraite de la valeur de cet attribut dans la *première* définition du chemin. Ce nom peut être :

  – Une file d'attente alias
  – Une file d'attente locale
  – Définition locale d'une file d'attente éloignée
  – Un alias de gestionnaire de files d'attente
  – Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName*)

  La valeur de *DefPersistence* est copiée dans la zone *Persistence* lorsque le message est inséré. Si *DefPersistence* est modifié par la suite, les messages qui ont déjà été insérés ne sont pas affectés.

Les messages persistants et non persistants peuvent exister dans la même file d'attente.

Lors de la réponse à un message, les applications doivent utiliser la persistance du message de demande pour le message de réponse.

Pour un appel MQGET, la valeur renvoyée est MQPER_PERSISTENT ou MQPER_NOT_PERSISTENT.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQPER_PERSISTENCE_AS_Q_DEF.

Priorité (MQLONG)

Pour les appels MQPUT et MQPUT1 , la valeur doit être supérieure ou égale à zéro ; zéro est la priorité la plus basse. La valeur spéciale suivante peut également être utilisée:

MQPRI_PRIORITY_AS_Q_DEF

- Si la file d'attente est une file d'attente de cluster, la priorité du message est extraite de l'attribut *DefPriority* défini sur le gestionnaire de files d'attente *destination* qui possède l'instance particulière de la file d'attente dans laquelle le message est placé. En général, toutes les instances d'une file d'attente de cluster ont la même valeur pour l'attribut *DefPriority* , bien que cela ne soit pas obligatoire.

  La valeur de *DefPriority* est copiée dans la zone *Priority* lorsque le message est placé dans la file d'attente de destination. Si *DefPriority* est modifié ultérieurement, les messages qui ont déjà été placés dans la file d'attente ne sont pas affectés.
- Si la file d'attente n'est pas une file d'attente de cluster, la priorité du message est extraite de l'attribut *DefPriority* défini sur le gestionnaire de files d'attente *local* , même si le gestionnaire de files d'attente de destination est éloigné.

  S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité par défaut est extraite de la valeur de cet attribut dans la *première* définition du chemin. Ce nom peut être :

  – Une file d'attente alias
  – Une file d'attente locale
  – Définition locale d'une file d'attente éloignée
  – Un alias de gestionnaire de files d'attente
  – Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName*)

  La valeur de *DefPriority* est copiée dans la zone *Priority* lorsque le message est inséré. Si *DefPriority* est modifié par la suite, les messages qui ont déjà été insérés ne sont pas affectés.

La valeur renvoyée par l'appel MQGET est toujours supérieure ou égale à zéro ; la valeur MQPRI_PRIORITY_AS_Q_DEF n'est jamais renvoyée.

Si un message est inséré avec une priorité supérieure à la valeur maximale prise en charge par le gestionnaire de files d'attente local (cette valeur maximale est donnée par l'attribut de gestionnaire de files d'attente *MaxPriority*), le message est accepté par le gestionnaire de files d'attente, mais placé dans la file d'attente avec la priorité maximale du gestionnaire de files d'attente ; l'appel MQPUT ou MQPUT1 se termine avec MQCC_WARNING et le code anomalie MQRC_PRIORITY_DÉPASSS_MAXIMUM. Toutefois, la zone *Priority* conserve la valeur spécifiée par l'application qui a inséré le message.

Sous z/OS, si un message avec un numéro de message MsgSeqégal à 1 est inséré dans une file d'attente dont la séquence de distribution des messages est MQMDS_PRIORITY et dont le type d'index est MQIT_GROUP_ID, la file d'attente peut traiter le message avec une priorité différente. Si le message a été placé dans la file d'attente avec une priorité de 0 ou 1, il est traité comme s'il avait une priorité de 2. En effet, l'ordre des messages placés dans ce type de file d'attente est optimisé pour permettre des tests de complétude de groupe efficaces. Pour plus d'informations sur la séquence de distribution des messages MQMDS_PRIORITY et le type d'index MQIT_GROUP_ID, voir Attribut de séquenceMsgDelivery.

Lors de la réponse à un message, les applications doivent utiliser la priorité du message de demande pour le message de réponse. Dans d'autres cas, la spécification de MQPRI_PRIORITY_AS_Q_DEF permet d'optimiser les priorités sans modifier l'application.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQPRI_PRIORITY_AS_Q_DEF.

PutApplNom (MQCHAR28)
Il s'agit du nom de l'application qui a inséré le message et qui fait partie du *contexte d'origine* du message. Le contenu diffère d'une plateforme à l'autre et peut varier d'une édition à l'autre.

Pour plus d'informations sur le contexte de message, voir «Présentation de MQMD», à la page 395 et Contexte de message .

Le format de *PutApplName* dépend de la valeur de *PutApplType* et peut varier d'une édition à l'autre. Les modifications sont rares, mais elles se produisent si l'environnement change.

Lorsque le gestionnaire de files d'attente définit cette zone (c'est-à-dire, pour toutes les options à l'exception de MQPMO_SET_ALL_CONTEXT), il définit la zone sur une valeur déterminée par l'environnement:

- Sous z/OS, le gestionnaire de files d'attente utilise:
  - Pour le lot z/OS , nom de travail à 8 caractères provenant de la carte de travail JES
  - Pour TSO, l'ID utilisateur TSO à 7 caractères
  - Pour CICS, ID application à 8 caractères, suivi de l'ID transaction à 4 caractères
  - Pour IMS, identificateur de système IMS à 8 caractères, suivi du nom de bloc de spécification de programme à 8 caractères
  - Pour XCF, nom de groupe XCF à 8 caractères, suivi du nom de membre XCF à 16 caractères
  - Pour un message généré par un gestionnaire de files d'attente, les 28 premiers caractères du nom du gestionnaire de files d'attente
  - Pour la mise en file d'attente répartie sans CICS, nom de travail à 8 caractères de l'initiateur de canal suivi du nom à 8 caractères du module placé dans la file d'attente de rebut, suivi d'un identificateur de tâche à 8 caractères.

  Le ou les noms sont chacun complétés à droite par des blancs, comme n'importe quel espace dans le reste de la zone. Lorsqu'il y a plusieurs noms, il n'y a pas de séparateur entre eux.
- Sur les systèmes Windows , le gestionnaire de files d'attente utilise:
  - Pour une application CICS , le nom de transaction CICS
  - Pour une application non CICS, les 28 caractères les plus à droite du nom qualifié complet de l'exécutable

- Sous IBM i, le gestionnaire de files d'attente utilise le nom de travail qualifié complet.
- Sur les systèmes UNIX , le gestionnaire de files d'attente utilise:
  - Pour une application CICS , le nom de transaction CICS
  - Pour une application non CICS, MQ demande le nom du processus au système d'exploitation. Ce nom est renvoyé en tant que nom de fichier programme, sans chemin d'accès complet. Ensuite, MQ place ce nom de processus dans MQMD.PutApplName de PutApplName se présente comme suit:

    AIX
    Si le nom est inférieur ou égal à 28 octets, le nom est inséré, complété à droite par des espaces.

    Si le nom est supérieur à 28 octets, les 28 octets les plus à gauche du nom sont insérés.

    Linux et Solaris
    Si le nom est inférieur ou égal à 15 octets, le nom est inséré, complété à droite par des espaces.

    Si le nom est supérieur à 15 octets, les 15 octets les plus à gauche du nom sont insérés, complétés à droite par des espaces.

    HP-UX
    Si le nom est inférieur ou égal à 14 octets, le nom est inséré, complété à droite par des espaces.

    Si le nom est supérieur à 14 octets, les 14 octets les plus à gauche du nom sont insérés, complétés à droite par des espaces.

    Par exemple, si vous exécutez `/opt/mqm/samp/bin/amqsput QNAME QMNAME`, le nom PutApplest `'amqsput'`. Cette zone MQCHAR28 contient 21 caractères d'espace de remplissage. Notez que le chemin d'accès complet incluant `/opt/mqm/samp/bin` n'est pas inclus dans le nom PutAppl.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO_SET_ALL_CONTEXT est spécifié dans le paramètre *PutMsgOpts* . Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le caractère null et tous les caractères suivants sont convertis en blancs par le gestionnaire de files d'attente. Si MQPMO_SET_ALL_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Type PutAppl(MQLONG)

Il s'agit du type d'application qui a inséré le message et qui fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir «Présentation de MQMD», à la page 395 et Contexte de message.

PutApplType peut avoir l'un des types standard suivants. Vous pouvez également définir vos propres types, mais uniquement avec des valeurs comprises entre MQAT_USER_FIRST et MQAT_USER_LAST.

MQAT_AIX
Application AIX (même valeur que MQAT_UNIX).

MQAT_COURTIER
Courtier.

MQAT_CICS
Transaction CICS .

MQAT_CICS_BRIDGE
Pont CICS .

MQAT_CICS_VSE
Transaction CICS/VSE .

MQAT_DOS
WebSphere MQ Application client MQI sur PC DOS.

MQAT_DQM
Agent de gestionnaire de files d'attente réparties.

MQAT_GARDIEN
Application Tandem Guardian (même valeur que MQAT_NSK).

MQAT_IMS
Application IMS .

MQAT_IMS_BRIDGE
Passerelle IMS .

MQAT_JAVA
Java.

MQAT_MVS
Application MVS ou TSO (même valeur que MQAT_ZOS).

MQAT_NOTES_AGENT
Lotus Notes Application Agent.

MQAT_NSK
Application HP Integrity NonStop Server .

MQAT_OS390
Application OS/390 (même valeur que MQAT_ZOS).

MQAT_OS400
Application IBM i .

MQAT_QMGR
Gestionnaire de files d'attente.

MQAT_UNIX
Application UNIX .

MQAT_VOS
Application Stratus VOS.

MQAT_WINDOWS
Application Windows 16 bits.

MQAT_WINDOWS_NT
Application Windows 32 bits.

MQAT_WLM
Application de gestionnaire de charge de travail z/OS .

MQAT_XCF
XCF.

MQAT_ZOS
Application z/OS .

MQAT_PAR DEFAUT
Type d'application par défaut.

Il s'agit du type d'application par défaut pour la plateforme sur laquelle l'application s'exécute.

Remarque : La valeur de cette constante est spécifique à l'environnement. Pour cette raison, compilez toujours l'application à l'aide des fichiers d'en-tête, d'inclusion ou COPY appropriés à la plateforme sur laquelle l'application s'exécutera.

MQAT_INCONNU
Utilisez cette valeur pour indiquer que le type d'application est inconnu, même si d'autres informations de contexte sont présentes.

MQAT_USER_FIRST
Valeur la plus faible pour le type d'application défini par l'utilisateur.

MQAT_USER_LAST
Valeur la plus élevée pour le type d'application défini par l'utilisateur.

La valeur spéciale suivante peut également se produire:

MQAT_NO_CONTEXT
Cette valeur est définie par le gestionnaire de files d'attente lorsqu'un message est inséré sans contexte (c'est-à-dire que l'option de contexte MQPMO_NO_CONTEXT est spécifiée).

Lorsqu'un message est extrait, *PutApplType* peut être testé pour cette valeur afin de déterminer si le message a un contexte (il est recommandé que *PutApplType* ne soit jamais défini sur MQAT_NO_CONTEXT, par une application utilisant MQPMO_SET_ALL_CONTEXT, si l'une des autres zones de contexte n'est pas vide).

Lorsque le gestionnaire de files d'attente génère ces informations à la suite d'une insertion d'application, la zone est définie sur une valeur déterminée par l'environnement. Sous IBM i, il est défini sur MQAT_OS400; le gestionnaire de files d'attente n'utilise jamais MQAT_CICS sous IBM i.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO_SET_ALL_CONTEXT est spécifié dans le paramètre *PutMsgOpts* . Si MQPMO_SET_ALL_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La valeur initiale de cette zone est MQAT_NO_CONTEXT.

PutDate (MQCHAR8)

Il s'agit de la date à laquelle le message a été inséré et qui fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir «Présentation de MQMD», à la page 395 et Contexte de message.

Le format utilisé pour la date à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- AAAAMMJJ

où les caractères représentent:

AAAA
année (quatre chiffres)

MM
mois de l'année (01 à 12)

JJ
jour du mois (01 à 31)

Le temps moyen de Greenwich (GMT) est utilisé pour les zones *PutDate* et *PutTime* , à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

Si le message a été inséré dans une unité de travail, la date est celle à laquelle le message a été inséré et non celle à laquelle l'unité de travail a été validée.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO_SET_ALL_CONTEXT est spécifié dans le paramètre *PutMsgOpts* . Le contenu de la zone n'est pas vérifié par le gestionnaire de files d'attente, sauf que les informations qui suivent un caractère null dans la zone sont supprimées. Le gestionnaire de files d'attente convertit le caractère null et les caractères suivants en blancs. Si MQPMO_SET_ALL_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ_PUT_DATE_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 8 caractères blancs dans les autres langages de programmation.

PutTime (MQCHAR8)

Il s'agit de l'heure à laquelle le message a été inséré et il fait partie du **contexte d'origine** du message. Pour plus d'informations sur le contexte de message, voir «Présentation de MQMD», à la page 395 et Contexte de message.

Le format utilisé pour l'heure à laquelle cette zone est générée par le gestionnaire de files d'attente est le suivant:

- HHMMSSTH

où les caractères représentent (dans l'ordre):

hh
heures (00 à 23)

MM
minutes (00 à 59)

SS
secondes (00 à 59 ; voir remarque)

T
dixièmes de seconde (0 à 9)

H
centièmes de seconde (0 à 9)

Remarque : Si l'horloge système est synchronisée selon une norme de temps très précise, il est possible en de rares occasions que 60 ou 61 soient renvoyés pour les secondes dans *PutTime*. Cela se produit lorsque des secondes bissextiles sont insérées dans la norme de temps globale.

Le temps moyen de Greenwich (GMT) est utilisé pour les zones *PutDate* et *PutTime* , à condition que l'horloge système soit définie avec précision sur le temps moyen de Greenwich.

Si le message a été inséré dans une unité de travail, l'heure est celle à laquelle le message a été inséré et non celle à laquelle l'unité de travail a été validée.

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO_SET_ALL_CONTEXT est spécifié dans le paramètre *PutMsgOpts* . Le gestionnaire de files d'attente ne vérifie pas le contenu de la zone, sauf que les informations qui suivent un caractère null dans la zone sont supprimées. Le gestionnaire de files d'attente convertit le caractère null et les caractères suivants en blancs. Si MQPMO_SET_ALL_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ_PUT_TIME_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 8 caractères blancs dans les autres langages de programmation.

ReplyToQ (MQCHAR48)

Il s'agit du nom de la file d'attente de messages à laquelle l'application qui a émis la demande d'obtention du message envoie les messages MQMT_REPLY et MQMT_REPORT. Il s'agit du nom local d'une file d'attente définie sur le gestionnaire de files d'attente identifié par *ReplyToQMgr*. Cette file d'attente ne doit pas être une file d'attente modèle, bien que le gestionnaire de files d'attente émetteur ne la vérifie pas lorsque le message est inséré.

Pour les appels MQPUT et MQPUT1 , cette zone ne doit pas être vide si la zone *MsgType* a la valeur MQMT_REQUEST ou si des messages de rapport sont demandés par la zone *Report* . Toutefois, la valeur spécifiée (ou remplacée) est transmise à l'application qui émet la demande d'obtention du message, quel que soit le type de message.

Si la zone *ReplyToQMgr* est vide, le gestionnaire de files d'attente local recherche le nom *ReplyToQ* dans ses propres définitions de file d'attente. S'il existe une définition locale d'une file d'attente éloignée portant ce nom, la valeur *ReplyToQ* dans le message transmis est remplacée par la valeur de l'attribut *RemoteQName* de la définition de la file d'attente éloignée et cette valeur est renvoyée dans le descripteur de message lorsque l'application de réception émet un appel MQGET pour le message. Si une définition locale d'une file d'attente éloignée n'existe pas, *ReplyToQ* n'est pas modifié.

Si le nom est indiqué, il peut contenir des blancs de fin ; le premier caractère nul et les caractères qui le suivent sont traités comme des blancs. Sinon, il n'est pas vérifié que le nom respecte les règles de dénomination des files d'attente ; cela est également vrai pour le nom transmis, si *ReplyToQ* est remplacé dans le message transmis. La seule vérification effectuée est qu'un nom a été spécifié, si les circonstances l'exigent.

Si une file d'attente de réponse n'est pas requise, définissez la zone *ReplyToQ* sur des blancs ou (dans le langage de programmation C) sur la chaîne NULL ou sur un ou plusieurs blancs suivis d'un caractère NULL ; ne laissez pas la zone non initialisée.

Pour l'appel MQGET, le gestionnaire de files d'attente renvoie toujours le nom complété par des blancs à la longueur de la zone.

Si un message nécessitant un message de rapport ne peut pas être distribué et que le message de rapport ne peut pas non plus être distribué dans la file d'attente spécifiée, le message d'origine et le message de rapport sont placés dans la file d'attente de rebut (message non distribué) (voir l'attribut *DeadLetterQName* décrit dans «Attributs du gestionnaire de files d'attente», à la page 787).

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

ReplyToGestionnaire de files d'attente (MQCHAR48)

Il s'agit du nom du gestionnaire de files d'attente auquel le message de réponse ou de rapport doit être envoyé. *ReplyToQ* est le nom local d'une file d'attente définie sur ce gestionnaire de files d'attente.

Si la zone *ReplyToQMgr* est vide, le gestionnaire de files d'attente local recherche le nom *ReplyToQ* dans ses définitions de file d'attente. S'il existe une définition locale d'une file d'attente éloignée portant ce nom, la valeur *ReplyToQMgr* dans le message transmis est remplacée par la valeur de l'attribut *RemoteQMgrName* de la définition de la file d'attente éloignée et cette valeur est renvoyée dans le descripteur de message lorsque l'application de réception émet un appel MQGET pour le message. S'il n'existe pas de définition locale d'une file d'attente éloignée, le *ReplyToQMgr* transmis avec le message est le nom du gestionnaire de files d'attente local.

Si le nom est indiqué, il peut contenir des blancs de fin ; le premier caractère nul et les caractères qui le suivent sont traités comme des blancs. Sinon, il n'est pas vérifié que le nom respecte les règles de dénomination des gestionnaires de files d'attente ou que ce nom est connu du gestionnaire de files d'attente émetteur ; cela est également vrai pour le nom transmis, si *ReplyToQMgr* est remplacé dans le message transmis.

Si une file d'attente de réponse n'est pas requise, définissez la zone *ReplyToQMgr* sur des blancs ou (dans le langage de programmation C) sur la chaîne NULL ou sur un ou plusieurs blancs suivis d'un caractère NULL ; ne laissez pas la zone non initialisée.

Pour l'appel MQGET, le gestionnaire de files d'attente renvoie toujours le nom complété par des blancs à la longueur de la zone.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

Rapport (MQLONG)

Un message de rapport est un message relatif à un autre message, utilisé pour informer une application des événements attendus ou inattendus liés au message d'origine. La zone *Report* permet à l'application qui envoie le message d'origine de spécifier quels sont les messages de rapport requis, si les données de message d'application doivent y être incluses, ainsi que (pour les rapports et les réponses) la manière dont les identificateurs de message et de corrélation du message de rapport ou de réponse doivent être définis. Tout ou partie (ou aucun) des types de message de rapport suivants peut être demandé:

- Exception
- Expiration dans
- Confirmer à l'arrivée (COA)
- Confirmer à la livraison (COD)
- Notification d'action positive (PAN)
- Notification d'action négative (NAN)

Si plusieurs types de message de rapport sont requis, ou si d'autres options de rapport sont nécessaires, les valeurs peuvent être:

- Ajouté ensemble (ne pas ajouter la même constante plus d'une fois), ou
- combinées par le biais de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations par bit).

L'application qui reçoit le message de rapport peut déterminer la raison pour laquelle le rapport a été généré en examinant la zone *Feedback* dans le MQMD ; voir la zone *Feedback* pour plus de détails.

L'utilisation des options de rapport lors de l'insertion d'un message dans une rubrique peut entraîner la génération de zéro, un ou plusieurs messages de rapport et leur envoi à l'application. En effet, le message de publication peut être envoyé à zéro, une ou plusieurs applications d'abonnement.

Options d'exception: spécifiez l'une des options répertoriées pour demander un message de rapport d'exception.

MQRO_EXCEPTION
Un agent MCA génère ce type de rapport lorsqu'un message est envoyé à un autre gestionnaire de files d'attente et qu'il ne peut pas être distribué à la file d'attente de destination spécifiée. Par exemple, la file d'attente de destination ou une file d'attente de transmission intermédiaire peut être saturée ou le message peut être trop volumineux pour la file d'attente.

La génération du message de rapport d'exception dépend de la persistance du message d'origine et de la vitesse du canal de transmission des messages (normal ou rapide) par lequel le message d'origine est transmis:

- Pour tous les messages persistants et pour les messages non persistants qui transitent par des canaux de messages normaux, le rapport d'exception est généré *uniquement* si l'action spécifiée par l'application émettrice pour la condition d'erreur peut aboutir. L'application émettrice peut spécifier l'une des actions suivantes pour contrôler la disposition du message d'origine lorsque la condition d'erreur se produit:
  - MQRO_DEAD_LETTER_Q (le message d'origine est placé dans la file d'attente des messages non livrés).
  - MQRO_DISCARD_MSG (supprime le message d'origine).

  Si l'action spécifiée par l'application émettrice ne peut pas aboutir, le message d'origine est laissé dans la file d'attente de transmission et aucun message de rapport d'exception n'est généré.
- Pour les messages non persistants qui transitent par des canaux de messages rapides, le message d'origine est supprimé de la file d'attente de transmission et le rapport d'exception est généré *même si* l'action spécifiée pour la condition d'erreur ne peut pas aboutir. Par exemple, si MQRO_DEAD_LETTER_Q est spécifié, mais que le message d'origine ne peut pas être placé dans la file d'attente de rebut car cette file d'attente est saturée, le message de rapport d'exception est généré et le message d'origine est supprimé.

  Pour plus d'informations sur les canaux de messages normaux et rapides, voir Vitesse de messages non persistants (NPMSPEED) .

Aucun rapport d'exception n'est généré si l'application qui a inséré le message d'origine peut être avertie de manière synchrone de l'incident au moyen du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 .

Les applications peuvent également envoyer des rapports d'exception pour indiquer qu'un message ne peut pas être traité (par exemple, parce qu'il s'agit d'une transaction de débit qui entraînerait le dépassement de la limite de crédit du compte).

Les données du message d'origine ne sont pas incluses dans le message de rapport.

N'indiquez pas plus d'une seule des valeurs suivantes: MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA et MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_EXCEPTION_WITH_DATA
Identique à MQRO_EXCEPTION, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ , elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

N'indiquez pas plus d'une seule des valeurs suivantes: MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA et MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_EXCEPTION_WITH_FULL_DATA
Des rapports d'exception avec des données complètes sont requis.

Identique à MQRO_EXCEPTION, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

N'indiquez pas plus d'une seule des valeurs suivantes: MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA et MQRO_EXCEPTION_WITH_FULL_DATA.

Options d'expiration: spécifiez l'une des options répertoriées pour demander un message de rapport d'expiration.

MQRO_EXPIRATION
Ce type de rapport est généré par le gestionnaire de files d'attente si le message est supprimé avant la livraison à une application en raison du délai d'expiration (voir la zone *Expiry*). Si cette option n'est pas définie, aucun message de rapport n'est généré si un message est supprimé pour cette raison (même si vous spécifiez l'une des options MQRO_EXCEPTION_ *).

Les données du message d'origine ne sont pas incluses dans le message de rapport.

N'indiquez pas plus d'une des valeurs suivantes: MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA et MQRO_EXPIRATION_WITH_FULL_DATA.

MQRO_EXPIRATION_WITH_DATA
Identique à MQRO_EXPIRATION, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ , elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

N'indiquez pas plus d'une des valeurs suivantes: MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA et MQRO_EXPIRATION_WITH_FULL_DATA.

MQRO_EXPIRATION_WITH_FULL_DATA
Identique à MQRO_EXPIRATION, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

N'indiquez pas plus d'une des valeurs suivantes: MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA et MQRO_EXPIRATION_WITH_FULL_DATA.

Options de confirmation à l'arrivée: indiquez l'une des options répertoriées pour demander un message de rapport de confirmation à l'arrivée.

MQRO_COA
Ce type de rapport est généré par le gestionnaire de files d'attente propriétaire de la file d'attente de destination lorsque le message est placé dans la file d'attente de destination. Les données du message d'origine ne sont pas incluses dans le message de rapport.

Si le message est inséré dans une unité de travail et que la file d'attente de destination est une file d'attente locale, le message de rapport COA généré par le gestionnaire de files d'attente peut être extrait uniquement si l'unité de travail est validée.

Aucun rapport COA n'est généré si la zone *Format* du descripteur de message est MQFMT_XMIT_Q_HEADER ou MQFMT_DEAD_LETTER_HEADER. Cela empêche la génération d'un rapport COA si le message est placé dans une file d'attente de transmission ou s'il n'est pas livrable et placé dans une file d'attente de rebut.

Dans le cas d'une file d'attente de pont IMS , le rapport COA est généré lorsque le message atteint la file d'attente IMS (accusé de réception reçu d' IMS) et non lorsque le message est inséré dans la file d'attente de pont MQ . Cela signifie que si IMS n'est pas actif, aucun rapport COA n'est généré tant que IMS n'est pas démarré et qu'un message est mis en file d'attente dans la file d'attente IMS .

Utilisateur qui exécute un programme qui insère un message avec MQMD.Report= MQRO_COA doit disposer des droits d'accès + passid sur la file d'attente de réponses. Si l'utilisateur ne dispose

pas des droits + passid, le message de rapport COA n'atteint pas la file d'attente de réponses. Une tentative est effectuée pour placer le message de rapport dans la file d'attente des messages non livrés.

N'indiquez pas plus d'un des éléments MQRO_COA, MQRO_COA_WITH_DATA et MQRO_COA_WITH_FULL_DATA.

MQRO_COA_WITH_DATA
Il est identique à MQRO_COA, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ , elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

N'indiquez pas plus d'un des éléments MQRO_COA, MQRO_COA_WITH_DATA et MQRO_COA_WITH_FULL_DATA.

MQRO_COA_WITH_FULL_DATA
Identique à MQRO_COA, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

N'indiquez pas plus d'un des éléments MQRO_COA, MQRO_COA_WITH_DATA et MQRO_COA_WITH_FULL_DATA.

Options de confirmation de livraison: spécifiez l'une des options répertoriées pour demander un message de rapport de confirmation de livraison.

MQRO_COD
Ce type de rapport est généré par le gestionnaire de files d'attente lorsqu'une application extrait le message de la file d'attente de destination d'une manière qui supprime le message de la file d'attente. Les données du message d'origine ne sont pas incluses dans le message de rapport.

Si le message est extrait dans le cadre d'une unité de travail, le message de rapport est généré dans la même unité de travail, de sorte que le rapport n'est pas disponible tant que l'unité de travail n'est pas validée. Si l'unité de travail est annulée, le rapport n'est pas envoyé.

Un rapport COD n'est pas toujours généré si un message est extrait avec l'option MQGMO_MARK_SKIP_BACKOUT. Si l'unité de travail principale est annulée mais que l'unité de travail secondaire est validée, le message est supprimé de la file d'attente, mais aucun rapport COD n'est généré.

Aucun rapport COD n'est généré si la zone *Format* du descripteur de message est MQFMT_DEAD_LETTER_HEADER. Cela empêche la génération d'un rapport COD si le message n'est pas livrable et qu'il est placé dans une file d'attente de rebut.

MQRO_COD n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un MQRO_COD, MQRO_COD_WITH_DATA et MQRO_COD_WITH_FULL_DATA.

MQRO_COD_WITH_DATA
Cette valeur est identique à MQRO_COD, sauf que les 100 premiers octets des données de message d'application du message d'origine sont inclus dans le message de rapport. Si le message d'origine contient une ou plusieurs structures d'en-tête MQ , elles sont incluses dans le message de rapport, en plus des 100 octets de données d'application.

Si MQGMO_ACCEPT_TRUNCATED_MSG est spécifié dans l'appel MQGET pour le message d'origine et que le message extrait est tronqué, la quantité de données de message d'application placée dans le message de rapport dépend de l'environnement:

- Sous z/OS, il s'agit de la valeur minimale de:
  - Longueur du message d'origine
  - Longueur de la mémoire tampon utilisée pour extraire le message
  - 100 octets.
- Dans les autres environnements, il s'agit du minimum de:

- Longueur du message d'origine
- 100 octets.

MQRO_COD_WITH_DATA n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un MQRO_COD, MQRO_COD_WITH_DATA et MQRO_COD_WITH_FULL_DATA.

MQRO_COD_WITH_FULL_DATA
Cette valeur est identique à MQRO_COD, sauf que toutes les données de message d'application du message d'origine sont incluses dans le message de rapport.

MQRO_COD_WITH_FULL_DATA n'est pas valide si la file d'attente de destination est une file d'attente XCF.

N'indiquez pas plus d'un MQRO_COD, MQRO_COD_WITH_DATA et MQRO_COD_WITH_FULL_DATA.

Options de notification d'action: spécifiez l'une des options répertoriées ou les deux pour demander à l'application réceptrice d'envoyer un message de rapport d'action positive ou négative.

MQRO_PAN
Ce type de rapport est généré par l'application qui extrait le message et agit en conséquence. Elle indique que l'action demandée dans le message a été exécutée avec succès. L'application qui génère le rapport détermine si des données doivent être incluses dans le rapport.

A l'exception de la transmission de cette demande à l'application qui extrait le message, le gestionnaire de files d'attente n'effectue aucune action en fonction de cette option. L'application d'extraction doit générer le rapport le cas échéant.

MQRO_NAN
Ce type de rapport est généré par l'application qui extrait le message et agit en conséquence. Il indique que l'action demandée dans le message n'a *pas* été exécutée correctement. L'application qui génère le rapport détermine si des données doivent être incluses dans le rapport. Par exemple, vous pouvez inclure des données indiquant la raison pour laquelle la demande n'a pas pu être exécutée.

A l'exception de la transmission de cette demande à l'application qui extrait le message, le gestionnaire de files d'attente n'effectue aucune action en fonction de cette option. L'application d'extraction doit générer le rapport le cas échéant.

L'application doit déterminer les conditions qui correspondent à une action positive et celles qui correspondent à une action négative. Toutefois, si la requête n'a été que partiellement exécutée, générez un rapport NAN plutôt qu'un rapport PAN si demandé. Chaque condition possible doit correspondre soit à une action positive, soit à une action négative, mais pas les deux.

Options d'identificateur de message: spécifiez l'une des options répertoriées pour contrôler la manière dont le *MsgId* du message de rapport (ou du message de réponse) doit être défini.

MQRO_NEW_MSG_ID
Il s'agit de l'action par défaut qui indique que si un rapport ou une réponse est généré suite à ce message, un nouveau *MsgId* est généré pour le message de rapport ou de réponse.

MQRO_PASS_MSG_ID
Si un rapport ou une réponse est généré à la suite de ce message, le *MsgId* de ce message est copié dans le *MsgId* du message de rapport ou de réponse.

Le *MsgId* d'un message de publication sera différent pour chaque abonné qui reçoit une copie de la publication et, par conséquent, le *MsgId* copié dans le message de rapport ou de réponse sera différent pour chaque abonné.

Si cette option n'est pas spécifiée, MQRO_NEW_MSG_ID est utilisé.

Options d'identificateur de corrélation: spécifiez l'une des options répertoriées pour contrôler la manière dont le *CorrelId* du message de rapport (ou du message de réponse) doit être défini.

MQRO_COPY_MSG_ID_TO_CORREL_ID
Il s'agit de l'action par défaut, qui indique que si un rapport ou une réponse est généré suite à ce message, le *MsgId* de ce message est copié dans le *CorrelId* du message de rapport ou de réponse.

Le *MsgId* d'un message de publication sera différent pour chaque abonné qui reçoit une copie de la publication et, par conséquent, le *MsgId* copié dans le *CorrelId* du message de rapport ou de réponse sera différent pour chacun d'eux.

MQRO_PASS_CORREL_ID
Si un rapport ou une réponse est généré à la suite de ce message, le *CorrelId* de ce message est copié dans le *CorrelId* du message de rapport ou de réponse.

Le *CorrelId* d'un message de publication est spécifique à un abonné, sauf s'il utilise l'option MQSO_SET_CORREL_ID et définit la zone d'ID SubCorreldans le MQSD sur MQCI_NONE. Par conséquent, il est possible que le *CorrelId* copié dans le *CorrelId* du message de rapport ou de réponse soit différent pour chacun d'eux.

Si cette option n'est pas spécifiée, MQRO_COPY_MSG_ID_TO_CORREL_ID est utilisé.

Les serveurs qui répondent aux demandes ou qui génèrent des messages de rapport doivent vérifier si les options MQRO_PASS_MSG_ID ou MQRO_PASS_CORREL_ID ont été définies dans le message d'origine. Si tel est le cas, les serveurs doivent effectuer l'action décrite pour ces options. Si aucun n'est défini, les serveurs doivent effectuer l'action par défaut correspondante.

Options de disposition: Indiquez l'une des options répertoriées pour contrôler la disposition du message d'origine lorsqu'il ne peut pas être distribué à la file d'attente de destination. L'application peut définir les options d'élimination indépendamment de la demande de rapports d'exception.

MQRO_DEAD_LETTER_Q
Il s'agit de l'action par défaut qui place le message dans la file d'attente de rebut si le message ne peut pas être distribué à la file d'attente de destination. Cela se produit dans les situations suivantes:

- Lorsque l'application qui a inséré le message d'origine ne peut pas être informée de manière synchrone du problème à l'aide du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 . Un message de rapport d'exception est généré si l'expéditeur en a demandé un.
- Lorsque l'application qui a inséré le message d'origine a inséré une rubrique

MQRO_DISCARD_MSG
Le message est supprimé s'il ne peut pas être distribué à la file d'attente de destination. Cela se produit dans les situations suivantes:

- Lorsque l'application qui a inséré le message d'origine ne peut pas être informée de manière synchrone du problème à l'aide du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 . Un message de rapport d'exception est généré si l'expéditeur en a demandé un.
- Lorsque l'application qui a inséré le message d'origine a inséré une rubrique

Si vous souhaitez renvoyer le message d'origine à l'expéditeur, sans que le message d'origine soit placé dans la file d'attente de rebut, l'expéditeur doit spécifier MQRO_DISCARD_MSG avec MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_PASS_DISCARD_ET_EXPIRATION
Si cette option est définie sur un message et qu'un rapport ou une réponse est généré en raison de celui-ci, le descripteur de message du rapport hérite:

- MQRO_DISCARD_MSG, s'il a été défini.
- Heure d'expiration restante du message (s'il ne s'agit pas d'un rapport d'expiration). S'il s'agit d'un rapport d'expiration, le délai d'expiration est de 60 secondes.

Option d'activité

ACTIVITE MQRO
L'utilisation de cette valeur permet de tracer la route de **tout** message sur un réseau de gestionnaires de files d'attente. L'option de rapport peut être spécifiée sur n'importe quel message utilisateur en cours, ce qui vous permet instantanément de commencer à calculer la route du message via le réseau.

Si l'application qui génère le message ne peut pas activer les rapports d'activité, les rapports peuvent être activés à l'aide d'un exit de croisement d'API fourni par les administrateurs du gestionnaire de files d'attente.

Remarque :

1. Moins les gestionnaires de files d'attente du réseau sont en mesure de générer des rapports d'activité, moins la route est détaillée.
2. Les rapports d'activité peuvent être difficiles à placer dans le bon ordre pour déterminer la route empruntée.
3. Il se peut que les rapports d'activité ne parviennent pas à trouver une route vers leur destination demandée.
4. Les messages avec ce jeu d'options de rapport doivent être acceptés par n'importe quel gestionnaire de files d'attente, même s'ils ne comprennent pas l'option. Cela permet de définir l'option de rapport sur n'importe quel message utilisateur, même s'ils sont traités par un gestionnaire de files d'attente autre que la version 6.0 ou ultérieure.
5. Si un processus, qu'il s'agisse d'un gestionnaire de files d'attente ou d'un processus utilisateur, effectue une activité sur un message avec cette option définie, il peut choisir de générer et d'insérer un rapport d'activité.

Option par défaut: spécifiez ce qui suit si aucune option de rapport n'est requise:

MQRO_AUCUN
Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. MQRO_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

Informations générales :

1. Tous les types de rapport requis doivent être spécifiquement demandés par l'application qui envoie le message d'origine. Par exemple, si un rapport COA est demandé mais qu'un rapport d'exception ne l'est pas, un rapport COA est généré lorsque le message est placé dans la file d'attente de destination, mais aucun rapport d'exception n'est généré si la file d'attente de destination est saturée lorsque le message y arrive. Si aucune option *Report* n'est définie, aucun message de rapport n'est généré par le gestionnaire de files d'attente ou l'agent MCA.

   Certaines options de rapport peuvent être spécifiées même si le gestionnaire de files d'attente local ne les reconnaît pas ; cela est utile lorsque l'option doit être traitée par le gestionnaire de files d'attente *de destination* . Pour plus d'informations, voir «Options de rapport et indicateurs de message», à la page 894.

   Si un message de rapport est demandé, le nom de la file d'attente à laquelle envoyer le rapport doit être spécifié dans la zone *ReplyToQ* . Lorsqu'un message de rapport est reçu, la nature du rapport peut être déterminée en examinant la zone *Feedback* dans le descripteur de message.
2. Si le gestionnaire de files d'attente ou l'agent MCA qui génère un message de rapport ne peut pas placer le message de rapport dans la file d'attente de réponses (par exemple, parce que la file d'attente de réponses ou la file d'attente de transmission est saturée), le message de rapport est placé dans la file d'attente de rebut. Si cette *également* échoue ou qu'il n'existe pas de file d'attente de rebut, l'action effectuée dépend du type de message de rapport:

   - Si le message de rapport est un rapport d'exception, le message qui a généré le rapport d'exception est laissé dans sa file d'attente de transmission, ce qui garantit que le message n'est pas perdu.
   - Pour tous les autres types de rapport, le message de rapport est supprimé et le traitement se poursuit normalement. Cela est dû au fait que le message d'origine a déjà été distribué en toute sécurité (pour les messages de rapport COA ou COD) ou qu'il n'a plus d'intérêt (pour un message de rapport d'expiration).

   Une fois qu'un message de rapport a été correctement placé dans une file d'attente (file de destination ou file de transmission intermédiaire), le message n'est plus soumis à un traitement spécial ; il est traité comme tout autre message.
3. Lorsque le rapport est généré, la file d'attente *ReplyToQ* est ouverte et le message de rapport est inséré à l'aide des droits de l'utilisateur *UserIdentifier* dans le MQMD du message à l'origine du rapport, sauf dans les cas suivants:

- Les rapports d'exception générés par un agent MCA récepteur sont placés avec les droits utilisés par l'agent MCA lorsqu'il a tenté d'insérer le message à l'origine du rapport.
- Les rapports COA générés par le gestionnaire de files d'attente sont insérés avec les droits d'accès utilisés lorsque le message à l'origine du rapport a été inséré dans le gestionnaire de files d'attente qui génère le rapport. Par exemple, si le message a été inséré par un agent MCA récepteur à l'aide de l'identificateur utilisateur de l'agent MCA, le gestionnaire de files d'attente insère le rapport COA à l'aide de l'identificateur utilisateur de l'agent MCA.

Les applications qui génèrent des rapports doivent utiliser les mêmes droits qu'elles utilisent pour générer une réponse ; il s'agit généralement des droits de l'ID utilisateur dans le message d'origine.

Si le rapport doit être acheminé vers une destination éloignée, les expéditeurs et les destinataires peuvent décider de l'accepter ou non, de la même manière qu'ils le font pour les autres messages.

4. Si un message de rapport contenant des données est demandé:
   - Le message de rapport est toujours généré avec la quantité de données demandée par l'expéditeur du message d'origine. Si le message d'état est trop volumineux pour la file d'attente de réponses, le traitement décrit ci-dessus se produit ; le message d'état n'est jamais tronqué pour tenir dans la file d'attente de réponses.
   - Si le *Format* du message d'origine est MQFMT_XMIT_Q_HEADER, les données incluses dans le rapport n'incluent pas MQXQH. Les données de rapport commencent par le premier octet des données au-delà du MQXQH dans le message d'origine. Cela se produit que la file d'attente soit ou non une file d'attente de transmission.
5. Si un message COA, COD ou de rapport d'expiration est reçu dans la file d'attente de réponses, il est garanti que le message d'origine est arrivé, a été distribué ou a expiré, selon le cas. Toutefois, si un ou plusieurs de ces messages de rapport sont demandés et *ne sont pas* reçus, l'inverse ne peut pas être pris en compte car l'un des cas suivants peut s'être produit:
   a. Le message de rapport est suspendu car un lien est arrêté.
   b. Le message de rapport est suspendu car une condition de blocage existe dans une file d'attente de transmission intermédiaire ou dans la file d'attente de réponse (par exemple, la file d'attente est saturée ou bloquée pour les insertions).
   c. Le message de rapport se trouve dans une file d'attente de rebut.
   d. Lorsque le gestionnaire de files d'attente a tenté de générer le message de rapport, il n'a pu le placer ni dans la file d'attente appropriée, ni dans la file d'attente de rebut, de sorte que le message de rapport n'a pas pu être généré.
   e. Une défaillance du gestionnaire de files d'attente s'est produite entre l'action signalée (arrivée, distribution ou expiration) et la génération du message de rapport correspondant. (Cela ne se produit pas pour les messages de rapport COD si l'application extrait le message d'origine dans une unité de travail, car le message de rapport COD est généré dans la même unité de travail.)

   Les messages de rapport d'exception peuvent être conservés de la même manière pour les raisons 1, 2 et 3 ci-dessus. Toutefois, lorsqu'un agent MCA ne peut pas générer de message de rapport d'exception (le message de rapport ne peut pas être placé dans la file d'attente de réponses ou dans la file d'attente de rebut), le message d'origine reste dans la file d'attente de transmission de l'émetteur et le canal est fermé. Cela se produit indépendamment du fait que le message de rapport devait être généré à l'extrémité émettrice ou réceptrice du canal.
6. Si le message d'origine est temporairement bloqué (ce qui entraîne la génération d'un message de rapport d'exception et l'insertion du message d'origine dans une file d'attente de rebut), mais que le blocage est effacé et qu'une application lit ensuite le message d'origine dans la file d'attente de rebut et le place à nouveau à sa destination, les situations suivantes peuvent se produire:
   - Même si un message de rapport d'exception a été généré, le message d'origine arrive finalement à sa destination.
   - Plusieurs messages de rapport d'exception sont générés pour un seul message d'origine, car le message d'origine risque de rencontrer un autre blocage ultérieurement.

Signaler les messages lors de l'insertion dans une rubrique:

1. Des rapports peuvent être générés lors de l'insertion d'un message dans une rubrique. Ce message sera envoyé à tous les abonnés à la rubrique, qui peut être zéro, un ou plusieurs. Cela doit être pris en compte lorsque vous choisissez d'utiliser les options de rapport car un grand nombre de messages de rapport peuvent être générés en conséquence.
2. Lors de l'insertion d'un message dans une rubrique, il se peut que de nombreuses files d'attente de destination reçoivent une copie du message. Si certaines de ces files d'attente de destination présentent un problème, par exemple une file d'attente saturée, la réussite de l'opération MQPUT dépend de la valeur de NPMSGDLV ou de PMSGDLV (en fonction de la persistance du message). Si le paramètre est tel que la distribution des messages à la file d'attente de destination doit aboutir (par exemple, s'il s'agit d'un message persistant envoyé à un abonné durable et que PMSGDLV est défini sur ALL ou ALLDUR), la réussite est définie comme l'un des critères suivants:
   - Insertion réussie dans la file d'attente de l'abonné
   - Utilisation de MQRO_DEAD_LETTER_Q et insertion réussie dans la file d'attente de rebut si la file d'attente de l'abonné ne peut pas prendre le message
   - Utilisation de MQRO_DISCARD_MSG si la file d'attente de l'abonné ne peut pas prendre le message.

Messages de rapport pour les segments de message:

1. Des messages de rapport peuvent être demandés pour les messages dont la segmentation est autorisée (voir la description de l'indicateur MQMF_SEGMENTATION_ALLOWED). Si le gestionnaire de files d'attente juge nécessaire de segmenter le message, un message de rapport peut être généré pour chacun des segments qui rencontre ensuite la condition appropriée. Les applications doivent être préparées pour recevoir plusieurs messages de rapport pour chaque type de message de rapport demandé. Utilisez la zone *GroupId* dans le message de rapport pour corréler les différents rapports avec l'identificateur de groupe du message d'origine, et la zone *Feedback* identifie le type de chaque message de rapport.
2. Si MQGMO_LOGICAL_ORDER est utilisé pour extraire des messages de rapport pour des segments, sachez que des rapports de *différents types* peuvent être renvoyés par les appels MQGET successifs. Par exemple, si les rapports COA et COD sont demandés pour un message segmenté par le gestionnaire de files d'attente, les appels MQGET des messages de rapport peuvent renvoyer les messages de rapport COA et COD imbriqués de manière imprévisible. Evitez cela en utilisant l'option MQGMO_COMPLETE_MSG (éventuellement avec MQGMO_ACCEPT_TRUNCATED_MSG). MQGMO_COMPLETE_MSG entraîne le gestionnaire de files d'attente à réassembler les messages de rapport ayant le même type de rapport. Par exemple, le premier appel MQGET peut réassembler tous les messages COA relatifs au message d'origine et le deuxième appel MQGET peut réassembler tous les messages COD. Ce qui est réassemblé en premier dépend du type de message de rapport qui apparaît en premier dans la file d'attente.
3. Les applications qui elles-mêmes placent des segments peuvent spécifier des options de rapport différentes pour chaque segment. Notez toutefois les points suivants:
   - Si les segments sont extraits à l'aide de l'option MQGMO_COMPLETE_MSG, seules les options de rapport du *premier* segment sont prises en compte par le gestionnaire de files d'attente.
   - Si les segments sont extraits un par un et que la plupart d'entre eux possèdent l'une des options MQRO_COD_ *, mais pas au moins un segment, vous ne pouvez pas utiliser l'option MQGMO_COMPLETE_MSG pour extraire les messages de rapport avec un seul appel MQGET ou utiliser l'option MQGMO_ALL_SEGMENTS_AVAILABLE pour détecter l'arrivée de tous les messages de rapport.
4. Dans un réseau MQ , les gestionnaires de files d'attente peuvent avoir des fonctions différentes. Si un message de rapport pour un segment est généré par un gestionnaire de files d'attente ou un agent MCA qui ne prend pas en charge la segmentation, le gestionnaire de files d'attente ou l'agent MCA n'inclut pas par défaut les informations de segment nécessaires dans le message de rapport, ce qui peut rendre difficile l'identification du message d'origine à l'origine de la génération du rapport. Evitez cette difficulté en demandant des données avec le message de rapport, c'est-à-dire en spécifiant les options MQRO_ * _WITH_DATA ou MQRO_ * _WITH_FULL_DATA appropriées. Toutefois, sachez que si MQRO_ * _WITH_DATA est spécifié, *moins de* 100 octets de données de message d'application peuvent être renvoyés à l'application qui extrait le message de rapport, si le message de rapport

est généré par un gestionnaire de files d'attente ou un agent MCA qui ne prend pas en charge la segmentation.

Contenu du descripteur de message d'un message de rapport: Lorsque le gestionnaire de files d'attente ou l'agent MCA génère un message de rapport, il définit les zones du descripteur de message sur les valeurs suivantes, puis place le message de manière normale.

Zone dans MQMD	Valeur utilisée
StrucId	ID_STRUCD_MQM
Version	MQMD_VERSION_2
Report	MQRO_AUCUN
MsgType	MQMT_REPORT
Expiry	MQEI_UNLIMITED
Feedback	Selon la nature du rapport (MQFB_COA, MQFB_COD, MQFB_EXPIRATION ou une valeur MQRC_ *)
Encoding	Copié à partir du descripteur de message d'origine
CodedCharSetId	Copié à partir du descripteur de message d'origine
Format	Copié à partir du descripteur de message d'origine
Priority	Copié à partir du descripteur de message d'origine
Persistence	Copié à partir du descripteur de message d'origine
MsgId	Comme indiqué par les options de rapport dans le descripteur de message d'origine
CorrelId	Comme indiqué par les options de rapport dans le descripteur de message d'origine
BackoutCount	0
ReplyToQ	Espaces vides
ReplyToQMgr	Nom du gestionnaire de files d'attente
UserIdentifier	Comme défini par l'option MQPMO_PASS_IDENTITY_CONTEXT
AccountingToken	Comme défini par l'option MQPMO_PASS_IDENTITY_CONTEXT
ApplIdentityData	Comme défini par l'option MQPMO_PASS_IDENTITY_CONTEXT
PutApplType	MQAT_QMGR, ou selon le cas, pour l'agent MCA
PutApplName	28 premiers octets du nom du gestionnaire de files d'attente ou du nom de l'agent de canal de transmission de messages. Pour les messages de rapport générés par la passerelle IMS , cette zone contient le nom de groupe XCF et le nom de membre XCF du système IMS auquel le message est lié.
PutDate	Date d'envoi du message de rapport
PutTime	Heure d'envoi du message de rapport
ApplOriginData	Espaces vides
GroupId	Copié à partir du descripteur de message d'origine
MsgSeqNumber	Copié à partir du descripteur de message d'origine
Offset	Copié à partir du descripteur de message d'origine
MsgFlags	Copié à partir du descripteur de message d'origine

Zone dans MQMD	Valeur utilisée
OriginalLength	Copié à partir du descripteur de message d'origine s'il n'est pas défini par MQOL_UNDEFINED, et défini sur la longueur des données de message d'origine dans le cas contraire

Une application générant un rapport est recommandée pour définir des valeurs similaires, à l'exception des suivantes:

- La zone *ReplyToQMgr* peut être mise à blanc (le gestionnaire de files d'attente le remplace par le nom du gestionnaire de files d'attente local lors de l'insertion du message).
- Définissez les zones de contexte à l'aide de l'option qui aurait été utilisée pour une réponse, normalement MQPMO_PASS_IDENTITY_CONTEXT.

Analyse de la zone de rapport: la zone *Report* contient des sous-zones ; pour cette raison, les applications qui doivent vérifier si l'expéditeur du message a demandé un rapport particulier doivent utiliser l'une des techniques décrites dans «Analyse de la zone de rapport», à la page 895.

Il s'agit d'une zone de sortie pour l'appel MQGET et d'une zone d'entrée pour les appels MQPUT et MQPUT1 . La valeur initiale de cette zone est MQRO_NONE.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure, qui doit être:

ID_STRUCD_MQM
Identificateur de la structure de descripteur de message.

Pour le langage de programmation C, la constante MQMD_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQMD_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQMD_STRUC_ID.

UserIdentifier (MQCHAR12)

Il fait partie du **contexte d'identité** du message. Pour plus d'informations sur le contexte de message, voir «Présentation de MQMD», à la page 395 et Contexte de message .

UserIdentifier indique l'ID utilisateur de l'application à l'origine du message. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format.

Une fois qu'un message a été reçu, utilisez *UserIdentifier* dans la zone *AlternateUserId* du paramètre *ObjDesc* d'un appel MQOPEN ou MQPUT1 ultérieur pour effectuer la vérification d'autorisation de l'utilisateur *UserIdentifier* au lieu de l'application effectuant l'ouverture.

Lorsque le gestionnaire de files d'attente génère ces informations pour un appel MQPUT ou MQPUT1 :

- Sous z/OS, le gestionnaire de files d'attente utilise le paramètre *AlternateUserId* du paramètre *ObjDesc* de l'appel MQOPEN ou MQPUT1 si l'option MQOO_ALTERNATE_USER_AUTHORITY ou MQPMO_ALTERNATE_USER_AUTHORITY a été spécifiée. Si l'option appropriée n'a pas été spécifiée, le gestionnaire de files d'attente utilise un identificateur utilisateur déterminé à partir de l'environnement.
- Dans les autres environnements, le gestionnaire de files d'attente utilise toujours un identificateur utilisateur déterminé à partir de l'environnement.

Lorsque l'ID utilisateur est déterminé à partir de l'environnement:

- Sous z/OS, le gestionnaire de files d'attente utilise:
  - Pour MVS (par lots), ID utilisateur de la carte de travail JES ou de la tâche démarrée
  - Pour TSO, l'ID utilisateur est propagé au travail lors de la soumission du travail
  - Pour CICS, l'ID utilisateur associé à la tâche
  - Pour IMS, l'ID utilisateur dépend du type d'application:

- Pour :
  - Régions BMP sans message
  - Régions IFP sans message
  - Les régions BMP de message et IFP de message qui n'ont *pas* émis d'appel GU réussi

  le gestionnaire de files d'attente utilise l'ID utilisateur de la carte JOB JES de la région ou l'ID utilisateur TSO. S'ils sont vides ou nuls, ils utilisent le nom du bloc de spécification de programme (PSB).
- Pour :
  - Les régions BMP de message et IFP de message qui *ont* émis un appel GU réussi
  - Régions MPP

  le gestionnaire de files d'attente utilise l'un des éléments suivants:
  - ID utilisateur connecté associé au message
  - Nom du terminal logique (LTERM)
  - Identificateur utilisateur de la carte de travail JES de la région
  - ID utilisateur TSO
  - Nom du bloc de spécification de programme

- Sous IBM i, le gestionnaire de files d'attente utilise le nom du profil utilisateur associé au travail d'application.
- Sur les systèmes UNIX , le gestionnaire de files d'attente utilise:
  - Nom de connexion de l'application
  - ID utilisateur effectif du processus si aucune connexion n'est disponible
  - Identificateur utilisateur associé à la transaction, si l'application est une transaction CICS
- Sur les systèmes Windows , le gestionnaire de files d'attente utilise les 12 premiers caractères du nom d'utilisateur connecté.

Cette zone est normalement une zone de sortie générée par le gestionnaire de files d'attente, mais pour un appel MQPUT ou MQPUT1 , vous pouvez définir cette zone comme zone d'entrée-sortie et spécifier la zone UserIdentification au lieu de laisser le gestionnaire de files d'attente générer ces informations. Indiquez MQPMO_SET_IDENTITY_CONTEXT ou MQPMO_SET_ALL_CONTEXT dans le paramètre PutMsgOpts et indiquez un ID utilisateur dans la zone UserIdentifier si vous ne souhaitez pas que le gestionnaire de files d'attente génère la zone UserIdentifier pour un appel MQPUT ou MQPUT1 .

Pour les appels MQPUT et MQPUT1 , il s'agit d'une zone d'entrée-sortie si MQPMO_SET_IDENTITY_CONTEXT ou MQPMO_SET_ALL_CONTEXT est spécifié dans le paramètre *PutMsgOpts* . Toutes les informations qui suivent un caractère null dans la zone sont supprimées. Le gestionnaire de files d'attente convertit le caractère null et les caractères suivants en blancs. Si MQPMO_SET_IDENTITY_CONTEXT ou MQPMO_SET_ALL_CONTEXT n'est pas spécifié, cette zone est ignorée en entrée et est une zone de sortie uniquement.

Une fois qu'un appel MQPUT ou MQPUT1 a abouti, cette zone contient le *UserIdentifier* qui a été transmis avec le message s'il a été inséré dans une file d'attente. Il s'agit de la valeur de *UserIdentifier* qui est conservée avec le message (voir la description de MQPMO_RETAIN pour plus de détails sur les publications conservées) mais qui n'est pas utilisée en tant que *UserIdentifier* lorsque le message est envoyé en tant que publication aux abonnés car ils fournissent une valeur pour remplacer *UserIdentifier* dans toutes les publications qui leur sont envoyées. Si le message n'a pas de contexte, la zone est entièrement vide.

Il s'agit d'une zone de sortie pour l'appel MQGET. La longueur de cette zone est indiquée par MQ_USER_ID_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 12 caractères blancs dans les autres langages de programmation.

Version (MQLONG)

Il s'agit du numéro de version de la structure, qui doit être l'un des suivants:

MQMD_VERSION_1
Structure de descripteur de message Version-1 .

Cette version est prise en charge dans tous les environnements.

MQMD_VERSION_2
Structure de descripteur de message Version-2 .

Cette version est prise en charge dans tous les environnements WebSphere MQ V6.0 et versions ultérieures, ainsi que dans les clients WebSphere MQ MQI connectés à ces systèmes.

Remarque : Lorsqu'un MQMD version-2 est utilisé, le gestionnaire de files d'attente effectue des vérifications supplémentaires sur les structures d'en-tête MQ qui peuvent être présentes au début des données de message d'application. Pour plus de détails, voir les remarques sur l'utilisation de l'appel MQPUT.

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

MQMD_CURRENT_VERSION
Version actuelle de la structure de descripteur de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQMD_VERSION_1.

Valeurs initiales et déclarations de langue pour MQMD

Tableau 515. Valeurs initiales des zones dans MQMD pour MQMD

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUCD_MQM	'MD'
Version	MQMD_VERSION_1	1
Report	MQRO_AUCUN	0
MsgType	MQMT_DATAGRAM	8
Expiry	MQEI_UNLIMITED	-1
Feedback	MQFB_AUCUN	0
Encoding	MQENC_NATIVE	Dépend de l'environnement
CodedCharSetId	MQCCSI_Q_DIR	0
Format	MQFMT_AUCUN	Espaces vides
Priority	MQPRI_PRIORITY_AS_Q_DEF	-1
Persistence	MQPER_PERSISTENCE_AS_Q_DEF	2
MsgId	MQMI_AUCUN	Valeurs NULL
CorrelId	MQCI_NONE	Valeurs NULL
BackoutCount	Aucun	0
ReplyToQ	Aucun	Chaîne nulle ou blancs
ReplyToQMgr	Aucun	Chaîne nulle ou blancs
UserIdentifier	Aucun	Chaîne nulle ou blancs
AccountingToken	MQACT_AUCUN	Valeurs NULL
ApplIdentityData	Aucun	Chaîne nulle ou blancs

Tableau 515. Valeurs initiales des zones dans MQMD pour MQMD (suite)

Nom de zone	Nom de la constante	Valeur de la constante
PutApplType	MQAT_NO_CONTEXT	0
PutApplName	Aucun	Chaîne nulle ou blancs
PutDate	Aucun	Chaîne nulle ou blancs
PutTime	Aucun	Chaîne nulle ou blancs
ApplOriginData	Aucun	Chaîne nulle ou blancs
GroupId	MQGI_AUCUN	Valeurs NULL
MsgSeqNumber	Aucun	1
Offset	Aucun	0
MsgFlags	MQMF_AUCUN	0
OriginalLength	MQOL_UNDEFINI	-1

Remarques :

1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
2. Dans le langage de programmation C, la variable macroMQMD_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQMD MyMD = {MQMD_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Extension de descripteur de message

Le tableau suivant récapitule les zones de la structure.

Tableau 516. Zones dans MQMDE

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
StrucLength	Longueur de la structure MQMDE	StrucLength
Encoding	Codage numérique des données qui suit MQMDE	Codage
CodedCharSetId	Identificateur de jeu de caractères des données qui suit MQMDE	CodedCharSetId
Format	Nom du format des données qui suit MQMDE	Format
Flags	Indicateurs généraux	Indicateurs
GroupId	Identificateur de groupe	GroupId
MsgSeqNumber	Numéro de séquence du message logique dans le groupe	MsgSeqNumber
Offset	Décalage des données du message physique à partir du début du message logique	offset
MsgFlags	Indicateurs de message	MsgFlags
OriginalLength	Longueur du message d'origine	OriginalLength

Présentation de MQMDE

Disponibilité: Tous les systèmes WebSphere MQ , plus les clients WebSphere MQ connectés à ces systèmes.

Objectif: La structure MQMDE décrit les données qui précèdent parfois les données de message d'application. La structure contient les zones MQMD qui existent dans le MQMD version-2 , mais pas dans le MQMD version-1 .

Nom de format: MQFMT_MD_EXTENSION.

Jeu de caractères et codage: les données dans MQMDE doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local ; ces données sont fournies par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et MQENC_NATIVE pour le langage de programmation C.

Définissez le jeu de caractères et le codage de MQMDE dans les zones *CodedCharSetId* et *Encoding* dans:

- MQMD (si la structure MQMDE est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQMDE (tous les autres cas).

Si le MQMDE ne fait pas partie du jeu de caractères et du codage du gestionnaire de files d'attente, il est accepté mais non respecté, c'est-à-dire que le MQMDE est traité comme des données de message.

Remarque : Sous Windows, les applications compilées avec Micro Focus COBOL utilisent une valeur de MQENC_NATIVE différente du codage du gestionnaire de files d'attente. Bien que les zones numériques de la structure MQMD sur les appels MQPUT, MQPUT1et MQGET doivent être dans le codage Micro Focus COBOL, les zones numériques de la structure MQMDE doivent être dans le codage du gestionnaire de files d'attente. Ce dernier est donné par MQENC_NATIVE pour le langage de programmation C, et a la valeur 546.

Utilisation: Les applications qui utilisent un MQMD version-2 ne détecteront pas de structure MQMDE. Toutefois, des applications spécialisées et des applications qui continuent d'utiliser un MQMD version-1 peuvent rencontrer un MQMDE dans certaines situations. La structure MQMDE peut se produire dans les cas suivants:

- Spécifié sur les appels MQPUT et MQPUT1
- Renvoyé par l'appel MQGET
- Dans les messages des files d'attente de transmission

MQMDE spécifié sur les appels MQPUT et MQPUT1: sur les appels MQPUT et MQPUT1 , si l'application fournit un MQMD version-1 , l'application peut éventuellement préfixer les données de message avec un MQMDE, en définissant la zone *Format* dans MQMD sur MQFMT_MD_EXTENSION pour indiquer qu'un MQMDE est présent. Si l'application ne fournit pas de MQMDE, le gestionnaire de files d'attente utilise les valeurs par défaut pour les zones du MQMDE. Les valeurs par défaut utilisées par le gestionnaire de files d'attente sont identiques aux valeurs initiales de la structure ; voir Tableau 518, à la page 451.

Si l'application fournit un version-2 MQMD *et* préfixe les données de message d'application avec un MQMDE, les structures sont traitées comme indiqué dans la Tableau 517, à la page 448.

Tableau 517. Action du gestionnaire de files d'attente lorsque MQMDE est spécifié sur MQPUT ou MQPUT1 pour MQMDE

Version MQMD	Valeurs des zones version-2	Valeurs des zones correspondantes dans MQMDE	Action effectuée par le gestionnaire de files d'attente
1	-	Valide	MQMDE est honoré
2	Par défaut	Valide	MQMDE est honoré
2	Pas de valeur par défaut	Valide	MQMDE est traité comme des données de message

Tableau 517. Action du gestionnaire de files d'attente lorsque MQMDE est spécifié sur MQPUT ou MQPUT1 pour MQMDE (suite)

Version MQMD	Valeurs des zones version-2	Valeurs des zones correspondantes dans MQMDE	Action effectuée par le gestionnaire de files d'attente
1 ou 2	Tous	Non valide	L'appel échoue avec un code anomalie approprié
1 ou 2	Tous	MQMDE est dans un jeu de caractères ou un codage incorrect, ou est une version non prise en charge	MQMDE est traité comme des données de message
Remarque : Sous z/OS, si l'application spécifie un MQMD version-1 avec un MQMDE, le gestionnaire de files d'attente valide le MQMDE uniquement si la file d'attente a un *IndexType* de MQIT_GROUP_ID.			

Il y a un cas particulier. Si l'application utilise un MQMD version-2 pour insérer un message qui est un segment (c'est-à-dire que l'indicateur MQMF_SEGMENT ou MQMF_LAST_SEGMENT est défini) et que le nom de format dans le MQMD est MQFMT_DEAD_LETTER_HEADER, le gestionnaire de files d'attente génère une structure MQMDE et l'insère *entre* la structure MQDLH et les données qui la suivent. Dans le MQMD conservé par le gestionnaire de files d'attente avec le message, les zones version-2 sont définies sur leurs valeurs par défaut.

Plusieurs des zones qui existent dans le MQMD version-2 mais pas dans le MQMD version-1 sont des zones d'entrée-sortie dans MQPUT et MQPUT1. Toutefois, le gestionnaire de files d'attente ne renvoie *aucune* valeur dans les zones équivalentes de MQMDE dans la sortie des appels MQPUT et MQPUT1 ; si l'application requiert ces valeurs de sortie, elle doit utiliser un MQMD version-2 .

MQMDE renvoyé par l'appel MQGET: dans l'appel MQGET, si l'application fournit un MQMD version-1 , le gestionnaire de files d'attente préfixe le message renvoyé avec un MQMDE, mais uniquement si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut. Le gestionnaire de files d'attente définit la zone *Format* dans MQMD sur la valeur MQFMT_MD_EXTENSION pour indiquer qu'un MQMDE est présent.

Si l'application fournit un MQMDE au début du paramètre *Buffer* , MQMDE est ignoré. En retour de l'appel MQGET, il est remplacé par le MQMDE pour le message (si nécessaire) ou remplacé par les données de message d'application (si le MQMDE n'est pas nécessaire).

Si l'appel MQGET renvoie un MQMDE, les données du MQMDE se trouvent généralement dans le jeu de caractères et le codage du gestionnaire de files d'attente. Toutefois, le MQMDE peut être dans un autre jeu de caractères et codage si:

- Le MQMDE a été traité comme des données sur l'appel MQPUT ou MQPUT1 (voir Tableau 517, à la page 448 pour connaître les circonstances qui peuvent être à l'origine de cette erreur).
- Le message a été reçu d'un gestionnaire de files d'attente éloignées connecté par une connexion TCP et l'agent MCA (Message Channel Agent) récepteur n'a pas été configuré correctement.

Remarque : Sous Windows, les applications compilées avec Micro Focus COBOL utilisent une valeur de MQENC_NATIVE différente du codage du gestionnaire de files d'attente (voir ci-dessus).

MQMDE dans les messages dans les files d'attente de transmission: les messages dans les files d'attente de transmission sont précédés de la structure MQXQH, qui contient un MQMD version-1 . Un MQMDE peut également être présent, positionné entre la structure MQXQH et les données de message d'application, mais il n'est généralement présent que si une ou plusieurs des zones du MQMDE ont une valeur autre que la valeur par défaut.

D'autres structures d'en-tête MQ peuvent également se produire entre la structure MQXQH et les données de message d'application. Par exemple, lorsque l'en-tête de rebut MQDLH est présent et que le message n'est pas un segment, l'ordre est le suivant:

- MQXQH (contenant un MQMD version-1)
- MQMDE
- MQDLH
- données de message d'application

Zones pour MQMDE

La structure MQMDE contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

CodedCharSetId (MQLONG)

Indique l'identificateur de jeu de caractères des données qui suivent la structure MQMDE ; il ne s'applique pas aux données de type caractères de la structure MQMDE elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que cette zone est valide. La valeur spéciale suivante peut être utilisée:

MQCCSI_HÉRITER
Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI_INHERIT n'est pas renvoyée par l'appel MQGET.

MQCCSI_INHERIT ne peut pas être utilisé si la valeur de la zone *PutApplType* dans MQMD est MQAT_BROKER.

Cette valeur est prise en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus les clients WebSphere MQ connectés à ces systèmes.

La valeur initiale de cette zone est MQCCSI_UNDEFINED.

Codage (MQLONG)

Indique le codage numérique des données qui suivent la structure MQMDE ; il ne s'applique pas aux données numériques de la structure MQMDE elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que la zone est valide. Pour plus d'informations sur les codages de données, voir la zone *Encoding* décrite dans «MQMD-Descripteur de message», à la page 394 .

La valeur initiale de cette zone est MQENC_NATIVE.

Indicateurs (MQLONG)

L'indicateur suivant peut être spécifié:

MQMDEF_AUCUN
Aucun indicateur.

La valeur initiale de cette zone est MQMDEF_NONE.

Format (MQCHAR8)

Indique le nom de format des données qui suivent la structure MQMDE.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Le gestionnaire de files d'attente ne vérifie pas que cette zone est valide. Pour plus d'informations sur les noms de format, voir la zone *Format* décrite dans «MQMD-Descripteur de message», à la page 394 .

La valeur initiale de cette zone est MQFMT_NONE.

GroupId (MQBYTE24)

Voir la zone *GroupId* décrite dans «MQMD-Descripteur de message», à la page 394. La valeur initiale de cette zone est MQGI_NONE.

MsgFlags (MQLONG)

Voir la zone *MsgFlags* décrite dans «MQMD-Descripteur de message», à la page 394. La valeur initiale de cette zone est MQMF_NONE.

MsgSeqNuméro (MQLONG)

Voir la zone *MsgSeqNumber* décrite dans «MQMD-Descripteur de message», à la page 394. La valeur initiale de cette zone est 1.

Décalage (MQLONG)

Voir la zone *Offset* décrite dans «MQMD-Descripteur de message», à la page 394. La valeur initiale de cette zone est 0.

OriginalLength (MQLONG)

Voir la zone *OriginalLength* décrite dans «MQMD-Descripteur de message», à la page 394. La valeur initiale de cette zone est MQOL_UNDEFINED.

StrucId (MQCHAR4)

La valeur doit être:

ID_STRUCDE_MQM
Identificateur de la structure d'extension du descripteur de message.

Pour le langage de programmation C, la constante MQMDE_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQMDE_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQMDE_STRUC_ID.

StrucLength (MQLONG)

Longueur de la structure MQMDE ; la valeur suivante est définie:

MQMDE_LENGTH_2
Longueur de la structure d'extension du descripteur de message version-2 .

La valeur initiale de cette zone est MQMDE_LENGTH_2.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQMDE_VERSION_2
Structure d'extension du descripteur de message Version-2 .

La constante suivante indique le numéro de version de la version en cours:

MQMDE_CURRENT_VERSION
Version actuelle de la structure d'extension du descripteur de message.

La valeur initiale de cette zone est MQMDE_VERSION_2.

Valeurs initiales et déclarations de langue pour MQMDE

Tableau 518. Valeurs initiales des zones dans MQMDE pour MQMDE

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUCDE_MQM	'MDE¬'

Tableau 518. Valeurs initiales des zones dans MQMDE pour MQMDE (suite)

Nom de zone	Nom de la constante	Valeur de la constante
Version	MQMDE_VERSION_2	2
StrucLength	MQMDE_LENGTH_2	72
Encoding	MQENC_NATIVE	Dépend de l'environnement
CodedCharSetId	MQCCSI_UNDEFINI	0
Format	MQFMT_AUCUN	Espaces vides
Flags	MQMDEF_AUCUN	0
GroupId	MQGI_AUCUN	Valeurs NULL
MsgSeqNumber	Aucun	1
Offset	Aucun	0
MsgFlags	MQMF_AUCUN	0
OriginalLength	MQOL_UNDEFINI	-1

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQMDE_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQMDE MyMDE = {MQMDE_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Options de traitement des messages vers la mémoire tampon

Le tableau suivant récapitule les zones de la structure. Structure MQMHBO-descripteur de message vers options de mémoire tampon

Tableau 519. Zones dans MQMHBO		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options contrôlant l'action de MQMHBUF	Options

Présentation de MQMHBO

Disponibilité: Tous les systèmes WebSphere MQ et WebSphere MQ clients MQI.

Objectif: La structure MQMHBO permet aux applications de spécifier des options qui contrôlent la façon dont les mémoires tampon sont produites à partir des descripteurs de message. La structure est un paramètre d'entrée dans l'appel MQMHBUF.

Jeu de caractères et codage: les données de MQMHBO doivent figurer dans le jeu de caractères de l'application et dans le codage de l'application (MQENC_NATIVE).

Zones pour MQMHBO

Descripteur de message vers la structure des options de mémoire tampon-zones

La structure MQMHBO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

Options (MQLONG)
Descripteur de message pour la structure des options de mémoire tampon-Zone Options

Ces options contrôlent l'action de MQMHBUF.

Vous devez spécifier l'option suivante:

MQMHBO_PROPERTIES_IN_MQRFH2
Lors de la conversion des propriétés d'un descripteur de message en mémoire tampon, convertissez-les au format MQRFH2 .

Si vous le souhaitez, vous pouvez également spécifier la valeur suivante. Si les valeurs requises peuvent être:

- Ajouté ensemble (ne pas ajouter la même constante plus d'une fois), ou
- combinées par le biais de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations par bit).

MQMHBO_DELETE_PROPERTIES
Les propriétés ajoutées à la mémoire tampon sont supprimées du descripteur de message. Si l'appel échoue, aucune propriété n'est supprimée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQMHBO_PROPERTIES_IN_MQRFH2.

StrucId (MQCHAR4)
Descripteur de message vers la structure d'options de mémoire tampon-Zone StrucId

Il s'agit de l'identificateur de structure. La valeur doit être:

MQMHBO_STRUC_ID
Identificateur de la structure des options de traitement des messages vers la mémoire tampon.

Pour le langage de programmation C, la constante MQMHBO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQMHBO_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQMHBO_STRUC_ID.

Version (MQLONG)
Descripteur de message pour la structure des options de mémoire tampon-Zone Version

Il s'agit du numéro de version de la structure. La valeur doit être:

MQMHBO_VERSION_1
Numéro de version de la structure des options de traitement des messages à la mémoire tampon.

La constante suivante indique le numéro de version de la version en cours:

MQMHBO_CURRENT_VERSION
Version actuelle de la structure des options du descripteur de message vers la mémoire tampon.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQMHBO_VERSION_1.

Valeurs initiales et déclarations de langue pour MQMHBO
Descripteur de message à la structure de la mémoire tampon-Valeurs initiales

Tableau 520. Valeurs initiales des zones dans MQMHBO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	MQMHBO_STRUC_ID	'MHBO'
Version	MQMHBO_VERSION_1	1
Options	MQMHBO_PROPERTIES_IN_MQRFH2	

Remarques :

1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
2. Dans le langage de programmation C, la variable macroMQMHBO_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQMHBO MyMHBO = {MQMHBO_DEFAULT};
```

Déclaration C
Descripteur de message dans la structure des options de la mémoire tampon-Déclaration en langage C

```
typedef struct tagMQMHBO MQMHBO;
struct tagMQMHBO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
```

```
                                MQMHBUF */
};
```

Déclaration COBOL
Descripteur de message dans la structure des options de mémoire tampon-Déclaration en langage COBOL

```
**   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.
```

Déclaration PL/I
Descripteur de message dans la structure des options de mémoire tampon-Déclaration en langage PL/I

```
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 */
```

Déclaration High Level Assembler
Descripteur de message dans la structure des options de la mémoire tampon-Déclaration du langage assembleur

```
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-Descripteur d'objet

Le tableau suivant récapitule les zones de la structure.

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
ObjectType	Type d'objet	ObjectType
ObjectName	Nom d'objet	ObjectName
ObjectQMgrName	Nom du gestionnaire de files d'attente d'objets	ObjectQMgrName
DynamicQName	Nom de la file d'attente dynamique	DynamicQName
AlternateUserId	Identificateur d'utilisateur de remplacement	AlternateUserid
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQOD_VERSION_2.		
RecsPresent	Nombre d'enregistrements d'objet présents	RecsPresent
KnownDestCount	Nombre de files d'attente locales ouvertes correctement	KnownDestCount

Zone	Description	Topic
UnknownDestCount	Nombre de files d'attente éloignées ouvertes correctement	UnknownDestCount
InvalidDestCount	Nombre de files d'attente dont l'ouverture a échoué	InvalidDestCount
ObjectRecOffset	Décalage du premier enregistrement d'objet à partir du début de MQOD	DécalageObjectRec
ResponseRecOffset	Décalage du premier enregistrement de réponse à partir du début de MQOD	DécalageResponseRec
ObjectRecPtr	Adresse du premier enregistrement d'objet	ObjectRecPtr
ResponseRecPtr	Adresse du premier enregistrement de réponse	ResponseRecPtr
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQOD_VERSION_3.		
AlternateSecurityId	Identificateur de sécurité de remplacement	AlternateSecurityId
ResolvedQName	Nom de file d'attente résolu	ResolvedQName
ResolvedQMgrName	Nom de gestionnaire de files d'attente résolu	ResolvedQMgrNom
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQOD_VERSION_4.		
ObjectString	Nom d'objet long	ObjectString
SelectionString	Chaîne de sélection	SelectionString
ResObjectString	Nom d'objet long résolu	ResObjectChaîne
ResolvedType	Type d'objet résolu	ResolvedType

Présentation de MQOD

Disponibilité: Tous les systèmes WebSphere MQ , plus WebSphere MQ clients MQI connectés à ces systèmes.

Objectif: La structure MQOD est utilisée pour spécifier un objet par nom. Les types d'objet suivants sont valides:

- File d'attente ou liste de distribution
- Liste de noms
- Définition de processus
- Gestionnaire de files d'attente
- Topic

La structure est un paramètre d'entrée-sortie dans les appels MQOPEN et MQPUT1 .

Version: la version actuelle de MQOD est MQOD_VERSION_4. Les applications que vous souhaitez porter entre plusieurs environnements doivent s'assurer que la version requise de MQOD est prise en charge dans tous les environnements concernés. Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions qui suivent.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQOD prise en charge par l'environnement, mais avec la valeur initiale de la zone *Version* définie sur MQOD_VERSION_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone *Version* sur le numéro de version de la version requise.

Pour ouvrir une liste de distribution, *Version* doit avoir la valeur MQOD_VERSION_2 ou une valeur supérieure.

Jeu de caractères et codage: les données dans MQOD doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Zones pour MQOD

La structure MQOD contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

ID AlternateSecurity(MQBYTE40)

Il s'agit d'un identificateur de sécurité qui est transmis avec *AlternateUserId* au service d'autorisation pour permettre l'exécution des vérifications d'autorisation appropriées. *AlternateSecurityId* est utilisé uniquement si:

- MQOO_ALTERNATE_USER_AUTHORITY est spécifié dans l'appel MQOPEN, ou
- MQPMO_ALTERNATE_USER_AUTHORITY est spécifié dans l'appel MQPUT1 ,

et la zone *AlternateUserId* n'est pas entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone.

Sous Windows, *AlternateSecurityId* peut être utilisé pour fournir l'identificateur de sécurité (SID) Windows qui identifie de manière unique le *AlternateUserId*. Le SID d'un utilisateur peut être obtenu à partir du système Windows à l'aide de l'appel API `LookupAccountName()` Windows .

Sous z/OS, cette zone est ignorée.

La zone *AlternateSecurityId* possède la structure suivante:

- Le premier octet est un entier binaire contenant la longueur des données significatives qui suivent ; la valeur exclut l'octet de longueur lui-même. Si aucun identificateur de sécurité n'est présent, la longueur est égale à zéro.
- Le deuxième octet indique le type d'identificateur de sécurité présent ; les valeurs suivantes sont possibles:

  ID_SÉCURITÉ_NT_MQSIDT_ID
  Identificateur de sécurité Windows .

  MQSIDT_AUCUN
  Aucun identificateur de sécurité.
- Le troisième octet et les suivants jusqu'à la longueur définie par le premier octet contiennent l'identifiant de sécurité lui-même.
- Les octets restants dans la zone sont définis sur zéro binaire.

Vous pouvez utiliser la valeur spéciale suivante:

MQSID_AUCUN
Aucun identificateur de sécurité n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQSID_NONE_ARRAY est également définie ; elle a la même valeur que MQSID_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ_SECURITY_ID_LENGTH. La valeur initiale de cette zone est MQSID_NONE. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_3.

ID AlternateUser(MQCHAR12)

Si vous spécifiez MQOO_ALTERNATE_USER_AUTHORITY pour l'appel MQOPEN ou MQPMO_ALTERNATE_USER_AUTHORITY pour l'appel MQPUT1 , cette zone contient un autre

identificateur utilisateur qui est utilisé pour vérifier l'autorisation d'ouverture, à la place de l'identificateur utilisateur sous lequel l'application est en cours d'exécution. Cependant, certaines vérifications sont toujours effectuées avec l'identificateur d'utilisateur en cours (par exemple, des vérifications de contexte).

Si MQOO_ALTERNATE_USER_AUTHORITY ou MQPMO_ALTERNATE_USER_AUTHORITY est spécifié et que cette zone est entièrement vide jusqu'au premier caractère null ou à la fin de la zone, l'ouverture ne peut aboutir que si aucune autorisation utilisateur n'est requise pour ouvrir cet objet avec les options spécifiées.

Si ni MQOO_ALTERNATE_USER_AUTHORITY ni MQPMO_ALTERNATE_USER_AUTHORITY n'est spécifié, cette zone est ignorée.

Les différences suivantes existent dans les environnements indiqués:

- Sous z/OS, seuls les 8 premiers caractères de *AlternateUserId* sont utilisés pour vérifier l'autorisation d'ouverture. Toutefois, l'ID utilisateur en cours doit être autorisé à indiquer cet ID utilisateur de remplacement particulier ; les 12 caractères de l'ID utilisateur de remplacement sont utilisés pour cette vérification. L'ID utilisateur ne doit contenir que des caractères autorisés par le gestionnaire de sécurité externe.

  Si *AlternateUserId* est spécifié pour une file d'attente, la valeur peut être utilisée ultérieurement par le gestionnaire de files d'attente lors de l'insertion de messages. Si les options MQPMO_ * _CONTEXT spécifiées dans l'appel MQPUT ou MQPUT1 entraînent la génération par le gestionnaire de files d'attente des informations de contexte d'identité, le gestionnaire de files d'attente place *AlternateUserId* dans la zone *UserIdentifier* du MQMD du message, à la place de l'ID utilisateur en cours.
- Dans les autres environnements, *AlternateUserId* est utilisé uniquement pour les vérifications de contrôle d'accès sur l'objet ouvert. Si l'objet est une file d'attente, *AlternateUserId* n'affecte pas le contenu de la zone *UserIdentifier* dans le MQMD des messages envoyés à l'aide de cet identificateur de file d'attente.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ_USER_ID_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 12 caractères blancs dans les autres langages de programmation.

DynamicQName (MQCHAR48)

Il s'agit du nom d'une file d'attente dynamique qui doit être créée par l'appel MQOPEN. Cela n'est pertinent que lorsque *ObjectName* spécifie le nom d'une file d'attente modèle ; dans tous les autres cas, *DynamicQName* est ignoré.

Les caractères valides dans le nom sont identiques à ceux de *ObjectName*, sauf qu'un astérisque est également valide. Un nom à blanc (ou dans lequel seuls des blancs apparaissent avant le premier caractère nul) n'est pas valide si *ObjectName* est le nom d'une file d'attente modèle.

Si le dernier caractère non blanc du nom est un astérisque (*), le gestionnaire de files d'attente remplace l'astérisque par une chaîne de caractères qui garantit que le nom généré pour la file d'attente est unique au niveau du gestionnaire de files d'attente local. Pour autoriser un nombre suffisant de caractères, l'astérisque est admis uniquement aux positions 1 à 33. Il ne doit pas y avoir d'autres caractères que des blancs ou un caractère nul après l'astérisque.

L'astérisque peut apparaître à la première position de caractère, auquel cas le nom se compose uniquement des caractères générés par le gestionnaire de files d'attente.

Sous z/OS, n'utilisez pas d'astérisque en première position car aucun contrôle de sécurité ne peut être effectué sur une file d'attente dont le nom complet est généré automatiquement.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone est déterminée par l'environnement:

- Sous z/OS, la valeur est `'CSQ.*'`.
- Sur les autres plateformes, la valeur est `'AMQ.*'`.

La valeur est une chaîne à terminaison nulle en C et une chaîne à remplissage vide dans d'autres langages de programmation.

Nombre d' InvalidDest(MQLONG)

Il s'agit du nombre de files d'attente de la liste de distribution dont l'ouverture a échoué. Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

Remarque : Si cette zone est présente, elle est définie *uniquement* si le paramètre *CompCode* de l'appel MQOPEN ou MQPUT1 est MQCC_OK ou MQCC_WARNING; elle n'est *pas* définie si le paramètre *CompCode* est MQCC_FAILED.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_1.

Nombre de KnownDest(MQLONG)

Il s'agit du nombre de files d'attente de la liste de distribution qui ont été résolues en files d'attente locales et qui ont été ouvertes avec succès. Le nombre n'inclut pas les files d'attente qui se résolvent en files d'attente éloignées (même si une file d'attente de transmission locale est utilisée initialement pour stocker le message). Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_1.

ObjectName (MQCHAR48)

Il s'agit du nom local de l'objet tel qu'il est défini sur le gestionnaire de files d'attente identifié par *ObjectQMgrName*. Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (_), pourcentage (%)

Le nom ne doit pas contenir d'espaces de début ou d'espaces imbriqués, mais peut contenir des espaces de fin. Utilisez un caractère null pour indiquer la fin des données significatives dans le nom ; la valeur null et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent dans les environnements indiqués:

- Sur les systèmes qui utilisent EBCDIC Katakana, les caractères minuscules ne peuvent pas être utilisés.
- Sous z/OS:
  - Evitez les noms qui commencent ou se terminent par un trait de soulignement ; ils ne peuvent pas être traités par les panneaux d'opérations et de contrôle.
  - Le caractère de pourcentage a une signification spéciale pour RACF. Si RACF est utilisé comme gestionnaire de sécurité externe, les noms ne doivent pas contenir le pourcentage. Dans ce cas, ces noms ne sont pas inclus dans les contrôles de sécurité lorsque des profils génériques RACF sont utilisés.
- Sous IBM i, les noms contenant des caractères minuscules, des barres obliques ou des pourcentages doivent être placés entre guillemets lorsqu'ils sont spécifiés dans les commandes. Ces guillemets ne doivent pas être indiqués pour les noms qui apparaissent en tant que zones dans les structures ou en tant que paramètres dans les appels.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*. Pour plus de détails sur l'utilisation de ces deux zones, voir «Utilisation de chaînes de rubrique», à la page 562.

Les points suivants s'appliquent aux types d'objet indiqués:

- Si *ObjectName* est le nom d'une file d'attente modèle, le gestionnaire de files d'attente crée une file d'attente dynamique avec les attributs de la file d'attente modèle et renvoie dans la zone *ObjectName* le nom de la file d'attente créée. Une file d'attente modèle ne peut être indiquée que sur l'appel MQOPEN ; une file d'attente modèle n'est pas valide sur l'appel MQPUT1 .
- Si *ObjectName* est le nom d'une file d'attente alias avec TARGTYPE (TOPIC), un contrôle de sécurité est d'abord effectué sur la file d'attente alias nommée ; cela est normal lorsque des files d'attente alias sont utilisées. Lorsque le contrôle de sécurité aboutit, l'appel MQOPEN se poursuit et se comporte comme un appel MQOPEN sur un MQOT_TOPIC; cela inclut un contrôle de sécurité par rapport à l'objet de rubrique d'administration.
- Si *ObjectName* et *ObjectQMgrName* identifient une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, il ne doit pas y avoir de définition de file d'attente du même nom sur le gestionnaire de files d'attente local. S'il existe une telle définition (file d'attente locale, file d'attente alias, file d'attente éloignée ou file d'attente modèle), l'appel échoue avec le code anomalie MQRC_OBJECT_NOT_UNIQUE.
- Si l'objet en cours d'ouverture est une liste de distribution (c'est-à-dire si *RecsPresent* est présent et supérieur à zéro), *ObjectName* doit être vide ou la chaîne null. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC_OBJECT_NAME_ERROR.
- Si *ObjectType* est MQOT_Q_MGR, des règles spéciales s'appliquent ; dans ce cas, le nom doit être entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone.

Il s'agit d'une zone d'entrée-sortie pour l'appel MQOPEN lorsque *ObjectName* est le nom d'une file d'attente modèle et d'une zone d'entrée uniquement dans tous les autres cas. La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

ObjectQMgrNom (MQCHAR48)

Il s'agit du nom du gestionnaire de files d'attente sur lequel l'objet *ObjectName* est défini. Les caractères valides dans le nom sont identiques à ceux de *ObjectName* (voir «ObjectName (MQCHAR48)», à la page 460). Un nom entièrement vide jusqu'au premier caractère NULL ou jusqu'à la fin de la zone indique le gestionnaire de files d'attente auquel l'application est connectée (le gestionnaire de files d'attente local).

Les points suivants s'appliquent aux types d'objet indiqués:

- Si *ObjectType* est MQOT_TOPIC, MQOT_NAMELIST, MQOT_PROCESS ou MQOT_Q_MGR, *ObjectQMgrName* doit être vide ou le nom du gestionnaire de files d'attente local.
- Si *ObjectName* est le nom d'une file d'attente modèle, le gestionnaire de files d'attente crée une file d'attente dynamique avec les attributs de la file d'attente modèle et renvoie dans la zone *ObjectQMgrName* le nom du gestionnaire de files d'attente sur lequel la file d'attente est créée ; il s'agit du nom du gestionnaire de files d'attente local. Une file d'attente modèle ne peut être indiquée que sur l'appel MQOPEN ; une file d'attente modèle n'est pas valide sur l'appel MQPUT1 .
- Si *ObjectName* est le nom d'une file d'attente de cluster et que *ObjectQMgrName* est vide, la destination des messages envoyés à l'aide de l'identificateur de file d'attente renvoyé par l'appel MQOPEN est choisie par le gestionnaire de files d'attente (ou l'exit de charge de travail de cluster, s'il est installé) comme suit:
  - Si MQOO_BIND_ON_OPEN est spécifié, le gestionnaire de files d'attente sélectionne une instance particulière de la file d'attente de cluster lors du traitement de l'appel MQOPEN et tous les messages insérés à l'aide de cet identificateur de file d'attente sont envoyés à cette instance.
  - Si MQOO_BIND_NOT_FIXED est spécifié, le gestionnaire de files d'attente peut choisir une autre instance de la file d'attente de destination (résidant sur un gestionnaire de files d'attente différent dans le cluster) pour chaque appel MQPUT successif qui utilise ce descripteur de file d'attente.

  Si l'application doit envoyer un message à une instance *spécifique* d'une file d'attente de cluster (c'est-à-dire une instance de file d'attente résidant sur un gestionnaire de files d'attente particulier du cluster), l'application doit spécifier le nom de ce gestionnaire de files d'attente dans la zone *ObjectQMgrName* . Cela force le gestionnaire de files d'attente local à envoyer le message au gestionnaire de files d'attente de destination spécifié.

- Si *ObjectName* est le nom d'une file d'attente partagée quiest détenu par le groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *ObjectQMgrName* peut être le nom du groupe de partage de files d'attente, le nom du gestionnaire de files d'attente local ou vide ; le message est placé dans la même file d'attente, quelle que soit la valeur spécifiée.

  Les groupes de partage de files d'attente sont pris en charge uniquement sous z/OS.
- Si *ObjectName* est le nom d'une file d'attente partagée appartenant à un groupe de partage de files d'attente éloignées (c'est-à-dire un groupe de partage de files d'attente auquel le gestionnaire de files d'attente local *n'appartient pas*), *ObjectQMgrName* muat est le nom du groupe de partage de files d'attente. Vous pouvez utiliser le nom d'un gestionnaire de files d'attente appartenant à ce groupe, mais cela peut retarder le message si ce gestionnaire de files d'attente particulier n'est pas disponible lorsque le message arrive au groupe de partage de files d'attente.
- Si l'objet ouvert est une liste de distribution (c'est-à-dire que *RecsPresent* est supérieur à zéro), *ObjectQMgrName* doit être vide ou la chaîne null. Si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC_OBJECT_Q_MGR_NAME_ERROR.

Il s'agit d'une zone d'entrée-sortie pour l'appel MQOPEN lorsque *ObjectName* est le nom d'une file d'attente modèle et d'une zone d'entrée uniquement dans tous les autres cas. La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

ObjectRec(MQLONG)

Décalage en octets du premier enregistrement d'objet MQOR à partir du début de la structure MQOD. Le décalage peut être positif ou négatif. *ObjectRecOffset* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *RecsPresent* a la valeur zéro.

Lorsqu'une liste de distribution est ouverte, un tableau d'un ou de plusieurs enregistrements d'objet MQOR doit être fourni afin de spécifier les noms des files d'attente de destination dans la liste de distribution. Cette opération peut être effectuée de l'une des deux manières suivantes:

- En utilisant la zone de décalage *ObjectRecOffset*.

  Dans ce cas, l'application doit déclarer sa propre structure contenant un MQOD suivi du tableau d'enregistrements MQOR (avec autant d'éléments de tableau que nécessaire) et définir *ObjectRecOffset* sur le décalage du premier élément du tableau à partir du début du MQOD. Assurez-vous que ce décalage est correct et qu'il comporte une valeur pouvant être prise en charge dans un MQLONG (le langage de programmation le plus restrictif est COBOL, pour lequel la plage valide est comprise entre -999 999 999 et +999 999 999).

  Utilisez *ObjectRecOffset* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur d'une manière qui n'est pas portable dans des environnements différents (par exemple, le langage de programmation COBOL).
- En utilisant la zone de pointeur *ObjectRecPtr*.

  Dans ce cas, l'application peut déclarer le tableau de structures MQOR séparément de la structure MQOD et définir *ObjectRecPtr* sur l'adresse du tableau.

  Utilisez *ObjectRecPtr* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

Quelle que soit la technique que vous choisissez, utilisez *ObjectRecOffset* et *ObjectRecPtr* ; l'appel échoue avec le code anomalie MQRC_OBJECT_RECORDS_ERROR si les deux sont nuls ou non nuls.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_2.

ObjectRecPtr (MQPTR)

Il s'agit de l'adresse du premier enregistrement d'objet MQOR. *ObjectRecPtr* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *RecsPresent* a la valeur zéro.

Vous pouvez utiliser *ObjectRecPtr* ou *ObjectRecOffset* pour spécifier les enregistrements d'objet, mais pas les deux. Pour plus de détails, voir la description de la zone *ObjectRecOffset* ci-dessus. Si vous n'utilisez pas *ObjectRecPtr*, définissez-le sur le pointeur null ou sur les octets null.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_2.

Remarque : Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

ObjectString (MQCHARV)
La zone ObjectString indique le nom d'objet long.

Indique le nom d'objet long à utiliser. Cette zone n'est référencée que pour certaines valeurs de *ObjectType*et est ignorée pour toutes les autres valeurs. Voir la description de *ObjectType* pour plus de détails sur les valeurs qui indiquent que cette zone est utilisée.

Si *ObjectString* est spécifié de manière incorrecte, en fonction de la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC_OBJECT_STRING_ERROR.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*. Pour plus de détails sur l'utilisation de ces deux zones, voir «Utilisation de chaînes de rubrique», à la page 562.

ObjectType (MQLONG)

Type d'objet nommé dans le descripteur d'objet. Les valeurs possibles sont les suivantes :

MQOT_CLNTCONN_CHANNEL
Canal de connexion client. Le nom de l'objet se trouve dans la zone *ObjectName* .

MQOT_Q
File d'attente. Le nom de l'objet se trouve dans la zone *ObjectName* .

MQOT_NAMELIST
NAMELIST. Le nom de l'objet se trouve dans la zone *ObjectName*

PROCESSUS MQ
Définition de processus. Le nom de l'objet se trouve dans la zone *ObjectName*

MQOT_Q_DIR
Gestionnaire de files d'attente. Le nom de l'objet se trouve dans la zone *ObjectName*

MQOT_TOPIC
:NONE. Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*.

Pour plus de détails sur l'utilisation de ces deux zones, voir «Utilisation de chaînes de rubrique», à la page 562.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQOT_Q.

RecsPresent (MQLONG)

Il s'agit du nombre d'enregistrements d'objet MQOR qui ont été fournis par l'application. Si ce nombre est supérieur à zéro, cela indique qu'une liste de distribution est en cours d'ouverture, *RecsPresent* étant le nombre de files d'attente de destination dans la liste. Une liste de distribution ne peut contenir qu'une seule destination.

La valeur de *RecsPresent* ne doit pas être inférieure à zéro, et si elle est supérieure à zéro, *ObjectType* doit être MQOT_Q ; l'appel échoue avec le code anomalie MQRC_RECS_Présentat_ERROR si ces conditions ne sont pas satisfaites.

Sous z/OS, cette zone doit être égale à zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_2.

Chaîne ResObject(MQCHARV)
La zone ResObjectString est le nom d'objet long après que le gestionnaire de files d'attente a résolu le nom fourni dans la zone *ObjectName* .

Cette zone est renvoyée uniquement pour les rubriques et les alias de file d'attente qui font référence à un objet de rubrique.

Si le nom d'objet long est fourni dans *ObjectString* et que rien n'est fourni dans *ObjectName*, la valeur renvoyée dans cette zone est la même que celle fournie dans *ObjectString*.

Si cette zone est omise (c'est-à-dire si ResObjectString.VSBufSize est égal à zéro), *ResObjectString* ne sera pas renvoyé, mais la longueur sera renvoyée dans ResObjectString.VSLength.

Si la longueur de la mémoire tampon (fournie dans ResObjectStrng.VSBufSize) est inférieure à la taille complète *ResObjectString*, la chaîne est tronquée et renvoie autant de caractères les plus à droite que possible dans la mémoire tampon fournie.

Si *ResObjectString* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC_RES_OBJECT_STRING_ERROR.

ResolvedQMgr(MQCHAR48)

Il s'agit du nom du gestionnaire de files d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom du gestionnaire de files d'attente qui possède la file d'attente identifiée par *ResolvedQName*. *ResolvedQMgrName* peut être le nom du gestionnaire de files d'attente local.

Si *ResolvedQName* est une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *ResolvedQMgrName* est le nom du groupe de partage de files d'attente. Si la file d'attente appartient à un autre groupe de partage de files d'attente, *ResolvedQName* peut être le nom du groupe de partage de files d'attente ou le nom d'un gestionnaire de files d'attente membre du groupe de partage de files d'attente (la nature de la valeur renvoyée est déterminée par les définitions de files d'attente qui existent sur le gestionnaire de files d'attente local).

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ouverte pour l'exploration, l'entrée ou la sortie (ou toute combinaison). Si l'objet ouvert est l'un des suivants, *ResolvedQMgrName* est mis à blanc:

- N'est pas une file d'attente
- Une file d'attente, mais non ouverte pour l'exploration, l'entrée ou la sortie
- Une file d'attente de cluster avec MQOO_BIND_NOT_FIXED spécifiée (ou avec MQOO_BIND_AS_Q_DEF en vigueur lorsque l'attribut de file d'attente *DefBind* a la valeur MQBND_BIND_NOT_FIXED)
- Une liste de distribution

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_3.

ResolvedQName (MQCHAR48)

Il s'agit du nom de la file d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom d'une file d'attente qui existe sur le gestionnaire de files d'attente identifié par *ResolvedQMgrName*.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ouverte pour l'exploration, l'entrée ou la sortie (ou toute combinaison). Si l'objet ouvert est l'un des suivants, *ResolvedQName* est mis à blanc:

- N'est pas une file d'attente
- Une file d'attente, mais non ouverte pour l'exploration, l'entrée ou la sortie
- Une liste de distribution
- Une file d'attente alias qui fait référence à un objet de rubrique (voir ResObjectString à la place).
- File d'attente alias qui se résout en objet de rubrique.

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_3.

ResolvedType (MQLONG)
Type de l'objet résolu (de base) en cours d'ouverture.

Les valeurs possibles sont les suivantes:

MQOT_Q
L'objet résolu est une file d'attente. Cette valeur s'applique lorsqu'une file d'attente est ouverte directement ou lorsqu'une file d'attente alias pointant vers une file d'attente est ouverte.

MQOT_TOPIC
L'objet résolu est une rubrique. Cette valeur s'applique lorsqu'une rubrique est ouverte directement ou lorsqu'une file d'attente alias pointant vers un objet de rubrique est ouverte.

MQOT_AUCUN
Le type résolu n'est ni une file d'attente ni une rubrique.

ResponseRec(MQLONG)

Décalage en octets du premier enregistrement de réponse MQRR à partir du début de la structure MQOD. Le décalage peut être positif ou négatif. *ResponseRecOffset* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *RecsPresent* a la valeur zéro.

Lorsqu'une liste de distribution est ouverte, vous pouvez fournir un tableau d'un ou de plusieurs enregistrements de réponse MQRR afin d'identifier les files d'attente dont l'ouverture a échoué (zone*CompCode* dans MQRR) et la raison de chaque échec (zone*Reason* dans MQRR). Les données sont renvoyées dans le tableau des enregistrements de réponse dans le même ordre que les noms de file d'attente dans le tableau des enregistrements d'objet. Le gestionnaire de files d'attente définit les enregistrements de réponse uniquement lorsque le résultat de l'appel est mixte (c'est-à-dire que certaines files d'attente ont été ouvertes avec succès alors que d'autres ont échoué, ou que toutes ont échoué, mais pour des raisons différentes) ; le code anomalie MQRC_MULTIPLE_MOTIFS de l'appel indique ce cas. Si le même code anomalie s'applique à toutes les files d'attente, ce code anomalie est renvoyé dans le paramètre *Reason* de l'appel MQOPEN ou MQPUT1 et les enregistrements de réponse ne sont pas définis. Les enregistrements de réponse sont facultatifs, mais s'ils sont fournis, ils doivent être *RecsPresent* .

Les enregistrements de réponse peuvent être fournis de la même manière que les enregistrements d'objet, soit en spécifiant un décalage dans *ResponseRecOffset*, soit en spécifiant une adresse dans *ResponseRecPtr*; voir la description de *ObjectRecOffset* ci-dessus pour plus de détails sur la procédure à suivre. Toutefois, *ResponseRecOffset* et *ResponseRecPtr* ne peuvent pas être utilisés ; l'appel échoue avec le code anomalie MQRC_RESPONSE_RECORDS_ERROR si les deux sont différents de zéro.

Pour l'appel MQPUT1 , ces enregistrements de réponse sont utilisés pour renvoyer des informations sur les erreurs qui se produisent lorsque le message est envoyé aux files d'attente de la liste de distribution, ainsi que sur les erreurs qui se produisent lorsque les files d'attente sont ouvertes. Le code achèvement et le code anomalie de l'opération d'insertion pour une file d'attente remplacent ceux de l'opération d'ouverture pour cette file d'attente uniquement si le code achèvement de cette dernière était MQCC_OK ou MQCC_WARNING.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_2.

ResponseRecPtr (MQPTR)

Il s'agit de l'adresse du premier enregistrement de réponse MQRR. *ResponseRecPtr* est utilisé uniquement lorsqu'une liste de distribution est ouverte. La zone est ignorée si *RecsPresent* a la valeur zéro.

Utilisez *ResponseRecPtr* ou *ResponseRecOffset* pour spécifier les enregistrements de réponse, mais pas les deux ; voir la description de la zone *ResponseRecOffset* ci-dessus pour plus de détails. Si vous n'utilisez pas *ResponseRecPtr*, définissez-le sur le pointeur null ou sur les octets null.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_2.

Remarque : Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

SelectionString (MQCHARV)
Il s'agit de la chaîne utilisée pour fournir les critères de sélection utilisés lors de l'extraction de messages d'une file d'attente.

SelectionString ne doit pas être fourni dans les cas suivants:

- Si *ObjectType* n'est pas MQOT_Q
- Si la file d'attente en cours d'ouverture n'est pas ouverte à l'aide de l'une des options MQOO_BROWSE ou MQOO_INPUT_ *

Si *SelectionString* est fourni dans ces cas, l'appel échoue avec le code anomalie MQRC_SELECTOR_INVALID_FOR_TYPE.

Si *SelectionString* est spécifié de manière incorrecte, conformément à la description du mode d'utilisation de la structure «MQCHARV-Chaîne de longueur variable», à la page 276 , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC_SELECTION_STRING_ERROR. La longueur maximale de *SelectionString* est MQ_SELECTOR_LENGTH.

L'utilisation de *SelectionString* est décrite dans Sélecteurs.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID_STRUC_MQODE
Identificateur de la structure de descripteur d'objet.

Pour le langage de programmation C, la constante MQOD_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQOD_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQOD_STRUC_ID.

Nombre UnknownDest(MQLONG)

Il s'agit du nombre de files d'attente de la liste de distribution qui sont résolues en files d'attente éloignées et qui ont été ouvertes avec succès. Si cette zone est présente, elle est également définie lors de l'ouverture d'une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQOD_VERSION_1.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être l'une des suivantes:

MQOD_VERSION_1
Structure de descripteur d'objet Version-1 .

MQOD_VERSION_2
Structure de descripteur d'objet Version-2 .

MQOD_VERSION_3
Structure de descripteur d'objet Version-3 .

MQOD_VERSION_4
Structure de descripteur d'objet Version-4 .

Toutes les versions sont prises en charge dans tous les environnements WebSphere MQ V7.0 .

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

VERSION MQOD_CURRENT_VERSION
Version actuelle de la structure de descripteur d'objet.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQOD_VERSION_1.

Valeurs initiales et déclarations de langue pour MQOD

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQODE	'OD¬¬'
Version	MQOD_VERSION_1	1
ObjectType	MQOT_Q	1
ObjectName	Aucun	Chaîne nulle ou blancs
ObjectQMgrName	Aucun	Chaîne nulle ou blancs
DynamicQName	Aucun	'CSQ.*' sous z/OS; 'AMQ.*' dans le cas contraire
AlternateUserId	Aucun	Chaîne nulle ou blancs
RecsPresent	Aucun	0
KnownDestCount	Aucun	0
UnknownDestCount	Aucun	0
InvalidDestCount	Aucun	0
ObjectRecOffset	Aucun	0
ResponseRecOffset	Aucun	0
ObjectRecPtr	Aucun	Pointeur null ou octets null
ResponseRecPtr	Aucun	Pointeur null ou octets null
AlternateSecurityId	MQSID_AUCUN	Valeurs NULL
ResolvedQName	Aucun	Chaîne nulle ou blancs
ResolvedQMgrName	Aucun	Chaîne nulle ou blancs
ObjectString	MQCHARV_VALEUR PAR DEFAUT	Comme défini pour MQCHARV
SelectionString	MQCHARV_VALEUR PAR DEFAUT	Comme défini pour MQCHARV
ResObjectString	MQCHARV_VALEUR PAR DEFAUT	Comme défini pour MQCHARV
ResolvedType	MQOT_AUCUN	0

Nom de zone	Nom de la constante	Valeur de la constante

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQOD_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQOD MyOD = {MQOD_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
** 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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Enregistrement d'objet

Le tableau suivant récapitule les zones de la structure.

Tableau 521. Zones dans MQOR

Zone	Description	Topic
ObjectName	Nom d'objet	ObjectName
ObjectQMgrName	Nom du gestionnaire de files d'attente d'objets	ObjectQMgrName

Présentation de MQOR

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus les clients WebSphere MQ MQI connectés à ces systèmes.

Objectif: Utilisez la structure MQOR pour spécifier le nom de file d'attente et le nom de gestionnaire de files d'attente d'une file d'attente de destination unique. MQOR est une structure d'entrée pour les appels MQOPEN et MQPUT1 .

Jeu de caractères et codage: les données dans MQOR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Syntaxe: en fournissant un tableau de ces structures sur l'appel MQOPEN, vous pouvez ouvrir une liste de files d'attente ; cette liste est appelée *liste de distribution*. Chaque insertion de message à l'aide de l'identificateur de file d'attente renvoyé par cet appel MQOPEN est placée dans chacune des files d'attente de la liste, à condition que l'ouverture de la file d'attente ait abouti.

Zones pour MQOR

La structure MQOR contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

ObjectName (MQCHAR48)

Il s'agit de la même zone que la zone *ObjectName* dans la structure MQOD (voir MQOD pour plus de détails), sauf que:

- Il doit s'agir du nom d'une file d'attente.
- Il ne doit pas s'agir du nom d'une file d'attente modèle.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

ObjectQMgrNom (MQCHAR48)

Il s'agit de la même zone que la zone *ObjectQMgrName* dans la structure MQOD (voir MQOD pour plus de détails).

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

Valeurs initiales et déclarations de langue pour MQOR

Tableau 522. Valeurs initiales des zones dans MQOR pour MQOR

Nom de zone	Nom de la constante	Valeur de la constante
ObjectName	Aucun	Chaîne nulle ou blancs
ObjectQMgrName	Aucun	Chaîne nulle ou blancs

Remarques :

1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
2. Dans le langage de programmation C, la variable macroMQOR_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQOR MyOR = {MQOR_DEFAULT};
```

Déclaration C

```
typedef struct tagMQOR MQOR;
struct tagMQOR {
  MQCHAR48  ObjectName;      /* Object name */
  MQCHAR48  ObjectQMgrName;  /* Object queue manager name */
};
```

Déclaration COBOL

```
**   MQOR structure
  10 MQOR.
**    Object name
   15 MQOR-OBJECTNAME     PIC X(48).
**    Object queue manager name
   15 MQOR-OBJECTQMGRNAME PIC X(48).
```

Déclaration PL/I

```
dcl
 1 MQOR based,
  3 ObjectName     char(48), /* Object name */
  3 ObjectQMgrName char(48); /* Object queue manager name */
```

Déclaration Visual Basic

```
Type MQOR
  ObjectName     As String*48 'Object name'
  ObjectQMgrName As String*48 'Object queue manager name'
End Type
```

MQPD-Descripteur de propriété

Le tableau suivant récapitule les zones de la structure.

Tableau 523. Zones dans MQPD

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options	Options
Support	Prise en charge requise pour la propriété de message	Support
Context	Contexte de message auquel appartient la propriété	Contexte
CopyOptions	Options de copie auxquelles appartient la propriété	CopyOptions

Présentation de MQPD

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OS et les clients WebSphere MQ MQI.

Objectif: **MQPD** est utilisé pour définir les attributs d'une propriété. La structure est un paramètre d'entrée-sortie dans l'appel MQSETMP et un paramètre de sortie dans l'appel MQINQMP.

Jeu de caractères et codage: les données dans **MQPD** doivent être dans le jeu de caractères de l'application et le codage de l'application (**MQENC_NATIVE**).

Zones pour MQPD

La structure MQPD contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

Contexte (MQLONG)

Décrit le contexte de message auquel appartient la propriété.

Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par WebSphere MQque le gestionnaire de files d'attente reconnaît comme incorrecte, il corrige la valeur de la zone *Context* .

L'option suivante peut être spécifiée:

CONTEXTE MQPD_USER_
La propriété est associée au contexte utilisateur.

Aucune autorisation spéciale n'est requise pour pouvoir définir une propriété associée au contexte utilisateur à l'aide de l'appel MQSETMP.

Sur un gestionnaire de files d'attente WebSphere MQ version 7.0 , une propriété associée au contexte utilisateur est sauvegardée comme décrit pour MQOO_SAVE_ALL_CONTEXT. Un appel

MQPUT avec MQPMO_PASS_ALL_CONTEXT spécifié entraîne la copie de la propriété à partir du contexte sauvegardé dans le nouveau message.

Si l'option décrite précédemment n'est pas requise, l'option suivante peut être utilisée:

MQPD_NO_CONTEXT
La propriété n'est pas associée à un contexte de message.

Une valeur non reconnue est rejetée avec le code *Reason*MQRC_PD_ERROR

Il s'agit d'une zone d'entrée-sortie de l'appel MQSETMP et d'une zone de sortie de l'appel MQINQMP. La valeur initiale de cette zone est MQPD_NO_CONTEXT.

CopyOptions (MQLONG)

Décrit le type de message dans lequel la propriété doit être copiée. Il s'agit d'une zone de sortie uniquement pour les propriétés WebSphere MQ définies ; WebSphere MQ définit la valeur appropriée.

Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie WebSphere MQ que le gestionnaire de files d'attente reconnaît comme incorrecte, il corrige la valeur de la zone *CopyOptions* .

Vous pouvez spécifier une ou plusieurs de ces options et, si vous en avez besoin, les valeurs peuvent être:

- ajoutées les unes aux autres (n'ajoutez pas la même constante plusieurs fois), ou
- combinées par le biais de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations par bit).

MQCOPY_FORWARD
Cette propriété est copiée dans un message en cours de transfert.

MQCOPY_PUBLICATION
Cette propriété est copiée dans le message reçu par un abonné lors de la publication d'un message.

REPONSE_MQCOPY_RÉPONSE
Cette propriété est copiée dans un message de réponse.

RAPPORT MQCOPY_REPORT
Cette propriété est copiée dans un message de rapport.

MQCOPY_ALL
Cette propriété est copiée dans tous les types de messages suivants.

Option par défaut: L'option suivante peut être spécifiée pour fournir l'ensemble d'options de copie par défaut:

MQCOPY_DEFAULT
Cette propriété est copiée dans un message en cours de transfert, dans un message de rapport ou dans un message reçu par un abonné lors de la publication d'un message.

Cela revient à spécifier la combinaison des options MQCOPY_FORWARD, MQCOPY_REPORT et MQCOPY_PUBLISH.

Si aucune des options décrites ci-dessus n'est requise, utilisez l'option suivante:

MQCOPY_NONE
Utilisez cette valeur pour indiquer qu'aucune autre option de copie n'est spécifiée ; à l'aide d'un programme, aucune relation n'existe entre cette propriété et les messages suivants. Cette valeur est toujours renvoyée pour les propriétés de descripteur de message.

Il s'agit d'une zone d'entrée-sortie de l'appel MQSETMP et d'une zone de sortie de l'appel MQINQMP. La valeur initiale de cette zone est MQCOPY_DEFAULT.

Options (MQLONG)

La valeur doit être:

MQPD_AUCUN
Aucune option spécifiée

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQPD_NONE.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID_STRUC_MQD
Identificateur de la structure de descripteur de propriété.

Pour le langage de programmation C, la constante **MQPD_STRUC_ID_ARRAY** est également définie ; elle a la même valeur que **MQPD_STRUC_ID**, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQPD_STRUC_ID**.

Prise en charge (MQLONG)

Cette zone indique le niveau de prise en charge de la propriété de message requis par le gestionnaire de files d'attente pour que le message contenant cette propriété soit inséré dans une file d'attente. Cela s'applique uniquement aux propriétés définies par WebSphere MQ; la prise en charge de toutes les autres propriétés est facultative.

La zone est automatiquement définie sur la valeur correcte lorsque la propriété WebSphere MQdéfinie est connue du gestionnaire de files d'attente. Si la propriété n'est pas reconnue, MQPD_SUPPORT_OPTIONAL est affectée. Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété définie par WebSphere MQque le gestionnaire de files d'attente reconnaît comme incorrecte, il corrige la valeur de la zone *Support* .

Lors de la définition d'une propriété WebSphere MQà l'aide de l'appel MQSETMP sur un descripteur de message dans lequel l'option MQCMHO_NO_VALIDATION a été définie, *Support* devient une zone d'entrée. Cela permet à une application d'insérer une propriété WebSphere MQdéfinie, avec la valeur correcte, où la propriété n'est pas prise en charge par le gestionnaire de files d'attente connecté, mais où le message doit être traité sur un autre gestionnaire de files d'attente.

La valeur MQPD_SUPPORT_OPTIONAL est toujours affectée à des propriétés qui ne sont pas des propriétés définies par WebSphere MQ.

Si un gestionnaire de files d'attente WebSphere MQ Version 7.0 , qui prend en charge les propriétés de message, reçoit une propriété contenant une valeur *Support* non reconnue, la propriété est traitée comme si:

- MQPD_SUPPORT_REQUIRED a été spécifié si l'une des valeurs non reconnues est contenue dans MQPD_REJECT_UNSUP_MASK.
- MQPD_SUPPORT_REQUIRED_IF_LOCAL a été spécifié si l'une des valeurs non reconnues est contenue dans le masque MQPD_ACCEPT_UNSUP_IF_XMIT_MASK
- MQPD_SUPPORT_OPTIONAL a été spécifié dans le cas contraire.

L'une des valeurs suivantes est renvoyée par l'appel MQINQMP ou l'une des valeurs peut être spécifiée lors de l'utilisation de l'appel MQSETMP sur un descripteur de message dans lequel l'option MQCMHO_NO_VALIDATION est définie:

MQPD_SUPPORT_XX_ENCODE_CASE_ONE facultatif
La propriété est acceptée par un gestionnaire de files d'attente même si elle n'est pas prise en charge. La propriété peut être supprimée pour que le message soit transmis à un gestionnaire de files d'attente qui ne prend pas en charge les propriétés de message. Cette valeur est également affectée aux propriétés qui ne sont pas définies par WebSphere MQ.

MQPD_SUPPORT_REQUIS
La prise en charge de la propriété est requise. Le message est rejeté par un gestionnaire de files d'attente qui ne prend pas en charge la propriété WebSphere MQdéfinie. L'appel MQPUT ou MQPUT1 échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_UNSUPPORTED_PROPERTY.

MQPD_SUPPORT_REQUIRED_IF_LOCAL
Le message est rejeté par un gestionnaire de files d'attente qui ne prend pas en charge la propriété WebSphere MQsi le message est destiné à une file d'attente locale. L'appel MQPUT ou MQPUT1 échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_UNSUPPORTED_PROPERTY.

L'appel MQPUT ou MQPUT1 aboutit si le message est destiné à un gestionnaire de files d'attente éloignées.

Il s'agit d'une zone de sortie dans l'appel MQINQMP et d'une zone d'entrée dans l'appel MQSETMP si le descripteur de message a été créé avec l'option MQCMHO_NO_VALIDATION définie. La valeur initiale de cette zone est MQPD_SUPPORT_OPTIONAL.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQPD_VERSION_1
Structure de descripteur de propriété Version-1 .

La constante suivante indique le numéro de version de la version en cours:

MQPD_CURRENT_VERSION
Version actuelle de la structure de descripteur de propriété.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQPD_VERSION_1**.

Valeurs initiales et déclarations de langage pour MQPD

Tableau 524. Valeurs initiales des zones dans MQPD

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQD	'PD'
Version	MQPD_VERSION_1	1
Options	MQPD_AUCUN	0
Support	MQPD_SUPPORT_XX_ENCODE_CASE_ONE facultatif	0
Context	MQPD_NO_CONTEXT	0
CopyOptions	MQCOPY_DEFAULT	0

Remarques :

1. Dans le langage de programmation C, la variable de macro MQPD_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQPD MyPD = {MQPD_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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-Options d'insertion de message

Le tableau suivant récapitule les zones de la structure.

Tableau 525. Structure MQPMO

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options qui contrôlent l'action de MQPUT et MQPUT1	Options
Timeout	Réservé	Timeout
Context	Descripteur d'objet de la file d'entrée	Contexte
KnownDestCount	Nombre de messages envoyés correctement aux files d'attente locales	KnownDestCount

Tableau 525. Structure MQPMO (suite)		
Zone	**Description**	**Topic**
UnknownDestCount	Nombre de messages envoyés correctement aux files d'attente éloignées	UnknownDestCount
InvalidDestCount	Nombre de messages n'ayant pas pu être envoyés	InvalidDestCount
ResolvedQName	Nom résolu de la file d'attente de destination	ResolvedQName
ResolvedQMgrName	Nom résolu du gestionnaire de files d'attente de destination	ResolvedQMgrNom
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQPMO_VERSION_2.		
RecsPresent	Nombre d'enregistrements d'insertion de message ou d'enregistrements de réponse présents	RecsPresent
PutMsgRecFields	Indicateurs identifiant les zones MQPMR présentes	PutMsgRecFields
PutMsgRecOffset	Décalage du premier enregistrement d'insertion de message à partir du début de MQPMO	PutMsgRecOffset
ResponseRecOffset	Décalage du premier enregistrement de réponse à partir du début de MQPMO	DécalageResponseRec
PutMsgRecPtr	Adresse du premier enregistrement d'insertion de message	PutMsgRecPtr
ResponseRecPtr	Adresse du premier enregistrement de réponse	ResponseRecPtr
Remarque : Les zones restantes sont ignorées si *Version* est inférieur à MQPMO_VERSION_3.		
OriginalMsgHandle	Descripteur de message d'origine	OriginalMsgIdentificateur
NewMsgHandle	Nouveau descripteur de message	DescripteurNewMsg
Action	Type d'insertion effectuée et relation entre le message d'origine spécifié par la zone *OriginalMsgHandle* et le nouveau message spécifié par la zone *NewMsgHandle*	Action
PubLevel	Niveau d'abonnement ciblé par la publication	PubLevel

Présentation de MQPMO

Disponibilité: Tous les systèmes WebSphere MQ , plus les clients WebSphere MQ connectés à ces systèmes.

Objectif: La structure MQPMO permet à l'application de spécifier des options qui contrôlent la façon dont les messages sont placés dans des files d'attente ou publiés dans des rubriques. La structure est un paramètre d'entrée-sortie dans les appels MQPUT et MQPUT1 .

Version: la version actuelle de MQPMO est MQPMO_VERSION_3. Certaines zones sont disponibles uniquement dans certaines versions de MQPMO. Si vous devez porter des applications entre plusieurs environnements, vous devez vous assurer que la version de MQPMO est cohérente dans tous les environnements. Les zones qui existent uniquement dans des versions particulières de la structure sont

identifiées comme telles dans «MQPMO-Options d'insertion de message», à la page 477 et dans les descriptions de zone.

Les fichiers d'en-tête, COPY et INCLUDE fournis pour les langages de programmation pris en charge contiennent la version la plus récente de MQPMO prise en charge par l'environnement, mais avec la valeur initiale de la zone *Version* définie sur MQPMO_VERSION_1. Pour utiliser des zones qui ne sont pas présentes dans la structure version-1 , l'application doit définir la zone *Version* sur le numéro de version de la version requise.

Jeu de caractères et codage: les données dans MQPMO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Zones pour MQPMO

La structure MQPMO contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

Action (MQLONG)
Indique le type d'insertion effectuée et la relation entre le message d'origine spécifié par la zone Descripteur OriginalMsget le nouveau message spécifié par la zone Descripteur NewMsg. Les propriétés du message sont choisies par le gestionnaire de files d'attente en fonction de la valeur de l'action spécifiée.

Vous pouvez choisir de fournir le contenu du descripteur de message à l'aide du paramètre MsgDesc dans les appels MQPUT ou MQPUT1 . Il est également possible de ne pas fournir le paramètre MsgDesc ou de spécifier qu'il s'agit d'une sortie uniquement en incluant MQPMO_MD_FOR_OUTPUT_ONLY dans la zone Options de la structure MQPMO.

Si le paramètre MsgDesc n'est pas fourni ou s'il est spécifié comme étant uniquement en sortie, le descripteur de message du nouveau message est renseigné à partir des zones de descripteur de message du MQPMO, conformément aux règles décrites dans cette rubrique.

Le paramètre de contexte et les activités de transmission décrites dans Contrôle des informations de contexte prennent effet une fois que le descripteur de message a été composé.

Si une valeur d'action incorrecte est spécifiée, l'appel échoue avec le code anomalie MQRC_ACTION_ERROR.

L'une des actions suivantes peut être spécifiée:

MQACTP_NOUVEAU
Un nouveau message est inséré et aucune relation avec un message précédent n'est indiquée par le programme. Le descripteur de message se compose comme suit:

- Si une valeur MsgDesc est fournie dans l'appel MQPUT ou MQPUT1 et que MQPMO_MD_FOR_OUTPUT_ONLY ne figure pas dans MQPMO.Options, utilisé comme descripteur de message non modifié.
- Si MsgDesc n'est pas fourni, ou si MQPMO_MD_FOR_OUTPUT_ONLY se trouve dans MQPMO.Options : le gestionnaire de files d'attente génère le descripteur de message à l'aide d'une combinaison de propriétés des identificateurs OriginalMsget NewMsg. Les zones de descripteur de message définies explicitement sur le nouveau descripteur de message sont prioritaires par rapport à celles du descripteur de message d'origine.

Les données de message sont extraites du paramètre de mémoire tampon MQPUT ou MQPUT1 .

MQACTP_FORWARD
Un message précédemment extrait est en cours de transfert. Le descripteur de message d'origine indique le message précédemment extrait.

Le nouveau descripteur de message spécifie toutes les modifications apportées aux propriétés (y compris celles du descripteur de message) dans le descripteur de message d'origine.

Le descripteur de message se compose comme suit:

- Si une valeur MsgDesc est fournie dans l'appel MQPUT ou MQPUT1 et que MQPMO_MD_FOR_OUTPUT_ONLY ne figure pas dans MQPMO.Options, utilisé comme descripteur de message non modifié.
- Si MsgDesc n'est pas fourni, ou si MQPMO_MD_FOR_OUTPUT_ONLY se trouve dans MQPMO.Options : le gestionnaire de files d'attente génère le descripteur de message à l'aide d'une combinaison de propriétés des identificateurs OriginalMsget NewMsg. Les zones de descripteur de message définies explicitement sur le nouveau descripteur de message sont prioritaires par rapport à celles du descripteur de message d'origine.
- Si MQPMO_NEW_MSG_ID ou MQPMO_NEW_CORREL_ID sont spécifiés dans MQPMO.Optionssont alors honorées.

Les propriétés de message sont composées comme suit:

- Toutes les propriétés du descripteur de message d'origine qui ont MQCOPY_FORWARD dans MQPD.CopyOptions
- Toutes les propriétés du nouveau descripteur de message. Pour chaque propriété du nouveau descripteur de message ayant le même nom qu'une propriété du descripteur de message d'origine, la valeur est extraite du nouveau descripteur de message. La seule exception à cette règle est le cas particulier où la propriété du nouveau descripteur de message porte le même nom qu'une propriété du descripteur de message d'origine, mais que la valeur de la propriété est null. Dans ce cas, la propriété est supprimée du message.

Les données de message à transmettre sont extraites du paramètre de mémoire tampon MQPUT ou MQPUT1 .

MQACTP_REPLY
Une réponse est en cours à un message précédemment extrait. Le descripteur de message d'origine indique le message précédemment extrait.

Le nouveau descripteur de message spécifie toutes les modifications apportées aux propriétés (y compris celles du descripteur de message) dans le descripteur de message d'origine.

Le descripteur de message se compose comme suit:

- Si une valeur MsgDesc est fournie dans l'appel MQPUT ou MQPUT1 et que MQPMO_MD_FOR_OUTPUT_ONLY ne figure pas dans MQPMO.Options, utilisé comme descripteur de message non modifié.
- Si MsgDesc n'est pas fourni, ou si MQPMO_MD_FOR_OUTPUT_ONLY se trouve dans MQPMO.Options et les zones de descripteur de message initial sont choisies comme suit:

Tableau 526. Transformation du descripteur de message de réponse

Zone dans MQMD	**Valeur utilisée**
Rapport	Si MQRO_PASS_DISCARD_AND_EXPIRATION et MQRO_DISCARD_MSG sont définis: MQRO_DISCARD_MSG sinon MQRO_AUCUN
MsgType	MQMT_REPLY
Expiration	Si MQRO_PASS_DISCARD_AND_EXPIRATION est défini: Copié à partir du message d'entrée sinon MQEI_UNLIMITED
Commentaires	MQFB_AUCUN

Tableau 526. Transformation du descripteur de message de réponse (suite)	
Zone dans MQMD	**Valeur utilisée**
MsgId	Si MQPMO_NEW_MSG_ID est défini: Un nouvel identificateur de message est généré else if MQRO_PASS_MSG_ID est défini: Copié à partir du message d'entrée sinon MQMI_AUCUN
CorrelId	Si MQPMO_NEW_CORREL_ID est défini: Un nouvel identificateur de corrélation est généré else if MQRO_COPY_MSG_ID_TO_CORREL_ID est défini: Copié à partir de la zone MsgId de la Message d'entrée sinon, si MQRO_PASS_CORREL_ID est défini: Copié à partir de la zone CorrelId de la Message d'entrée sinon MQCI_NONE
BackoutCount	0
ReplyToQ	Espaces vides
ReplyToQMgr	Espaces vides
GroupId	MQGI_AUCUN
MsgSeqNumber	1
Décalage	0
MsgFlags	MQMF_AUCUN
OriginalLength	MQOL_UNDEFINI

- Le descripteur de message est ensuite modifié par le nouveau descripteur de message-toutes les zones de descripteur de message explicitement définies comme propriétés dans le nouveau descripteur de message sont prioritaires sur les zones de descripteur de message décrites ci-dessus.

Les propriétés de message sont composées comme suit:

- Toutes les propriétés du descripteur de message d'origine qui ont MQCOPY_REPLY dans MQPD.CopyOptions
- Toutes les propriétés du nouveau descripteur de message. Pour chaque propriété du nouveau descripteur de message ayant le même nom qu'une propriété du descripteur de message d'origine, la valeur est extraite du nouveau descripteur de message. La seule exception à cette règle est le cas particulier où la propriété du nouveau descripteur de message porte le même nom qu'une propriété du descripteur de message d'origine, mais que la valeur de la propriété est null. Dans ce cas, la propriété est supprimée du message.

Les données de message à transmettre sont extraites du paramètre de mémoire tampon MQPUT/ MQPUT1 .

RAPPORT MQACTP_RAPPORT
Un rapport est généré suite à un message précédemment extrait. Le descripteur de message d'origine indique le message à l'origine de la génération du rapport.

Le nouveau descripteur de message spécifie toutes les modifications apportées aux propriétés (y compris celles du descripteur de message) dans le descripteur de message d'origine.

Le descripteur de message se compose comme suit:

- Si une valeur MsgDesc est fournie dans l'appel MQPUT ou MQPUT1 et que MQPMO_MD_FOR_OUTPUT_ONLY ne figure pas dans MQPMO.Options, utilisé comme descripteur de message non modifié.
- Si MsgDesc n'est pas fourni, ou si MQPMO_MD_FOR_OUTPUT_ONLY se trouve dans MQPMO.Options et les zones de descripteur de message initial sont choisies comme suit:

Tableau 527. Transformation du descripteur de message de rapport

Zone dans MQMD	**Valeur utilisée**
Rapport	Si MQRO_PASS_DISCARD_AND_EXPIRATION et MQRO_DISCARD_MSG sont définis: MQRO_DISCARD_MSG sinon MQRO_AUCUN
MsgType	MQMT_REPORT
Expiration	Si MQRO_PASS_DISCARD_AND_EXPIRATION est défini: Copié à partir du message d'entrée sinon MQEI_UNLIMITED
MsgId	Si MQPMO_NEW_MSG_ID est défini: Un nouvel identificateur de message est généré else if MQRO_PASS_MSG_ID est défini: Copié à partir du message d'entrée sinon MQMI_AUCUN
CorrelId	Si MQPMO_NEW_CORREL_ID est défini: Un nouvel identificateur de corrélation est généré else if MQRO_COPY_MSG_ID_TO_CORREL_ID est défini: Copié à partir de la zone MsgId de la Message d'entrée sinon, si MQRO_PASS_CORREL_ID est défini: Copié à partir de la zone CorrelId de la Message d'entrée sinon MQCI_NONE
BackoutCount	0
ReplyToQ	Espaces vides
ReplyToQMgr	Espaces vides
OriginalLength	Défini sur *BufferLength*

- Le descripteur de message est ensuite modifié par le nouveau descripteur de message-toutes les zones de descripteur de message explicitement définies comme propriétés dans le nouveau descripteur de message sont prioritaires sur les zones de descripteur de message décrites ci-dessus.

Les propriétés de message sont composées comme suit:

- Toutes les propriétés du descripteur de message d'origine qui comportent MQCOPY_REPORT dans MQPD.CopyOptions
- Toutes les propriétés du nouveau descripteur de message. Pour chaque propriété du nouveau descripteur de message ayant le même nom qu'une propriété du descripteur de message d'origine, la valeur est extraite du nouveau descripteur de message. La seule exception à cette règle est le cas particulier où la propriété du nouveau descripteur de message porte le même nom qu'une propriété du descripteur de message d'origine, mais que la valeur de la propriété est null. Dans ce cas, la propriété est supprimée du message.

La zone Feedback dans le MQMD résultant représente le rapport à générer. Une valeur de commentaire en retour de MQFB_NONE entraîne l'échec de l'appel MQPUT ou MQPUT1 avec le code anomalie MQRC_FEEDBACK_ERROR.

Pour choisir les données utilisateur du message de rapport, WebSphere MQ consulte les zones Rapport et Commentaires dans le MQMD résultant, ainsi que les paramètres Buffer et BufferLength de l'appel MQPUT ou MQPUT1 .

- Si les commentaires en retour sont MQFB_COA, MQFB_COD ou MQFB_EXPIRATION, la valeur du rapport est inspectée.
- Si l'un des cas suivants est vrai, les données de message complètes de Buffer pour une longueur de BufferLength sont utilisées.
  - Les commentaires en retour sont MQFB_EXPIRATION et le rapport contient MQRO_EXPIRATION_WITH_FULL_DATA
  - Les commentaires en retour sont MQFB_COD et le rapport contient MQRO_COD_WITH_FULL_DATA
  - Les commentaires en retour sont MQFB_COA et le rapport contient MQRO_COA_WITH_FULL_DATA
- Si l'un des cas suivants est vrai, les 100 premiers octets du message (ou BufferLength si cette valeur est inférieure à 100) de Buffer sont utilisés.
  - Le retour d'informations est MQFB_EXPIRATION et le rapport contient MQRO_EXPIRATION_WITH_DATA
  - Les commentaires en retour sont MQFB_COD et le rapport contient MQRO_COD_WITH_DATA
  - Les commentaires en retour sont MQFB_COA et le rapport contient MQRO_COA_WITH_DATA
- Si les commentaires en retour sont MQFB_EXPIRATION, MQFB_COD ou MQFB_COA et que le rapport ne contient pas les options * _WITH_FULL_DATA ou * _WITH_DATA associées à cette valeur de rétrofacturation, aucune donnée utilisateur n'est incluse dans le message.
- Si les commentaires en retour prennent une valeur différente de celles répertoriées ci-dessus, Buffer et BufferLength sont utilisés normalement.

La dérivation des données utilisateur est présentée dans le tableau suivant:

Tableau 528. Source des données utilisateur

	MQFB_COA	**MQFB_COD**	**EXPIRATION_MQFB**
MQRO_EXPIRATION_WITH_FULL_DATA	Aucun	Aucun	Mémoire tampon (longueur de la mémoire tampon)
MQRO_COD_WITH_FULL_DATA	Aucun	Mémoire tampon (longueur de la mémoire tampon)	Aucun
MQRO_COA_WITH_FULL_DATA	Mémoire tampon (longueur de la mémoire tampon)	Aucun	Aucun

Tableau 528. Source des données utilisateur (suite)			
	MQFB_COA	**MQFB_COD**	**EXPIRATION_MQFB**
MQRO_EXPIRATION_WITH_DATA	Aucun	Aucun	Mémoire tampon (100 premiers octets)
MQRO_COD_WITH_DATA	Aucun	Mémoire tampon (100 premiers octets)	Aucun
MQRO_COA_WITH_DATA	Mémoire tampon (100 premiers octets)	Aucun	Aucun

Contexte (MQHOBJ)

Si MQPMO_PASS_IDENTITY_CONTEXT ou MQPMO_PASS_ALL_CONTEXT est spécifié, cette zone doit contenir le descripteur de file d'attente d'entrée à partir duquel les informations de contexte à associer au message en cours d'insertion sont extraites.

Si ni MQPMO_PASS_IDENTITY_CONTEXT ni MQPMO_PASS_ALL_CONTEXT n'est spécifié, cette zone est ignorée.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

Nombre d' InvalidDest(MQLONG)

Il s'agit du nombre de messages qui n'ont pas pu être envoyés aux files d'attente de la liste de distribution. Le nombre inclut les files d'attente dont l'ouverture a échoué, ainsi que les files d'attente dont l'ouverture a abouti mais pour lesquelles l'opération d'insertion a échoué. Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

Remarque : Cette zone est définie si le paramètre *CompCode* de l'appel MQPUT ou MQPUT1 est MQCC_OK ou MQCC_WARNING; elle peut être définie si le paramètre *CompCode* est MQCC_FAILED, mais ne repose pas sur ce paramètre dans le code de l'application.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *Version* est inférieur à MQPMO_VERSION_1.

Cette zone n'est pas définie sur z/OS car les listes de distribution ne sont pas prises en charge.

Nombre de KnownDest(MQLONG)

Il s'agit du nombre de messages que l'appel MQPUT ou MQPUT1 a envoyés avec succès aux files d'attente de la liste de distribution qui sont des files d'attente locales. Ce nombre n'inclut pas les messages envoyés à des files d'attente qui se résolvent en files d'attente éloignées (même si une file d'attente de transmission locale est utilisée initialement pour stocker le message). Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *Version* est inférieur à MQPMO_VERSION_1.

Cette zone n'est pas définie sur z/OS car les listes de distribution ne sont pas prises en charge.

Descripteur NewMsg(MQHMSG)

Il s'agit d'un descripteur facultatif pour le message en cours d'insertion en fonction de la valeur de la zone Action. Il définit les propriétés du message et remplace les valeurs de *OriginalMsgHandle*, si elles sont spécifiées.

En cas de retour de l'appel **MQPUT** ou **MQPUT1** , le contenu du descripteur reflète le message qui a été inséré.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est **MQHM_NONE**. Cette zone est ignorée si la version est inférieure à **MQPMO_VERSION_3**.

Options MQPMO (MQLONG)
La zone Options contrôle le fonctionnement des appels **MQPUT** et **MQPUT1** .

Option de portée. Vous pouvez spécifier l'une ou l'autre des options MQPMO. Si plusieurs options sont requises, les valeurs que vous spécifiez pour les options peuvent être utilisées de la manière suivante:

- Les valeurs peuvent être ajoutées. N'ajoutez pas la même constante plus d'une fois.
- Les valeurs peuvent être combinées à l'aide de l'opération OR bit à bit, si le langage de programmation prend en charge les opérations bit à bit.

Les combinaisons qui ne sont pas valides sont notées ; les autres combinaisons sont valides.

L'option suivante contrôle la portée des publications envoyées:

MQPMO_SCOPE_QMGR

La publication est envoyée uniquement aux abonnés qui se sont abonnés à ce gestionnaire de files d'attente. La publication n'est pas réacheminée vers les gestionnaires de files d'attente de publication / abonnement distants qui ont souscrit un abonnement à ce gestionnaire de files d'attente, ce qui remplace tout comportement défini à l'aide de l'attribut de rubrique PUBSCOPE.

Remarque : S'il n'est pas défini, la portée de la publication est déterminée par l'attribut de rubrique PUBSCOPE.

Options de publication. Les options suivantes contrôlent le mode de publication des messages dans une rubrique:

MQPMO_SUPPRESS_REPLYTO

Les informations indiquées dans les zones *ReplyToQ* et *ReplyToQMgr* du MQMD de cette publication ne sont pas transmises aux abonnés. Si cette option est utilisée avec une option de rapport qui requiert un *ReplyToQ*, l'appel échoue avec MQRC_MISSING_REPLY_TO_Q.

MQPMO_RETAIN

La publication envoyée doit être conservée par le gestionnaire de files d'attente. Cette conservation permet à un abonné de demander une copie de cette publication après sa publication, à l'aide de l'appel MQSUBRQ. Elle permet également d'envoyer une publication aux applications qui effectuent leur abonnement après la date de cette publication (sauf si elles choisissent de ne pas l'envoyer à l'aide de l'option MQSO_NEW_PUBLICATIONS_ONLY). Si une application reçoit une publication qui a été conservée, elle est indiquée par la propriété de message MQIsRetained de cette publication.

Une seule publication peut être conservée sur chaque noeud de l'arborescence de rubriques. Par conséquent, s'il existe déjà une publication conservée pour cette rubrique, publiée par toute autre application, elle est remplacée par cette publication. Il est donc préférable d'éviter que plus d'un diffuseur de publications conserve des messages sur le même sujet.

Lorsque des publications conservées sont demandées par un abonné, l'abonnement utilisé peut contenir un caractère générique dans la rubrique, auquel cas un certain nombre de publications conservées peuvent correspondre (sur différents noeuds de l'arborescence de rubriques) et plusieurs publications peuvent être envoyées à l'application demandeuse. Pour plus de détails, voir la description de l'appel à «MQSUBRQ-Demande d'abonnement», à la page 784 .

Pour plus d'informations sur la façon dont les publications conservées interagissent avec les niveaux d'abonnement, voir Interceptation des publications.

Si cette option est utilisée et que la publication ne peut pas être conservée, le message n'est pas publié et l'appel échoue avec MQRC_PUT_NOT_RETAINED.

MQPMO_NOT_OWN_SUBS
Indique au gestionnaire de files d'attente que l'application ne souhaite envoyer aucune de ses publications aux abonnements qu'elle possède. Les abonnements sont considérés comme appartenant à la même application si les descripteurs de connexion sont identiques.

MQPMO_WARN_SI_NO_SUBS_MIS en correspondance
Si aucun abonnement ne correspond à la publication, renvoyez un code achèvement (*CompCode*) de MQCC_WARNING et le code anomalie MQRC_NO_SUBS_MATCH.

Si MQRC_NO_SUBS_APPARIÉ est renvoyé par l'opération d'insertion, la publication n'a été distribuée à aucun abonnement. Toutefois, si l'option MQPMO_RETAIN est spécifiée sur l'opération d'insertion, le message est conservé et distribué à tout abonnement correspondant défini ultérieurement.

Un abonnement à la rubrique correspond à la publication si l'une des conditions suivantes est remplie:

- Le message est distribué dans la file d'attente d'abonnement
- Le message aurait été distribué à la file d'attente d'abonnement, mais un problème avec la file d'attente signifie que le message ne peut pas être placé dans la file d'attente et qu'il a donc été placé dans la file d'attente de rebut ou supprimé.
- Un exit de routage est défini pour supprimer la distribution du message à l'abonnement

Un abonnement à la rubrique ne correspond pas à la publication si l'une des conditions suivantes est remplie:

- L'abonnement comporte une chaîne de sélection qui ne correspond pas à la publication
- L'abonnement a spécifié l'option MQSO_PUBLICATION_ON_REQUEST
- La publication n'est pas distribuée car l'option MQPMO_NOT_OWN_SUBS a été spécifiée lors de l'opération d'insertion et l'abonnement correspond à l'identité du diffuseur de publications

Options de point de synchronisation. Les options suivantes concernent la participation de l'appel MQPUT ou MQPUT1 dans une unité d'oeuvre:

MQPMO_SYNCPOINT
La demande consiste à fonctionner dans le cadre des protocoles d'unité de travail normaux. Le message n'est pas visible en dehors de l'unité de travail tant que l'unité de travail n'est pas validée. Si l'unité d'oeuvre est annulée, le message est supprimé.

Si MQPMO_SYNCPOINT et MQPMO_NO_SYNCPOINT ne sont pas spécifiés, l'inclusion de la demande d'insertion dans les protocoles d'unité de travail est déterminée par l'environnement exécutant le gestionnaire de files d'attente et non par l'environnement exécutant l'application. Sous z/OS, la demande d'insertion se trouve dans une unité de travail. Dans tous les autres environnements, la demande d'insertion ne se trouve pas dans une unité de travail.

En raison de ces différences, une application que vous souhaitez porter ne doit pas autoriser cette option à utiliser par défaut ; indiquez MQPMO_SYNCPOINT ou MQPMO_NO_SYNCPOINT de manière explicite.

N'indiquez pas MQPMO_SYNCPOINT avec MQPMO_NO_SYNCPOINT.

MQPMO_NO_SYNCPOINT
La demande consiste à fonctionner en dehors des protocoles d'unité de travail normaux. Le message est disponible immédiatement et ne peut pas être supprimé par l'annulation d'une unité de travail.

Si MQPMO_NO_SYNCPOINT et MQPMO_SYNCPOINT ne sont pas spécifiés, l'inclusion de la demande d'insertion dans les protocoles d'unité de travail est déterminée par l'environnement exécutant le gestionnaire de files d'attente et non par l'environnement exécutant l'application. Sous z/OS, la demande d'insertion se trouve dans une unité de travail. Dans tous les autres environnements, la demande d'insertion ne se trouve pas dans une unité de travail.

En raison de ces différences, une application que vous souhaitez porter ne doit pas autoriser cette option à utiliser par défaut ; indiquez MQPMO_SYNCPOINT ou MQPMO_NO_SYNCPOINT de manière explicite.

Ne spécifiez pas MQPMO_NO_SYNCPOINT avec MQPMO_SYNCPOINT.

Options d'identificateur de message et d'identificateur de corrélation. Les options suivantes demandent au gestionnaire de files d'attente de générer un nouvel identificateur de message ou un nouvel identificateur de corrélation:

ID MQPMO_NEW_MSG_ID

Le gestionnaire de files d'attente remplace le contenu de la zone *MsgId* dans MQMD par un nouvel identificateur de message. Cet identificateur de message est envoyé avec le message et renvoyé à l'application dans la sortie de l'appel MQPUT ou MQPUT1 .

L'option MQPMO_NEW_MSG_ID peut également être spécifiée lorsque le message est inséré dans une liste de distribution. Pour plus de détails, voir la description de la zone *MsgId* dans la structure MQPMR.

L'utilisation de cette option évite à l'application de devoir réinitialiser la zone *MsgId* sur MQMI_NONE avant chaque appel MQPUT ou MQPUT1 .

MQPMO_NEW_CORREL_ID

Le gestionnaire de files d'attente remplace le contenu de la zone *CorrelId* dans MQMD par un nouvel identificateur de corrélation. Cet identificateur de corrélation est envoyé avec le message et renvoyé à l'application en sortie de l'appel MQPUT ou MQPUT1 .

L'option MQPMO_NEW_CORREL_ID peut également être spécifiée lorsque le message est inséré dans une liste de distribution. Pour plus de détails, voir la description de la zone *CorrelId* dans la structure MQPMR.

MQPMO_NEW_CORREL_ID est utile lorsque l'application requiert un identificateur de corrélation unique.

Options de groupe et de segment. Les options suivantes concernent le traitement des messages dans des groupes et des segments de messages logiques. Lisez les définitions qui suivent pour vous aider à comprendre l'option.

Avertissement : Vous ne pouvez pas utiliser de messages segmentés ou groupés avec la fonction de publication / abonnement.

Message physique

Il s'agit de la plus petite unité d'informations pouvant être placée dans une file d'attente ou supprimée d'une file d'attente ; elle correspond souvent aux informations spécifiées ou extraites sur un seul appel MQPUT, MQPUT1ou MQGET. Chaque message physique possède son propre descripteur de message (MQMD). En règle générale, les messages physiques se distinguent par des valeurs différentes pour l'identificateur de message (zone*MsgId* dans MQMD), bien que cela ne soit pas appliqué par le gestionnaire de files d'attente.

Message logique

Un message logique est une unité unique d'informations d'application pour les plateformes nonz/OS uniquement. En l'absence de contraintes système, un message logique est identique à un message physique. Toutefois, lorsque les messages logiques sont extrêmement volumineux, les contraintes du système peuvent rendre souhaitable ou nécessaire la division d'un message logique en deux ou plusieurs messages physiques, appelés *segments*.

Un message logique segmenté se compose d'au moins deux messages physiques ayant le même identificateur de groupe non nul (zone*GroupId* dans MQMD) et le même numéro de séquence de message (zone*MsgSeqNumber* dans MQMD). Les segments se distinguent par des valeurs différentes pour le décalage de segment (zone*Offset* dans MQMD), qui fournit le décalage des données dans le message physique à partir du début des données dans le message logique. Chaque segment étant un message physique, les segments d'un message logique ont généralement des identificateurs de message différents.

Un message logique qui n'a pas été segmenté, mais pour lequel la segmentation a été autorisée par l'application émettrice, possède également un identificateur de groupe non nul, bien que dans ce cas, il n'y ait qu'un seul message physique avec cet identificateur de groupe si le message logique n'appartient pas à un groupe de messages. Les messages logiques pour lesquels la segmentation a été inhibée par l'application émettrice ont un identificateur de groupe nul (MQGI_NONE), sauf si le message logique appartient à un groupe de messages.

Groupe de messages
Un groupe de messages est un ensemble d'un ou de plusieurs messages logiques ayant le même identificateur de groupe non nul. Les messages logiques du groupe se distinguent par des valeurs différentes pour le numéro de séquence de message, qui est un entier compris entre 1 et *n*, où *n* est le nombre de messages logiques du groupe. Si un ou plusieurs messages logiques sont segmentés, il y a plus de *n* messages physiques dans le groupe.

MQPMO_LOGICAL_ORDER
Cette option indique au gestionnaire de files d'attente comment l'application insère des messages dans des groupes et des segments de messages logiques. Elle ne peut être spécifiée que dans l'appel MQPUT ; elle n'est pas valide dans l'appel MQPUT1 .

Si MQPMO_LOGICAL_ORDER est spécifié, il indique que l'application utilise des appels MQPUT successifs pour :

1. Placer les segments dans chaque message logique dans l'ordre croissant des décalages de segment, à partir de 0, sans écart.
2. Placer tous les segments dans un message logique avant de les placer dans le message logique suivant.
3. Placer les messages logiques dans chaque groupe de messages dans l'ordre croissant des numéros de séquence de message, à partir de 1, sans écart.IBM WebSphere MQ incrémente automatiquement le numéro de séquence de message.
4. Placer tous les messages logiques dans un groupe de messages avant de les placer dans le groupe de messages suivant.

Pour plus d'informations sur MQPMO_LOGICAL_ORDER, voir Ordre logique et physique

Options de contexte. Les options suivantes contrôlent le traitement du contexte de message:

MQPMO_NO_CONTEXT
Le contexte d'identité et le contexte d'origine sont définis pour indiquer l'absence de contexte. Cela signifie que les zones de contexte dans MQMD sont définies sur:

- Blancs pour les zones de caractères
- Valeurs nulles pour les zones d'octet
- Zéros pour les champs numériques

MQPMO_CONTEXTE_PAR_DÉFAUT
Le message doit être associé à des informations de contexte par défaut, à la fois pour l'identité et l'origine. Le gestionnaire de files d'attente définit les zones de contexte dans le descripteur de message comme suit:

Zone dans MQMD	Valeur utilisée
UserIdentifier	Déterminé à partir de l'environnement si possible ; sinon, il est mis à blanc.
AccountingToken	Déterminé à partir de l'environnement si possible ; défini sur MQACT_NONE dans le cas contraire.
ApplIdentityData	Mettez à blanc.
PutApplType	Déterminé à partir de l'environnement.
PutApplName	Déterminé à partir de l'environnement si possible ; sinon, il est mis à blanc.
PutDate	Définissez la date à laquelle le message est inséré.
PutTime	Défini sur l'heure à laquelle le message est inséré.
ApplOriginData	Mettez à blanc.

Pour plus d'informations sur le contexte de message, voir Contexte de message.

Il s'agit des valeurs par défaut et des actions si aucune option de contexte n'est spécifiée.

MQPMO_PASS_IDENTITY_CONTEXT
Le message doit être associé à des informations de contexte. Le contexte d'identité est extrait de l'identificateur de file d'attente spécifié dans la zone *Context* . Les informations de contexte d'origine sont générées par le gestionnaire de files d'attente de la même manière que pour MQPMO_DEFAULT_CONTEXT (voir le tableau précédent pour les valeurs). Pour plus d'informations sur le contexte de message, voir Contexte de message.

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option MQOO_PASS_IDENTITY_CONTEXT (ou une option qui l'implique). Pour l'appel MQPUT1 , la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option MQOO_PASS_IDENTITY_CONTEXT.

MQPMO_PASS_ALL_CONTEXT
Le message doit être associé à des informations de contexte. Le contexte est extrait de l'identificateur de file d'attente spécifié dans la zone *Context* . Pour plus d'informations sur le contexte de message, voir Contrôle des informations de contexte.

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option MQOO_PASS_ALL_CONTEXT (ou une option qui l'implique). Pour l'appel MQPUT1 , la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option MQOO_PASS_ALL_CONTEXT.

MQPMO_SET_IDENTITY_CONTEXT
Le message doit être associé à des informations de contexte. L'application spécifie le contexte d'identité dans la structure MQMD. Les informations de contexte d'origine sont générées par le gestionnaire de files d'attente de la même manière que pour MQPMO_DEFAULT_CONTEXT (voir le tableau précédent pour les valeurs). Pour plus d'informations sur le contexte de message, voir Contexte de message.

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option MQOO_SET_IDENTITY_CONTEXT (ou une option qui l'implique). Pour l'appel MQPUT1 , la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option MQOO_SET_IDENTITY_CONTEXT.

MQPMO_SET_ALL_CONTEXT
Le message doit être associé à des informations de contexte. L'application spécifie l'identité, l'origine et le contexte utilisateur dans la structure MQMD. Pour plus d'informations sur le contexte de message, voir Contexte de message.

Pour l'appel MQPUT, la file d'attente doit avoir été ouverte avec l'option MQOO_SET_ALL_CONTEXT. Pour l'appel MQPUT1 , la même vérification d'autorisation est effectuée que pour l'appel MQOPEN avec l'option MQOO_SET_ALL_CONTEXT.

Vous ne pouvez spécifier qu'une seule des options de contexte MQPMO_ * _CONTEXT. Si vous n'en spécifiez aucun, MQPMO_DEFAULT_CONTEXT est pris en compte.

Options de propriété. L'option suivante concerne les propriétés du message:

MQPMO_MD_FOR_OUTPUT_ONLY
Le paramètre de descripteur de message ne doit être utilisé que pour la sortie afin de renvoyer le descripteur de message du message qui a été inséré. Les zones de descripteur de message associées aux zones *NewMsgHandle*, *OriginalMsgHandle*, ou aux deux, de la structure **MQPMO** doivent être utilisées pour l'entrée.

Si aucun descripteur de message valide n'est fourni, l'appel échoue avec le code anomalie **MQRC_MD_ERROR**.

Options de réponse d'insertion. Les options suivantes contrôlent la réponse renvoyée à un appel MQPUT ou MQPUT1 . Vous ne pouvez spécifier qu'une seule de ces options. Si MQPMO_ASYNC_RESPONSE et MQPMO_SYNC_RESPONSE ne sont pas spécifiés, MQPMO_RESPONSE_AS_Q_DEF ou MQPMO_RESPONSE_AS_TOPIC_DEF est utilisé.

MQPMO_ASYNC_RESPONSE

L'option MQPMO_ASYNC_RESPONSE demande qu'une opération MQPUT ou MQPUT1 soit terminée sans que l'application attende que le gestionnaire de files d'attente termine l'appel. L'utilisation de cette option permet d'améliorer les performances de la messagerie, en particulier pour les applications utilisant des liaisons client. Une application peut vérifier périodiquement, à l'aide de l'instruction MQSTAT, si une erreur s'est produite lors des appels asynchrones précédents.

Avec cette option, seules les zones suivantes sont garanties pour être renseignées dans le MQMD ;

- ApplIdentityData
- PutApplType
- PutApplName
- ApplOriginData

En outre, si l'un des MQPMO_NEW_MSG_ID ou MQPMO_NEW_CORREL_ID ou les deux sont spécifiés en tant qu'options, les valeurs MsgId et CorrelId renvoyées sont également renseignées. (MQPMO_NEW_MSG_ID peut être implicitement spécifié en spécifiant une zone MsgId vide).

Seules les zones indiquées ci-dessus sont renseignées. D'autres informations normalement renvoyées dans la structure MQMD ou MQPMO ne sont pas définies.

Lors de la demande de réponse d'insertion asynchrone pour MQPUT1, les noms ResolvedQName et ResolvedQMgrrenvoyés dans la structure MQOD ne sont pas définis.

Lorsque vous demandez une réponse d'insertion asynchrone pour MQPUT ou MQPUT1, un CompCode et une raison de MQCC_OK et MQRC_NONE ne signifient pas nécessairement que le message a été correctement inséré dans une file d'attente. Lors du développement d'une application MQI qui utilise une réponse d'insertion asynchrone et qui requiert la confirmation que des messages ont été insérés dans une file d'attente, vous devez vérifier à la fois les codes CompCode et les codes anomalie des opérations d'insertion et utiliser MQSTAT pour obtenir des informations sur les erreurs asynchrones.

Bien que la réussite ou l'échec de chaque appel MQPUT ou MQPUT1 individuel ne soit pas immédiatement renvoyé, la première erreur qui s'est produite lors d'un appel asynchrone peut être déterminée ultérieurement via un appel à MQSTAT.

Si un message persistant sous le point de synchronisation ne parvient pas à être distribué à l'aide d'une réponse d'insertion asynchrone et que vous tentez de valider la transaction, la validation échoue et la transaction est annulée avec le code achèvement MQCC_FAILED et le motif MQRC_BACKED_OUT. L'application peut appeler MQSTAT pour déterminer la cause d'un échec MQPUT ou MQPUT1 précédent.

MQPMO_SYNC_REPONSE

La spécification de ce type de réponse d'insertion garantit que l'opération MQPUT ou MQPUT1 est toujours émise de manière synchrone. Si l'opération d'insertion aboutit, toutes les zones de MQMD et MQPMO sont terminées.

Cette option garantit une réponse synchrone quelle que soit la valeur de réponse d'insertion par défaut définie dans la file d'attente ou l'objet de rubrique.

MQPMO_REPONSE_AS_Q_DEF

Si cette valeur est spécifiée pour un appel MQPUT, le type de réponse d'insertion utilisé est pris de la valeur DEFPRESP spécifiée dans la file d'attente lors de sa première ouverture par l'application. Si une application client est connectée à un gestionnaire de files d'attente à un niveau antérieur à la version 7.0, elle se comporte comme si MQPMO_SYNC_RESPONSE était spécifié.

Si cette option est spécifiée pour un appel MQPUT1 , la valeur de l'attribut DEFPRESP n'est pas connue avant l'envoi de la demande au serveur. Par défaut, si l'appel MQPUT1 utilise MQPMO_SYNCPOINT, il se comporte comme pour MQPMO_ASYNC_RESPONSE, et s'il utilise MQPMO_NO_SYNCPOINT, il se comporte comme pour MQPMO_SYNC_RESPONSE. Toutefois, vous pouvez remplacer ce comportement par défaut en définissant la propriété Put1DefaultAlwaysSync dans le fichier de configuration du client. Voir Strophe CHANNELS du fichier de configuration du client.

MQPMO_REPONSE_AS_TOPIC_DEF

MQPMO_RESPONSE_AS_TOPIC_DEF est un synonyme de MQPMO_RESPONSE_AS_Q_DEF pour une utilisation avec des objets de rubrique.

Autres options. Les options suivantes contrôlent la vérification des autorisations, ce qui se produit lorsque le gestionnaire de files d'attente est mis au repos et la résolution des noms de file d'attente et de gestionnaire de files d'attente:

MQPMO_ALTERNATE_USER_AUTHORITY
MQPMO_ALTERNATE_USER_AUTHORITY indique que la zone *AlternateUserId* du paramètre *ObjDesc* de l'appel MQPUT1 contient un identificateur utilisateur qui doit être utilisé pour valider les droits d'accès permettant d'insérer des messages dans la file d'attente. L'appel ne peut aboutir que si *AlternateUserId* est autorisé à ouvrir la file d'attente avec les options spécifiées, que l'ID utilisateur sous lequel l'application s'exécute soit ou non autorisé à le faire. (Cela ne s'applique pas aux options de contexte spécifiées, qui sont toujours vérifiées par rapport à l'identificateur utilisateur sous lequel l'application s'exécute.)

Cette option est valide uniquement avec l'appel MQPUT1 .

MQPMO_FAIL_IF_QUIESCING
Cette option force l'échec de l'appel MQPUT ou MQPUT1 si le gestionnaire de files d'attente est à l'état de mise au repos.

Sous z/OS, cette option force également l'appel MQPUT ou MQPUT1 à échouer si la connexion (pour une application CICS ou IMS) est à l'état de mise au repos.

L'appel renvoie le code achèvement MQCC_FAILED avec le code anomalie MQRC_Q_MGR_QUIESCING ou MQRC_CONNECTION_QUIESCING.

MQPMO_RESOLVE_LOCAL_Q
Cette option permet de renseigner *ResolvedQName* dans la structure MQPMO avec le nom de la file d'attente locale dans laquelle le message est inséré et *ResolvedQMgrName* avec le nom du gestionnaire de files d'attente local qui héberge la file d'attente locale. Pour plus d'informations sur MQPMO_RESOLVE_LOCAL_Q, voir la rubrique MQOO_RESOLVE_LOCAL_Q.

Si vous êtes autorisé à placer dans une file d'attente, vous disposez des droits requis pour spécifier cet indicateur dans l'appel MQPUT ; aucun droit spécial n'est requis.

Option par défaut. Si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

MQPMO_NONE
Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut. MQPMO_NONE est défini pour faciliter la documentation du programme ; il n'est pas prévu que cette option soit utilisée avec d'autres, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

MQPMO_NONE est une zone d'entrée. La valeur initiale de la zone *Options* est MQPMO_NONE.

Descripteur OriginalMsg(MQHMSG)

Il s'agit d'un descripteur facultatif d'un message. Il se peut qu'il ait été précédemment extrait d'une file d'attente. L'utilisation de ce descripteur est soumise à la valeur de la zone *Action* ; voir aussi NewMsgHandle.

Le contenu du descripteur de message d'origine ne sera pas modifié par l'appel **MQPUT** ou **MQPUT1** .

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est **MQHM_NONE**. Cette zone est ignorée si la version est inférieure à **MQPMO_VERSION_3**.

PubLevel (MQLONG)
La valeur initiale de cette zone est 9. Niveau d'abonnement ciblé par cette publication. Seuls les abonnements dont le SubLevel le plus élevé est inférieur ou égal à cette valeur reçoivent cette publication. Cette valeur doit être comprise entre zéro et 9 ; zéro est le niveau le plus bas. Toutefois, si une publication a été conservée, elle n'est plus disponible pour les abonnés de niveau supérieur car elle est republiée sur le site PubLevel 1.

Pour plus d'informations, voir Intercepting publications .

PutMsgRecFields (MQLONG)

Cette zone contient des indicateurs qui indiquent les zones MQPMR présentes dans les enregistrements de message d'insertion fournis par l'application. Utilisez *PutMsgRecFields* uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* est égal à zéro, ou si les deux valeurs *PutMsgRecOffset* et *PutMsgRecPtr* sont égales à zéro.

Pour les zones qui sont présentes, le gestionnaire de files d'attente utilise pour chaque destination les valeurs des zones de l'enregistrement de message d'insertion correspondant. Pour les zones absentes, le gestionnaire de files d'attente utilise les valeurs de la structure MQMD.

Utilisez un ou plusieurs des indicateurs suivants pour indiquer les zones qui sont présentes dans les enregistrements de message d'insertion:

ID MQPMRF_MSG_ID
La zone d'identificateur de message est présente.

ID_CORREL_MQPMRF_
La zone d'identificateur de corrélation est présente.

ID_GROUPE_MQPMRF_ID
La zone d'identificateur de groupe est présente.

MQPMRF_FEEDBACK
Un champ de commentaires est présent.

MQPMRF_COMPTING_TOKEN
La zone de jeton de comptabilité est présente.

Si vous spécifiez cet indicateur, indiquez MQPMO_SET_IDENTITY_CONTEXT ou MQPMO_SET_ALL_CONTEXT dans la zone *Options* ; si cette condition n'est pas satisfaite, l'appel échoue avec le code anomalie MQRC_PMO_RECORD_FLAGS_ERROR.

Si aucune zone MQPMR n'est présente, les éléments suivants peuvent être spécifiés:

MQPMRF_AUCUN
Aucune zone d'enregistrement d'insertion de message n'est présente.

Si cette valeur est spécifiée, soit *RecsPresent* doit être égal à zéro, soit les deux valeurs *PutMsgRecOffset* et *PutMsgRecPtr* doivent être égales à zéro.

MQPMRF_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette constante soit utilisée avec d'autres, mais comme sa valeur est nulle, une telle utilisation ne peut pas être détectée.

Si *PutMsgRecFields* contient des indicateurs non valides ou que des enregistrements de message d'insertion sont fournis mais que *PutMsgRecFields* a la valeur MQPMRF_NONE, l'appel échoue avec le code anomalie MQRC_PMO_RECORD_FLAGS_ERROR.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est MQPMRF_NONE. Cette zone est ignorée si *Version* est inférieur à MQPMO_VERSION_2.

PutMsgRecOffset (MQLONG)

Décalage en octets du premier enregistrement de message d'insertion MQPMR à partir du début de la structure MQPMO. Le décalage peut être positif ou négatif. *PutMsgRecOffset* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* a la valeur zéro.

Lorsque le message est inséré dans une liste de distribution, un tableau d'un ou de plusieurs enregistrements de message d'insertion MQPMR peut être fourni afin de spécifier certaines propriétés du message pour chaque destination individuellement ; ces propriétés sont les suivantes:

- Identificateur de message
- Identificateur de corrélation
- Identificateur de groupe
- Valeur de retour d'informations

- Jeton de comptabilité

Vous n'avez pas besoin de spécifier toutes ces propriétés, mais quel que soit le sous-ensemble que vous choisissez, spécifiez les zones dans le bon ordre. Pour plus de détails, voir la description de la structure MQPMR.

Généralement, il doit y avoir autant d'enregistrements de message d'insertion que d'enregistrements d'objet spécifiés par MQOD lors de l'ouverture de la liste de distribution ; chaque enregistrement de message d'insertion fournit les propriétés de message pour la file d'attente identifiée par l'enregistrement d'objet correspondant. Les files d'attente de la liste de distribution dont l'ouverture a échoué doivent toujours avoir des enregistrements de message alloués aux emplacements appropriés dans le tableau, bien que les propriétés de message soient ignorées dans ce cas.

Le nombre d'enregistrements de message d'insertion peut être différent du nombre d'enregistrements d'objet. S'il y a moins d'enregistrements de message d'insertion que d'enregistrements d'objet, les propriétés de message pour les destinations qui n'ont pas d'enregistrements de message d'insertion sont extraites des zones correspondantes dans le descripteur de message MQMD. S'il y a plus d'enregistrements de message d'insertion que d'enregistrements d'objet, l'excès n'est pas utilisé (bien qu'il soit toujours possible d'y accéder). Les enregistrements de message d'insertion sont facultatifs, mais s'ils sont fournis, ils doivent être *RecsPresent* .

Fournissez les enregistrements de message d'insertion de la même manière que les enregistrements d'objet dans MQOD, soit en spécifiant un décalage dans *PutMsgRecOffset*, soit en spécifiant une adresse dans *PutMsgRecPtr*; pour plus de détails sur cette opération, voir la zone *ObjectRecOffset* décrite dans «MQOD-Descripteur d'objet», à la page 456.

Vous ne pouvez pas utiliser plus d'un *PutMsgRecOffset* ou *PutMsgRecPtr* ; l'appel échoue avec le code anomalie MQRC_PUT_MSG_RECORDS_ERROR si les deux sont différents de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQPMO_VERSION_2.

PutMsgRecPtr (MQPTR)

Il s'agit de l'adresse du premier enregistrement de message d'insertion MQPMR. Utilisez *PutMsgRecPtr* uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* a la valeur zéro.

Vous pouvez utiliser *PutMsgRecPtr* ou *PutMsgRecOffset* pour spécifier les enregistrements de message d'insertion, mais pas les deux. Pour plus de détails, voir la description de la zone *PutMsgRecOffset* ci-dessus. Si vous n'utilisez pas *PutMsgRecPtr*, définissez-le sur le pointeur null ou sur les octets null.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQPMO_VERSION_2.

Remarque : Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

RecsPresent (MQLONG)

Il s'agit du nombre d'enregistrements de message d'insertion MQPMR ou d'enregistrements de réponse MQRR fournis par l'application. Ce nombre peut être supérieur à zéro uniquement si le message est inséré dans une liste de distribution. Les enregistrements de message d'insertion et les enregistrements de réponse sont facultatifs ; l'application n'a pas besoin de fournir d'enregistrements ou elle peut choisir de fournir des enregistrements d'un seul type. Toutefois, si l'application fournit des enregistrements des deux types, elle doit fournir des enregistrements *RecsPresent* de chaque type.

Il n'est pas nécessaire que la valeur de *RecsPresent* soit identique au nombre de destinations dans la liste de distribution. Si un trop grand nombre d'enregistrements est fourni, l'excès n'est pas utilisé ; si le nombre d'enregistrements est trop faible, les valeurs par défaut sont utilisées pour les propriétés de message des destinations qui n'ont pas d'enregistrements de message insérés (voir *PutMsgRecOffset*).

Si *RecsPresent* est inférieur à zéro ou supérieur à zéro mais que le message n'est pas inséré dans une liste de distribution, l'appel échoue avec le code anomalie MQRC_RECS_Présentat_ERROR.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQPMO_VERSION_2.

ResolvedQMgr(MQCHAR48)

Il s'agit du nom du gestionnaire de files d'attente de destination une fois la résolution de nom effectuée par le gestionnaire de files d'attente local. Le nom renvoyé correspond au nom du gestionnaire de files d'attente qui possède la file d'attente identifiée par *ResolvedQName*et peut correspondre au nom du gestionnaire de files d'attente local.

Si *ResolvedQName* est une file d'attente partagée appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local, *ResolvedQMgrName* est le nom du groupe de partage de files d'attente. Si la file d'attente appartient à un autre groupe de partage de files d'attente, *ResolvedQName* peut être le nom du groupe de partage de files d'attente ou le nom d'un gestionnaire de files d'attente membre du groupe de partage de files d'attente (la nature de la valeur renvoyée est déterminée par les définitions de files d'attente qui existent sur le gestionnaire de files d'attente local).

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ; si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

ResolvedQName (MQCHAR48)

Il s'agit du nom de la file d'attente de destination une fois la résolution de nom effectuée par le gestionnaire de files d'attente local. Le nom renvoyé est le nom d'une file d'attente qui existe sur le gestionnaire de files d'attente identifié par *ResolvedQMgrName*.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique ; si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

Il s'agit d'une zone de sortie. La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

ResponseRec(MQLONG)

Décalage en octets du premier enregistrement de réponse MQRR à partir du début de la structure MQPMO. Le décalage peut être positif ou négatif. *ResponseRecOffset* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* a la valeur zéro.

Lors de l'insertion du message dans une liste de distribution, vous pouvez fournir un tableau d'un ou de plusieurs enregistrements de réponse MQRR afin d'identifier les files d'attente auxquelles le message n'a pas été envoyé avec succès (zone*CompCode* dans MQRR), ainsi que la raison de chaque échec (zone*Reason* dans MQRR). Il se peut que le message n'ait pas été envoyé en raison de l'échec de l'ouverture de la file d'attente ou de l'échec de l'opération d'insertion. Le gestionnaire de files d'attente définit les enregistrements de réponse uniquement lorsque le résultat de l'appel est mixte (c'est-à-dire que certains messages ont été envoyés avec succès alors que d'autres ont échoué, ou que tous ont échoué, mais pour des raisons différentes) ; le code anomalie MQRC_MULTIPLE_MOTIFS de l'appel indique ce cas. Si le même code anomalie s'applique à toutes les files d'attente, ce code anomalie est renvoyé dans le paramètre *Reason* de l'appel MQPUT ou MQPUT1 et les enregistrements de réponse ne sont pas définis.

En général, il existe autant d'enregistrements de réponse que d'enregistrements d'objet spécifiés par MQOD lors de l'ouverture de la liste de distribution ; si nécessaire, chaque enregistrement de réponse est défini sur le code achèvement et le code raison de l'insertion dans la file d'attente identifiée par l'enregistrement d'objet correspondant. Les files d'attente de la liste de distribution dont l'ouverture a échoué doivent toujours avoir des enregistrements de réponse alloués aux emplacements appropriés de

la grappe, bien qu'elles soient définies sur le code achèvement et le code raison résultant de l'opération d'ouverture, plutôt que sur l'opération d'insertion.

Le nombre d'enregistrements de réponse peut différer du nombre d'enregistrements d'objet. S'il y a moins d'enregistrements de réponse que d'enregistrements d'objet, l'application peut ne pas être en mesure d'identifier toutes les destinations pour lesquelles l'opération d'insertion a échoué ou les raisons des échecs. S'il y a plus d'enregistrements de réponse que d'enregistrements d'objet, l'excédent n'est pas utilisé (bien qu'il soit toujours possible d'y accéder). Les enregistrements de réponse sont facultatifs, mais s'ils sont fournis, ils doivent être *RecsPresent* .

Fournissez les enregistrements de réponse de la même manière que les enregistrements d'objet dans MQOD, soit en spécifiant un décalage dans *ResponseRecOffset*, soit en spécifiant une adresse dans *ResponseRecPtr*; pour plus de détails sur cette opération, voir la zone *ObjectRecOffset* décrite dans «MQOD-Descripteur d'objet», à la page 456. Toutefois, n'utilisez pas plus de *ResponseRecOffset* et *ResponseRecPtr*; l'appel échoue avec le code anomalie MQRC_RESPONSE_RECORDS_ERROR si les deux valeurs sont différentes de zéro.

Pour l'appel MQPUT1 , cette zone doit être à zéro. En effet, les informations de réponse (si elles sont demandées) sont renvoyées dans les enregistrements de réponse spécifiés par le descripteur d'objet MQOD.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0. Cette zone est ignorée si *Version* est inférieur à MQPMO_VERSION_2.

ResponseRecPtr (MQPTR)

Il s'agit de l'adresse du premier enregistrement de réponse MQRR. *ResponseRecPtr* est utilisé uniquement lorsque le message est inséré dans une liste de distribution. La zone est ignorée si *RecsPresent* a la valeur zéro.

Utilisez *ResponseRecPtr* ou *ResponseRecOffset* pour spécifier les enregistrements de réponse, mais pas les deux ; voir la description de la zone *ResponseRecOffset* ci-dessus pour plus de détails. Si vous n'utilisez pas *ResponseRecPtr* , définissez-le sur le pointeur null ou sur les octets null.

Pour l'appel MQPUT1 , cette zone doit être le pointeur null ou des octets null. En effet, les informations de réponse (si elles sont demandées) sont renvoyées dans les enregistrements de réponse spécifiés par le descripteur d'objet MQOD.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire. Cette zone est ignorée si *Version* est inférieur à MQPMO_VERSION_2.

Remarque : Sur les plateformes où le langage de programmation ne prend pas en charge le type de données pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée, la valeur initiale étant la chaîne d'octets tout-null.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID_STRUC_MQPMO_
Identificateur de la structure des options d'insertion de message.

Pour le langage de programmation C, la constante MQPMO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQPMO_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQPMO_STRUC_ID.

Délai d'attente (MQLONG)

Il s'agit d'une zone réservée ; sa valeur n'est pas significative. La valeur initiale de cette zone est -1.

Nombre UnknownDest(MQLONG)

Il s'agit du nombre de messages que l'appel MQPUT ou MQPUT1 en cours a correctement envoyés aux files d'attente de la liste de distribution qui sont résolues en files d'attente éloignées. Messages que le gestionnaire de files d'attente conserve temporairement sous forme de liste de distribution, en tant que nombre de destinations individuelles que ces listes de distribution contiennent. Cette zone est également définie lors de l'insertion d'un message dans une file d'attente unique qui ne figure pas dans une liste de distribution.

Il s'agit d'une zone de sortie. La valeur initiale de cette zone est 0. Cette zone n'est pas définie si *Version* est inférieur à MQPMO_VERSION_1.

Cette zone n'est pas définie sur z/OS car les listes de distribution ne sont pas prises en charge.

Version (MQLONG)
Numéro de version de la structure.

Il doit s'agir de l'une des valeurs suivantes :

MQPMO_VERSION_1
Structure des options d'insertion de message Version-1 .

Cette version est prise en charge dans tous les environnements.

MQPMO_VERSION_2
Structure des options d'insertion de message Version-2 .

Cette version est prise en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus les clients WebSphere MQ MQI connectés à ces systèmes.

MQPMO_VERSION_3
Structure des options d'insertion de message Version-3 .

Cette version est prise en charge dans tous les environnements.

Les zones qui existent uniquement dans la version la plus récente de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

MQPMO_CURRENT_VERSION
Version actuelle de la structure des options d'insertion de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQPMO_VERSION_1.

Valeurs initiales et déclarations de langue pour MQPMO

Tableau 529. Valeurs initiales des zones dans MQPMO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQPMO_	'PMO¬'
Version	MQPMO_VERSION_1	1
Options	MQPMO_NONE	0
Timeout	Aucun	-1
Context	Aucun	0
KnownDestCount	Aucun	0
UnknownDestCount	Aucun	0
InvalidDestCount	Aucun	0
ResolvedQName	Aucun	Chaîne nulle ou blancs
ResolvedQMgrName	Aucun	Chaîne nulle ou blancs

Tableau 529. Valeurs initiales des zones dans MQPMO (suite)

Nom de zone	Nom de la constante	Valeur de la constante
RecsPresent	Aucun	0
PutMsgRecFields	MQPMRF_AUCUN	0
PutMsgRecOffset	Aucun	0
ResponseRecOffset	Aucun	0
PutMsgRecPtr	Aucun	Pointeur null ou octets null
ResponseRecPtr	Aucun	Pointeur null ou octets null
OriginalMsgHandle	MQHM_NONE	0
NewMsgHandle	MQHM_NONE	0
Action	MQACTP_NOUVEAU	0
PubLevel	Aucun	9

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQPMO_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQPMO MyPMO = {MQPMO_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Enregistrement de message d'insertion

Le tableau suivant récapitule les zones de la structure.

Tableau 530. Zones dans MQPMR		
Zone	**Description**	**Topic**
MsgId	Identificateur de message	MsgId
CorrelId	Identificateur de corrélation	CorrelId
GroupId	Identificateur de groupe	GroupId
Feedback	Commentaires en retour ou code anomalie	Commentaires
AccountingToken	Jeton de comptabilité	AccountingToken

Présentation de MQPMR

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ connectés à ces systèmes.

Objectif: Utilisez la structure MQPMR pour spécifier diverses propriétés de message pour une destination unique lors de l'insertion d'un message dans une liste de distribution. MQPMR est une structure d'entrée/sortie pour les appels MQPUT et MQPUT1 .

Jeu de caractères et codage: les données dans MQPMR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ , la structure doit être dans le jeu de caractères et le codage du client.

Utilisation: en fournissant un tableau de ces structures sur l'appel MQPUT ou MQPUT1 , vous pouvez spécifier des valeurs différentes pour chaque file d'attente de destination dans une liste de distribution. Certaines des zones sont des entrées uniquement, d'autres sont des entrées / sorties.

Remarque : Cette structure est inhabituelle dans la mesure où elle n'a pas de présentation fixe. Les zones de cette structure sont facultatives et la présence ou l'absence de chaque zone est indiquée par les indicateurs de la zone *PutMsgRecFields* dans MQPMO. Les zones qui sont présentes ***doivent apparaître dans l'ordre suivant***:

- *MsgId*
- *CorrelId*
- *GroupId*
- *Feedback*
- *AccountingToken*

Les zones absentes n'occupent aucun espace dans l'enregistrement.

Comme MQPMR n'a pas de présentation fixe, aucune définition de celle-ci n'est fournie dans les fichiers d'en-tête, COPY et INCLUDE pour les langages de programmation pris en charge. Le programmeur d'application doit créer une déclaration contenant les zones requises par l'application et définir les indicateurs dans *PutMsgRecFields* pour indiquer les zones présentes.

Zones pour MQPMR

La structure MQPMR contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

AccountingToken (MQBYTE32)

Il s'agit du jeton de comptabilité à utiliser pour le message envoyé à la file d'attente avec le nom indiqué par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *AccountingToken* dans MQMD pour une insertion dans une file d'attente unique. Voir la description de *AccountingToken* dans «MQMD-Descripteur de message», à la page 394 pour plus d'informations sur le contenu de cette zone.

Si cette zone n'est pas présente, la valeur de MQMD est utilisée.

Il s'agit d'une zone d'entrée.

CorrelId (MQBYTE24)

Il s'agit de l'identificateur de corrélation à utiliser pour le message envoyé à la file d'attente avec un nom qui a été spécifié par l'élément correspondant dans le tableau de structures MQOR fourni sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *CorrelId* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *CorrelId* .

Si MQPMO_NEW_CORREL_ID est spécifié, un nouvel identificateur de corrélation *unique* est généré et utilisé pour toutes les destinations de la liste de distribution, qu'elles comportent ou non des enregistrements MQPMR. Cette méthode est différente de celle utilisée pour le traitement de MQPMO_NEW_MSG_ID (voir la zone *MsgId*).

Il s'agit d'une zone d'entrée/sortie.

Commentaires en retour (MQLONG)

Il s'agit du code retour à utiliser pour le message envoyé à la file d'attente avec le nom qui a été spécifié par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *Feedback* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente, la valeur de MQMD est utilisée.

Il s'agit d'une zone d'entrée.

GroupId (MQBYTE24)

GroupId est l'identificateur de groupe à utiliser pour le message envoyé à la file d'attente avec le nom spécifié par l'élément correspondant dans le tableau de structures MQOR fourni dans l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *GroupId* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *GroupId* . La valeur est traitée comme indiqué dans Ordre physique dans une file d'attente, mais avec les différences suivantes:

- GroupId est créé à partir du nom de gestionnaire de files d'attente et d'un horodatage. Par conséquent, pour conserver un ID de groupe (GroupId) unique, les noms de gestionnaire de files d'attente doivent également être uniques. Ne redéfinissez pas non plus les horloges sur la machine des gestionnaires de files d'attente.
- Dans les cas où un nouvel identificateur de groupe serait utilisé, le gestionnaire de files d'attente génère un identificateur de groupe différent pour chaque destination (c'est-à-dire que deux destinations n'ont pas le même identificateur de groupe).
- Dans les cas où la valeur de la zone serait utilisée, l'appel échoue avec le code anomalie MQRC_GROUP_ID_ERROR

Il s'agit d'une zone d'entrée/sortie.

MsgId (MQBYTE24)

Il s'agit de l'identificateur de message à utiliser pour le message envoyé à la file d'attente avec un nom qui a été spécifié par l'élément correspondant dans le tableau de structures MQOR fourni sur l'appel MQOPEN ou MQPUT1 . Il est traité de la même manière que la zone *MsgId* dans MQMD pour une insertion dans une file d'attente unique.

Si cette zone n'est pas présente dans l'enregistrement MQPMR ou s'il y a moins d'enregistrements MQPMR que de destinations, la valeur de MQMD est utilisée pour les destinations qui n'ont pas d'enregistrement MQPMR contenant une zone *MsgId* . Si cette valeur est MQMI_NONE, un nouvel identificateur de message est généré pour *chacune* de ces destinations (c'est-à-dire que deux de ces destinations n'ont pas le même identificateur de message).

Si MQPMO_NEW_MSG_ID est spécifié, de nouveaux identificateurs de message sont générés pour toutes les destinations de la liste de distribution, qu'elles comportent ou non des enregistrements MQPMR. Cette méthode est différente de celle utilisée pour le traitement de MQPMO_NEW_CORREL_ID (voir la zone *CorrelId*).

Il s'agit d'une zone d'entrée/sortie.

Valeurs initiales et déclarations de langage pour MQPMR

Aucune valeur initiale n'est définie pour cette structure, car aucune déclaration de structure n'est fournie dans les fichiers d'en-tête, COPY et INCLUDE pour les langages de programmation pris en charge. Les exemples de déclaration montrent comment déclarer la structure si toutes les zones sont requises.

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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).
```

Déclaration PL/I

```
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 */
```

Déclaration Visual Basic

```
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-Règles et en-tête de formatage

Cette section décrit les règles et l'en-tête de formatage, les zones qu'il contient et les valeurs initiales de ces zones.

Présentation de MQRFH

Disponibilité: Tous les systèmes WebSphere MQ , plus WebSphere MQ clients MQI connectés à ces systèmes.

Objectif: La structure MQRFH définit la présentation des règles et l'en-tête de formatage. Utilisez cet en-tête pour envoyer des données de chaîne sous la forme de paires nom / valeur.

Nom de format: MQFMT_RF_HEADER.

Jeu de caractères et codage: les zones de la structure MQRFH (y compris *NameValueString*) se trouvent dans le jeu de caractères et le codage donnés par les zones *CodedCharSetId* et *Encoding* de la structure d'en-tête qui précède MQRFH, ou par ces zones de la structure MQMD si MQRFH se trouve au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

Champs pour MQRFH

La structure MQRFH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

CodedCharSetId (MQLONG)

Indique l'identificateur de jeu de caractères des données qui suivent *NameValueString*; il ne s'applique pas aux données de type caractères de la structure MQRFH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

MQCCSI_HÉRITER
Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI_INHERIT n'est pas renvoyée par l'appel MQGET.

MQCCSI_INHERIT ne peut pas être utilisé si la valeur de la zone *PutApplType* dans MQMD est MQAT_BROKER.

La valeur initiale de cette zone est MQCCSI_UNDEFINED.

Codage (MQLONG)

Indique le codage numérique des données qui suit *NameValueString*; il ne s'applique pas aux données numériques de la structure MQRFH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est MQENC_NATIVE.

Indicateurs (MQLONG)

Les éléments suivants peuvent être spécifiés:

MQRFH_AUCUN
Aucun indicateur.

La valeur initiale de cette zone est MQRFH_NONE.

Format (MQCHAR8)

Indique le nom de format des données qui suivent *NameValueString*.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La valeur initiale de cette zone est MQFMT_NONE.

Chaîne NameValue(MQCHARn)

Il s'agit d'une chaîne de caractères de longueur variable contenant des paires nom / valeur au format suivant:

```
name1 value1 name2 value2 name3 value3 ...
```

Chaque nom ou valeur doit être séparé du nom ou de la valeur adjacente par un ou plusieurs caractères blancs ; ces blancs ne sont pas significatifs. Un nom ou une valeur peut contenir des blancs significatifs en ajoutant en préfixe et en suffixe le nom ou la valeur avec des guillemets ; tous les caractères entre les guillemets doubles ouverts et les guillemets fermants correspondants sont considérés comme significatifs. Dans l'exemple suivant, le nom est FAMOUS_WORDSet la valeur est `Hello World`:

```
FAMOUS_WORDS "Hello World"
```

Un nom ou une valeur peut contenir des caractères autres que le caractère null (qui sert de délimiteur pour *NameValueString*). Toutefois, pour faciliter l'interopérabilité, une application peut limiter les noms aux caractères suivants:

- Premier caractère: majuscule ou minuscule (A à Z ou a à z) ou trait de soulignement.
- Caractères suivants: caractères alphabétiques majuscules ou minuscules, chiffres décimaux (0 à 9), trait de soulignement, trait d'union ou point.

Si un nom ou une valeur contient un ou plusieurs guillemets, le nom ou la valeur doit être placé entre guillemets et chaque guillemet dans la chaîne doit être doublé:

```
Famous_Words "The program displayed ""Hello World"""
```

Les noms et les valeurs sont sensibles à la casse, c'est-à-dire que les minuscules ne sont pas considérées comme des majuscules. Par exemple, FAMOUS_WORDS et `Famous_Words` sont deux noms différents.

La longueur en octets de *NameValueString* est égale à *StrucLength* moins MQRFH_STRUC_LENGTH_FIXED. Pour éviter les problèmes de conversion des données utilisateur dans certains environnements, faites de cette longueur un multiple de quatre. Placez *NameValueString* avec des blancs de cette longueur ou terminez-le plus tôt en plaçant un caractère nul après le dernier caractère significatif de la chaîne. Le caractère null et les octets qui le suivent, jusqu'à la longueur spécifiée de *NameValueString*, sont ignorés.

Remarque : Comme la longueur de cette zone n'est pas fixe, elle est omise dans les déclarations de la structure fournies pour les langages de programmation pris en charge.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID MQRFH_STRUC_ID
Identificateur des règles et de la structure d'en-tête de formatage.

Pour le langage de programmation C, la constante MQRFH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQRFH_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQRFH_STRUC_ID.

StrucLength (MQLONG)

Il s'agit de la longueur en octets de la structure MQRFH, y compris la zone *NameValueString* à la fin de la structure. La longueur n'inclut *pas* les données utilisateur qui suivent la zone *NameValueString* .

Pour éviter les problèmes de conversion des données utilisateur dans certains environnements, *StrucLength* doit être un multiple de quatre.

La constante suivante indique la longueur de la partie *fixe* de la structure, c'est-à-dire la longueur à l'exclusion de la zone *NameValueString* :

MQRFH_STRUC_LENGTH_FIXED
Longueur de la partie fixe de la structure MQRFH.

La valeur initiale de cette zone est MQRFH_STRUC_LENGTH_FIXED.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQRFH_VERSION_1
Règles Version-1 et structure d'en-tête de formatage.

La valeur initiale de cette zone est MQRFH_VERSION_1.

Valeurs initiales et déclarations de langue pour MQRFH

Tableau 531. Valeurs initiales des champs dans MQRFH pour MQRFH

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID MQRFH_STRUC_ID	'RFH¬'
Version	MQRFH_VERSION_1	1
StrucLength	MQRFH_STRUC_LENGTH_FIXED	32
Encoding	MQENC_NATIVE	Dépend de l'environnement
CodedCharSetId	MQCCSI_UNDEFINI	0
Format	MQFMT_AUCUN	Espaces vides
Flags	MQRFH_AUCUN	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQRFH_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQRFH MyRFH = {MQRFH_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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 - En-tête 2 de règles et de formatage

Cette section décrit les règles et le formatage de l'en-tête 2, les zones qu'il contient et les valeurs initiales de ces zones.

Présentation de MQRFH2

Disponibilité

Tous les systèmes WebSphere MQ , ainsi que les clients WebSphere MQ MQI connectés à ces systèmes.

Objet

L'en-tête MQRFH2 est basé sur l'en-tête MQRFH , mais il permet aux chaînes Unicode d'être transportées sans traduction, et il peut contenir des types de données numériques.

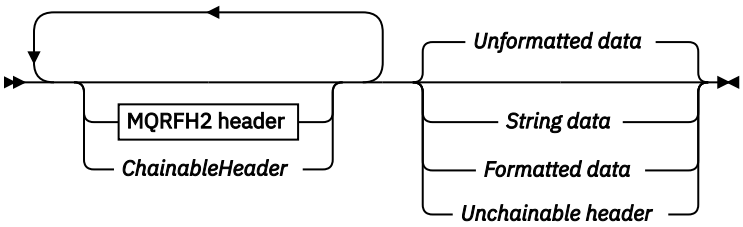
La structure MQRFH2 définit le format des règles version-2 et de l'en-tête de formatage. Utilisez cet en-tête pour envoyer des données qui ont été codées à l'aide d'une syntaxe de type XML. Un message peut contenir au moins deux structures MQRFH2 en série, les données utilisateur pouvant éventuellement suivre la dernière structure MQRFH2 de la série.

Format

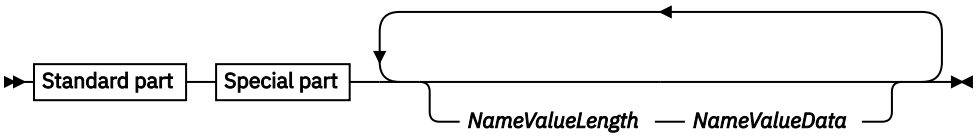
MQFMT_RF_HEADER_2

Syntax

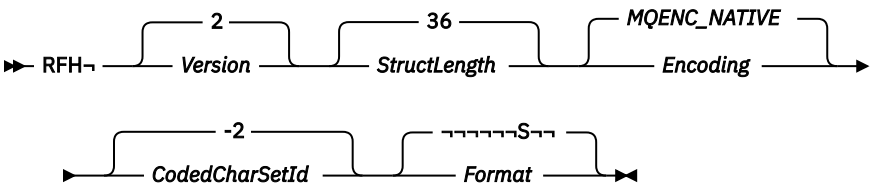
WebSphere MQ Message



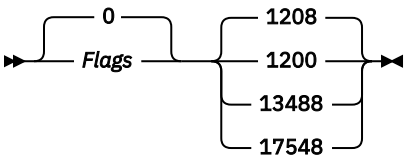
MQRFH2 header



Standard part



Special part



Jeu de caractères et codage

Des règles spéciales s'appliquent au jeu de caractères et au codage utilisés pour la structure MQRFH2 :

- Les zones autres que *NameValueData* sont dans le jeu de caractères et le codage donnés par les zones *CodedCharSetId* et *Encoding* de la structure d'en-tête qui précède MQRFH2, ou par les zones de la structure MQMD si MQRFH2 est au début des données de message d'application.

  Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

  Lorsque MQGMO_CONVERT est spécifié dans l'appel MQGET , le gestionnaire de files d'attente convertit les zones MQRFH2 , autres que *NameValueData*, en jeu de caractères et en codage requis.
- *NameValueData* correspond au jeu de caractères indiqué par la zone *NameValueCCSID* . Seuls les jeux de caractères Unicode répertoriés sont valides pour *NameValueCCSID* ; voir la description de *NameValueCCSID* pour plus de détails.

  Certains jeux de caractères ont une représentation qui dépend du codage. Si *NameValueCCSID* est l'un de ces jeux de caractères, *NameValueData* doit être dans le même codage que les autres zones du MQRFH2.

  Lorsque MQGMO_CONVERT est spécifié dans l'appel MQGET , le gestionnaire de files d'attente convertit *NameValueData* au codage demandé, mais ne modifie pas son jeu de caractères.

Zones pour MQRFH2

La structure MQRFH2 contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

CodedCharSetId (MQLONG)

Indique l'identificateur de jeu de caractères des données qui suivent la dernière zone *NameValueData* ; il ne s'applique pas aux données de type caractère de la structure MQRFH2 elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

MQCCSI_HÉRITER
Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI_INHERIT n'est pas renvoyée par l'appel MQGET.

MQCCSI_INHERIT ne peut pas être utilisé si la valeur de la zone *PutApplType* dans MQMD est MQAT_BROKER.

La valeur initiale de cette zone est MQCCSI_INHERIT.

Codage (MQLONG)

Indique le codage numérique des données qui suivent la dernière zone *NameValueData* ; il ne s'applique pas aux données numériques de la structure MQRFH2 elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est MQENC_NATIVE.

Indicateurs (MQLONG)

La valeur initiale de cette zone est MQRFH_NONE. MQRFH_NONE doit être précisé.

MQRFH_NONE
Aucun indicateur.

MQRFH_INTERNAL
L'en-tête MQRFH2 contient des propriétés définies en interne.

MQRFH_INTERNAL est destiné à être utilisé par le gestionnaire de files d'attente.

Les 16 premiers bits, MQRFH_FLAGS_RESTRICTED_MASK, sont réservés aux indicateurs des ensembles de gestionnaires de files d'attente. Les indicateurs qu'un utilisateur peut définir sont définis dans les 16 derniers bits.

Format (MQCHAR8)

Indique le nom de format des données qui suivent la dernière zone *NameValueData* .

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La valeur initiale de cette zone est MQFMT_NONE.

CCSID NameValue(MQLONG)

Indique l'identificateur de jeu de caractères codés des données de la zone *NameValueData* . Il est différent du jeu de caractères des autres chaînes de la structure MQRFH2 et peut être différent du jeu de caractères des données (le cas échéant) qui suit la dernière zone *NameValueData* à la fin de la structure.

NameValueCCSID doit avoir l'une des valeurs suivantes:

CCSID	Explication
1 200	UCS-2 ouvert
13488	UCS-2 2.0 sous-ensemble
17584	UCS-2 2.1 (inclut le symbole Euro)
1208	UTF-8

Pour les jeux de caractères UCS-2 , le codage (ordre des octets) de *NameValueData* doit être identique à celui des autres zones de la structure MQRFH2 . Les caractères de substitution (X'D800'à X'DFFF') ne sont pas pris en charge.

Remarque : Si *NameValueCCSID* ne possède pas l'une des valeurs répertoriées ci-dessus et que la structure MQRFH2 requiert une conversion sur l'appel MQGET, l'appel se termine avec le code anomalie MQRC_SOURCE_CCSID_ERROR et le message est renvoyé non converti.

La valeur initiale de cette zone est 1208.

Données NameValue(MQCHARn)
NameValueData est une zone de longueur variable qui contient un dossier contenant des paires nom / valeur de propriétés de message. Un dossier est une chaîne de caractères de longueur variable contenant des données codées à l'aide d'une syntaxe de type XML. La longueur en octets de la chaîne de caractères

est donnée par la zone *NameValueLength* qui précède la zone *NameValueData* . La longueur doit être un multiple de quatre.

Les zones *NameValueLength* et *NameValueData* sont facultatives, mais si elles sont présentes, elles doivent apparaître sous la forme d'une paire et être adjacentes. La paire de champs peut être répétée autant de fois que nécessaire, par exemple:

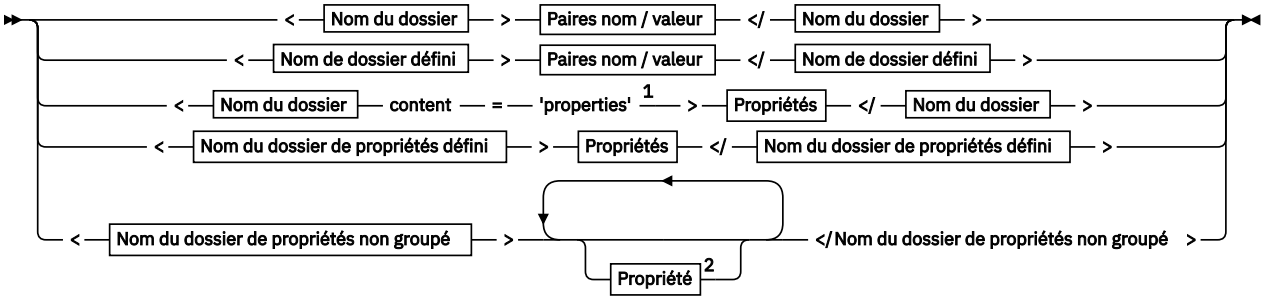
```
length1 data1 length2 data2 length3 data3
```

NameValueData n'est *pas* converti au jeu de caractères spécifié dans l'appel MQGET . Même si le message est extrait avec l'option MQGMO_CONVERT en vigueur, *NameValueData* reste dans son jeu de caractères d'origine. Toutefois, *NameValueData est* converti au codage spécifié dans l'appel MQGET .

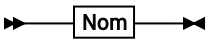
Remarque : Etant donné que ces zones sont facultatives, elles sont omises dans les déclarations de la structure fournies pour les différents langages de programmation pris en charge.

Remarque : Les termes "défini" et "réservé" sont utilisés dans le diagramme de syntaxe. "Défini" signifie que le nom est utilisé par IBM WebSphere MQ. "Réservé" signifie que le nom est réservé pour une utilisation ultérieure par WebSphere MQ.

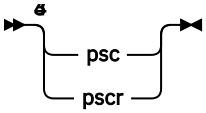
NameValueData Syntaxe



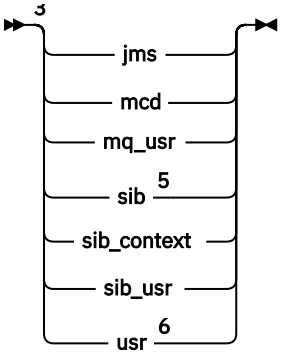
Nom du dossier



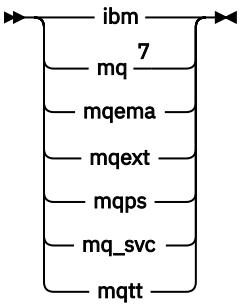
Nom de dossier défini



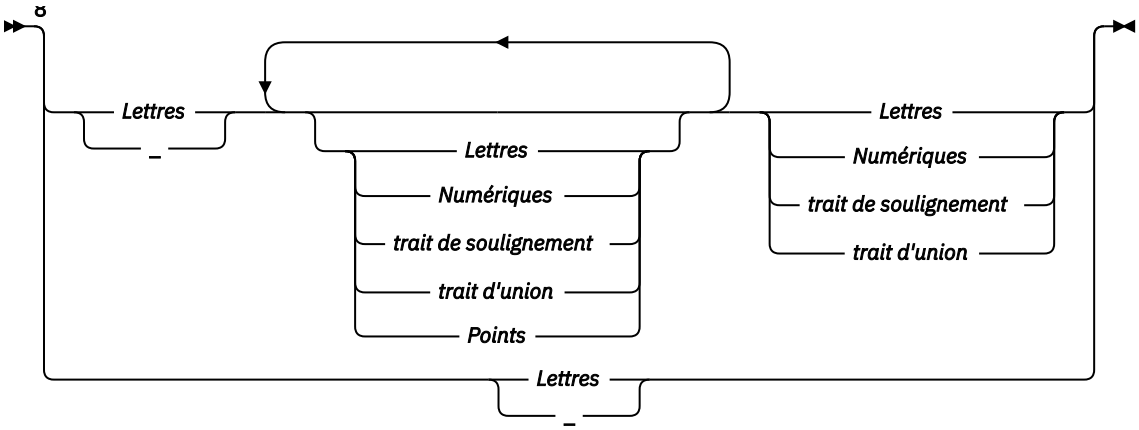
Nom du dossier de propriétés défini



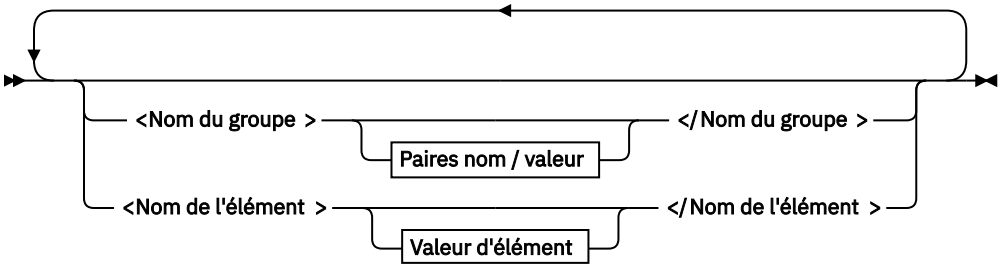
Nom du dossier de propriétés non groupé



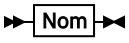
Nom



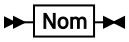
Paires nom / valeur



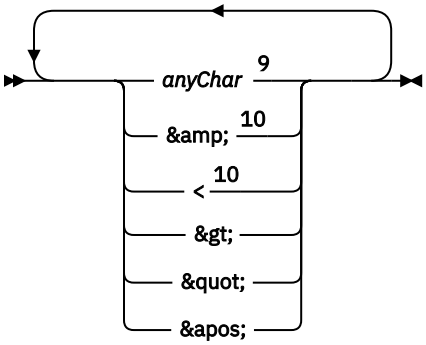
Nom du groupe



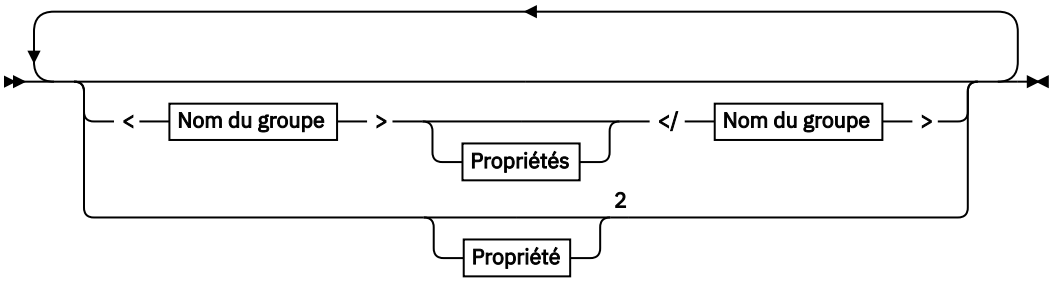
Nom de l'élément



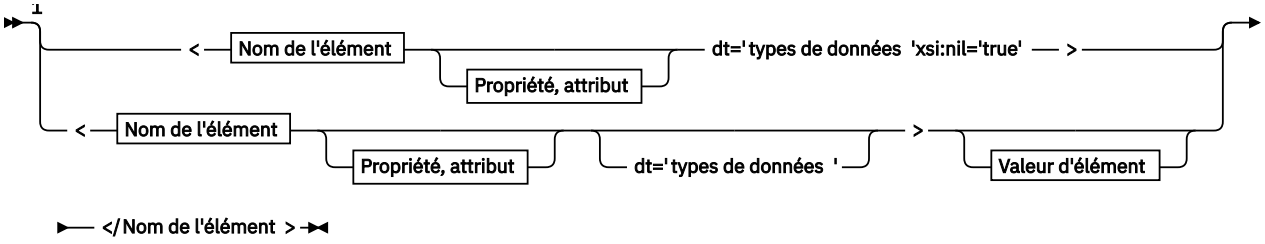
Valeur d'élément



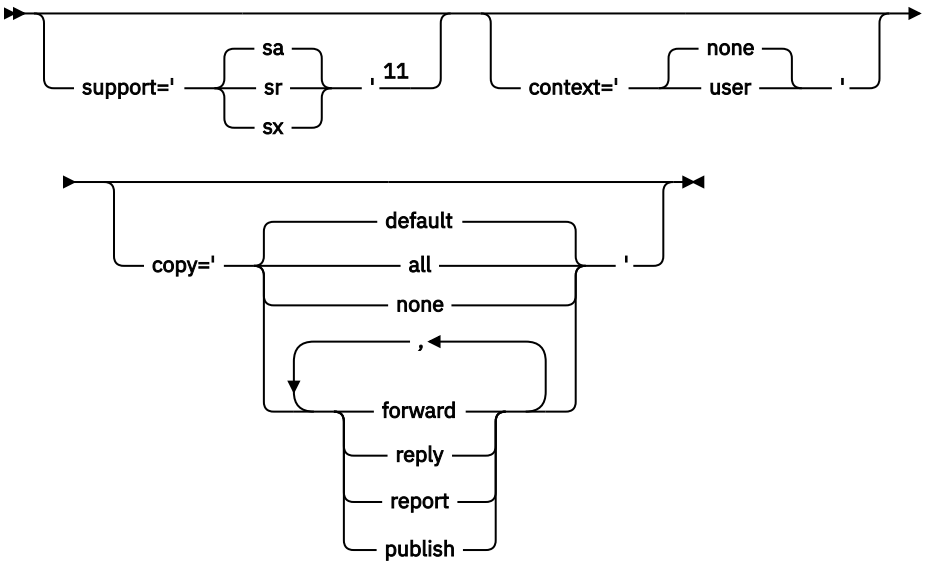
Propriétés



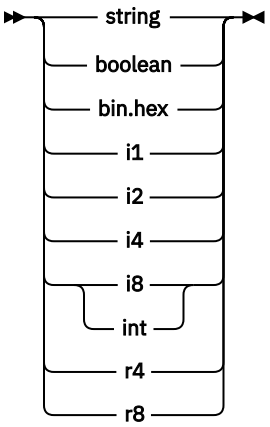
Propriété



Propriété, attribut



types de données



Remarques :

[1] Les guillemets ou les apostrophes sont valides.

[2] N'utilisez pas de nom de propriété non valide ; voir «Nom de propriété non valide», à la page 524. Utilisez un nom de propriété réservée uniquement à des fins définies ; voir «Noms de propriété définis», à la page 524.

[3] Le nom doit être en minuscules.

[4] Un seul dossier `psc` et `pscr` est pris en charge.

[5] Seules les propriétés du premier en-tête MQRFH2 sont significatives. WebSphere Application Server Service Integration Bus ignore les dossiers sib, sib_contextet sib_usr dans les en-têtes MQRFH2 suivants.

[6] Un MQRFH2ne doit pas contenir plus d'un dossier `usr` . Les propriétés du dossier `usr` ne doivent pas apparaître plus d'une fois.

[7] Seules les propriétés du premier dossier `mq` sont significatives. Si le dossier est `UTF-8`, seuls les caractères `UTF-8` mono-octet sont pris en charge. Le seul caractère blanc est Unicode `U+0020`.

[8] Les caractères valides sont définis dans la spécification XML W3C et se composent essentiellement de catégories Unicode `Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm,` et `Nd`.

[9] Tous les caractères sont significatifs. Les blancs de début et de fin font partie de la valeur de l'élément.

[10] N'utilisez pas de caractère non valide ; voir «Caractères non valides», à la page 524. Utilisez une séquence d'échappement plutôt que ces caractères non valides.

[11] L'attribut de propriété de support est valide uniquement sur le dossier `mq`

Nom du dossier

NameValueData contient un seul dossier. Pour créer plusieurs dossiers, créez plusieurs zones *NameValueData* . Vous pouvez créer plusieurs zones *NameValueData* dans un même en-tête MQRFH2 dans un message. Vous pouvez également créer plusieurs en-têtes MQRFH2 chaînés, chacun contenant plusieurs zones *NameValueData* .

L'ordre des en-têtes MQRFH2 et l'ordre des zones *NameValueData* ne font aucune différence avec le contenu logique d'un dossier. Si le même dossier est présent plusieurs fois dans un message, le dossier est analysé comme un tout. Si la même propriété se produit dans plusieurs instances du même dossier, elle est analysée en tant que liste.

Une analyse correcte d'un MQRFH2 n'est pas affectée par les autres manières dont un dossier peut être physiquement stocké dans un message.

Quatre dossiers ne respectent pas cette règle. Seule la première instance du dossier `mq`, `sib`, `sib_contextet` `sib_usr` est analysée.

Si la même propriété apparaît plusieurs fois dans le contenu combiné des en-têtes MQRFH2 chaînés, seule la première instance de la propriété est analysée. Si une propriété est définie à l'aide d'un appel d'API, tel que MQSETMP, et ajoutée à un MQRFH2 directement par une application, l'appel d'API est prioritaire.

Un nom de dossier est le nom d'un dossier contenant des paires nom-valeur ou des groupes. Les groupes et les paires nom-valeur peuvent être mélangés au même niveau dans l'arborescence des dossiers ; voir Figure 1, à la page 513. Ne combinez pas un nom de groupe et un nom d'élément ; voir Figure 2, à la page 513

```
<group1><nvp1>value</nvp1></group1><group2><nvp2>value</nvp2></group2>
<group3><nvp1>value</nvp1></group3><nvp3>value</nvp3>
```

Figure 1. Utilisations correctes des groupes et des paires nom-valeur

```
<group1><nvp1>value</nvp1>value</group1>
```

Figure 2. Utilisation incorrecte des groupes et des paires nom-valeur

N'utilisez pas de nom de dossier non valide ou réservé ; voir «Nom de chemin d'accès non valide», à la page 524 et «Nom du dossier réservé ou du dossier de propriétés», à la page 523. Utilisez un nom de dossier défini uniquement à des fins définies ; voir «Nom de dossier défini», à la page 514.

Si vous ajoutez l'attribut `'content=properties'` à la balise de nom de dossier, le dossier devient un dossier de propriétés ; voir Figure 3, à la page 514.

```
<myFolder></myfolder>
<myPropertyFolder contents='properties'></myPropertyFolder>
```

Figure 3. Exemple de dossier et de dossier de propriétés

Les noms de dossier sont sensibles à la casse. Les noms de dossier et les noms de dossier de propriété partagent le même espace de nom. Ils doivent avoir des noms différents. `Folder1` dans Figure 4, à la page 514 doit être un nom différent de `Folder2` dans Figure 5, à la page 514.

```
<Folder1><NVP1>value</NVP1></Folder1>
```

Figure 4. Espace de nom `Folder1`

```
<Folder2 content='properties'><Property1>value</Property1></Folder2>
```

Figure 5. Espace de nom `Folder2`

Les groupes, les propriétés et les paires nom-valeur des différents dossiers ont des espaces de nom différents. `Property1` dans Figure 5, à la page 514 est une propriété différente de `Property1` dans Figure 6, à la page 514.

```
<Folder3 content='properties'><Property1>value</Property1></Folder3>
```

Figure 6. Espace de nom `Folder3`

Les dossiers de propriétés sont différents des dossiers autres que des dossiers de propriétés à deux égards importants:

1. Les dossiers de propriétés contiennent des propriétés et les dossiers autres que les dossiers de propriétés contiennent des paires nom-valeur. Les dossiers diffèrent légèrement, du point de vue de la syntaxe.
2. Utilisez les interfaces définies, telles que les propriétés MQI ou les propriétés de message JMS, pour accéder aux propriétés de message. Les interfaces garantissent que les dossiers de propriétés dans MQRFH2 sont syntaxiquement appropriés. Un dossier de propriétés syntaxiquement correct est interopérable entre les gestionnaires de files d'attente sur différentes plateformes et différentes versions.

La propriété de message MQI est une méthode robuste de lecture et d'écriture d'un MQRFH2et évite les difficultés d'analyse syntaxique correcte d'un MQRFH2 .

Nom de dossier défini

Un nom de dossier défini est le nom d'un dossier réservé à WebSphere MQou à un autre produit. Ne créez pas de dossier du même nom et n'ajoutez pas vos propres paires nom / valeur aux dossiers. Les dossiers définis sont `psc` et `pscr`.

psc et pscr sont utilisés par la publication / l'abonnement en file d'attente.

Un message segmenté inséré avec MQMF_SEGMENT ou MQMF_SEGMENTATION_ALLOWED ne peut pas contenir de MQRFH2 avec un nom de dossier défini. Le MQPUT échoue avec le code anomalie 2443, MQRC_SEGMENTATION_NOT_ALLOWED.

Nom du dossier de propriétés défini

Un nom de dossier de propriétés défini est le nom d'un dossier de propriétés utilisé par IBM WebSphere MQou un autre produit. Pour les noms des dossiers et leur contenu, voir Dossiers de propriétés. Les noms de dossier de propriétés définis sont un sous-ensemble de tous les noms de dossier réservés par WebSphere MQ; voir «Nom du dossier réservé ou du dossier de propriétés», à la page 523.

Tout élément stocké dans un dossier de propriétés défini est une propriété. Un élément stocké dans un dossier de propriétés défini ne doit pas avoir d'attribut content='properties' .

Vous pouvez ajouter des propriétés uniquement aux dossiers de propriétés définis usr, mq_usret sib_usr. Dans d'autres dossiers de propriétés, tels que mq et sib, WebSphere MQ ignore ou rejette les propriétés qu'il ne reconnaît pas.

La description de chaque dossier de propriétés défini répertorie les propriétés définies par IBM WebSphere MQ qui peuvent être utilisées par les programmes d'application. Certaines des propriétés sont accessibles indirectement en définissant ou en obtenant une propriété JMS, et d'autres sont accessibles directement à l'aide des appels MQSETMP et MQINQMP MQI.

Les dossiers de propriétés définies contiennent également d'autres propriétés réservées par IBM WebSphere MQ , mais auxquelles les applications n'ont pas accès. Les noms des propriétés réservées ne sont pas répertoriés. Aucune propriété réservée n'est présente dans les dossiers de propriétés usr, mq_usret sib_usr . Mais ne créez pas de propriétés avec des noms de propriété non valides ; voir «Nom de propriété non valide», à la page 524.

Dossiers de propriétés

jms

jms contient des zones d'en-tête JMS et des propriétés JMSX qui ne peuvent pas être entièrement exprimées dans MQMD. Le dossier jms est toujours présent dans un MQRFH2JMS.

Tableau 532. Nom de la propriété jms, synonyme, type de données et dossier

Synonyme de propriété	Nom de la propriété	Type de données	Dossier
JMSDestination	jms.Dst	string	<jms><Dst>*destination*</Dst></jms>
JMSExpiration	jms.Exp	i8	<jms><Exp>*expiration*</Exp></jms>
corrélation JMS	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>

Tableau 532. Nom de la propriété jms, synonyme, type de données et dossier (suite)			
Synonyme de propriété	**Nom de la propriété**	**Type de données**	**Dossier**
JMSXGroupID	jms.Gid	string	<jms><Gid>*groupId*</Gid></jms>
JMSXGroupSeq	jms.Seq	i4	<jms><Seq>*messageSequenceNo*</Seq></jms>

N'ajoutez pas vos propres propriétés dans le dossier jms.

mcd

mcd contient des ropriétés qui décrivent le format du message. Par exemple, la propriété du domaine de service de message Msd identifie un message JMS comme étant JMSTextMessage, JMSBytesMessage, JMSStreamMessage, JMSMapMessage, JMSObjectMessage ou NULL.

Le dossier mcd est toujours présent dans un message JMS contenant un MQRFH2.

Il est toujours présent dans un message contenant un message MQRFH2 envoyé par WebSphere Message Broker. Il décrit le domaine, le format, le type et l'ensemble de messages d'un message.

Tableau 533. Nom de la propriété mcd, synonyme, type de données et dossier			
Synonyme de propriété	**Nom de la propriété**	**Type de données**	**Dossier**
	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>

N'ajoutez pas vos propres propriétés dans le dossier mcd.

mq_usr

mq_usr contient des propriétés définies par l'application qui ne sont pas exposées en tant que propriétés définies par l'utilisateur JMS. Les propriétés qui ne répondent pas aux exigences JMS peuvent être placées dans ce dossier.

Vous pouvez créer des propriétés dans le dossier mq_usr . Les propriétés que vous créez dans le mq_usr sont des propriétés que vous créez dans de nouveaux dossiers avec l'attribut content='properties' .

sib

sib contient des propriétés de message système WebSphere Application Server Service Integration Bus (WAS/SIB). Les propriétés sib ne sont pas exposées en tant que propriétés JMS dans les applications JMS IBM WebSphere MQ car elles ne sont pas des types pris en charge. Par exemple, certaines propriétés sib ne peuvent pas être exposées en tant que propriétés JMS car il s'agit de tableaux d'octets. Certaines propriétés sib sont exposées aux applications WAS/SIB en tant que

propriétés JMS_IBM_* ; elles incluent les propriétés des chemins de routage de réacheminement et de réacheminement inverse.

N'ajoutez pas vos propres propriétés dans le dossier sib.

sib_context

sib_context contient des propriétés de message système WAS/SIB qui ne sont pas exposées aux applications utilisateur WAS/SIB ou en tant que propriétés JMS. sib_context contient les propriétés de sécurité et transactionnelles utilisées pour les services Web.

N'ajoutez pas vos propres propriétés dans le dossier sib_context.

sib_usr

sib_usr contient des propriétés de message utilisateur WAS/SIB qui ne sont pas exposées en tant que propriétés utilisateur JMS car elles ne sont pas de types pris en charge. sib_usr est exposé aux applications WAS/SIB dans l'interface SIMessage ; voir Développement d'une intégration de services.

Le type d'une propriété sib_usr doit être bin.hexet la valeur doit être au format correct. Si une application IBM WebSphere MQ écrit un élément de type bin.hex dans le dossier dans un format incorrect, l'application reçoit un IOException. Si le type de données de la propriété n'est pas bin.hex , l'application reçoit un ClassCastException.

N'essayez pas de rendre les propriétés utilisateur JMS disponibles pour WAS/SIB à l'aide de ce dossier ; utilisez plutôt le dossier usr .

Vous pouvez créer des propriétés dans le dossier sib_usr .

usr

usr contient les propriétés JMS définies par l'application associées au message. Le dossier usr n'est présent que si une application a défini une propriété définie par l'application.

usr est le dossier de propriétés par défaut. Si une propriété est définie sans nom de dossier, elle est placée dans le dossier usr .

Tableau 534. Nom de la propriété usr, synonyme, type de données et dossier.

Les valeurs des propriétés des services Web sont décrites dans MQRFH2 Paramètres SOAP

Synonyme de propriété	Nom de la propriété	Type de données	Dossier
	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>

Vous pouvez créer des propriétés dans le dossier usr .

Un message segmenté inséré avec MQMF_SEGMENT ou MQMF_SEGMENTATION_ALLOWED ne peut pas contenir de MQRFH2 avec un nom de dossier de propriété défini. Le MQPUT échoue avec le code anomalie 2443, MQRC_SEGMENTATION_NOT_ALLOWED.

Nom du dossier de propriétés non groupé

ibm

ibm contient des propriétés qui sont utilisées uniquement par IBM WebSphere MQ.

Tableau 535. Nom de la propriété ibm, synonyme, type de données et dossier

Synonyme de propriété	Nom de la propriété	Type de données	Dossier
	ibm.rfp	string	<ibm><rfp>fingerprint</rfp></ibm>

N'ajoutez pas vos propres propriétés dans le dossier ibm.

mq

mq contient des propriétés qui sont utilisées uniquement par IBM WebSphere MQ.

Les restrictions suivantes s'appliquent aux propriétés du dossier mq :

- Seules les propriétés du premier dossier mq significatif du message sont traitées par MQ; les propriétés de tout autre dossier mq du message sont ignorées.
- Seuls les caractères UTF-8 mono-octet sont autorisés dans le dossier. Un caractère multi-octet dans le dossier peut entraîner l'échec de l'analyse syntaxique et le rejet du message.
- N'utilisez pas de chaînes d'échappement dans le dossier. Une chaîne d'échappement est traitée comme la valeur réelle de l'élément.
- Seul le caractère Unicode U+0020 est traité comme un espace dans le dossier. Tous les autres caractères sont traités comme significatifs et peuvent entraîner l'échec de l'analyse syntaxique du dossier et le rejet du message.

Si l'analyse du dossier mq échoue ou si le dossier ne respecte pas ces restrictions, le message est rejeté avec le code anomalie 2527, MQRC_RFH_RESTRICTED_FORMAT_ERR.

N'ajoutez pas vos propres propriétés dans le dossier mq.

mqema

mqema contient des propriétés utilisées uniquement par WebSphere Application Server. Le dossier a été remplacé par mqext.

N'ajoutez pas vos propres propriétés dans le dossier mqema.

mqext

mqext contient des propriétés utilisées uniquement par WebSphere Application Server. Le dossier est présent uniquement si l'application a défini au moins l'une des propriétés IBM définies.

Tableau 536. Nom de la propriété mqext, synonyme, type de données et dossier

Synonyme de propriété	Nom de la propriété	Type de données	Dossier
JMSArmCorrelator	mqext.Arm	string	<mqext><Arm>*armCorrelator*</Arm></mqext>

Tableau 536. Nom de la propriété mqext, synonyme, type de données et dossier (suite)			
Synonyme de propriété	**Nom de la propriété**	**Type de données**	**Dossier**
JMSRMCorrelator	mqext.Wrm	string	<mqext><Wrm>*wrmCorrelator*</Wrm></mqext>

N'ajoutez pas vos propres propriétés dans le dossier mqext.

mqps

mqps contient des propriétés qui sont utilisées uniquement par la publication/l'abonnement IBM WebSphere MQ. Le dossier est présent uniquement si l'application a défini au moins l'une des propriétés de publication/abonnement intégré.

Tableau 537. Nom de la propriété mqps, synonyme, type de données et dossier			
Synonyme de propriété	**Nom de la propriété**	**Type de données**	**Dossier**
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>

N'ajoutez pas vos propres propriétés dans le dossier mqps.

mq_svc

mq_svc contient les propriétés utilisées par SupportPac MA93.

N'ajoutez pas vos propres propriétés dans le dossier mq_svc.

mqtt

mqtt contient les propriétés utilisées par IBM WebSphere MQ Telemetry

Tableau 538. Nom de la propriété mqtt, synonyme, type de données et dossier			
Synonyme de propriété	**Nom de la propriété**	**Type de données**	**Dossier**
	mqtt.clientId	string	<mqtt><clientId>*topicString*</clientId></mqtt>
	mqtt.qos	i4	<mqtt><qos>*qualityOfService*</qos></mqtt>
	mqtt.msgid	string	<mqtt><msgid>*messageIdentifier*</msgid></mqtt>

N'ajoutez pas vos propres propriétés dans le dossier mqtt.

Un message segmenté inséré avec MQMF_SEGMENT ou MQMF_SEGMENTATION_ALLOWED ne peut pas contenir de MQRFH2 avec un nom de dossier de propriétés non groupé. Le MQPUT échoue avec le code anomalie 2443, MQRC_SEGMENTATION_NOT_ALLOWED.

Paires nom / valeur

Dans le diagramme de syntaxe, "paires nom / valeur" décrit le contenu d'un dossier ordinaire. Un dossier ordinaire contient des groupes et des éléments. Un élément est une paire nom / valeur. Un groupe contient des éléments et d'autres groupes.

En termes d'arbres, les éléments sont des noeuds feuille et les groupes sont des noeuds internes. Un noeud interne et le dossier, qui est le noeud racine, peuvent contenir un mélange de noeuds internes et de noeuds feuilles. Un noeud ne peut pas être à la fois un noeud interne et un noeud feuille ; voir Figure 2, à la page 513.

Propriétés

Dans le diagramme de syntaxe, "Propriétés" décrit le contenu d'un dossier de propriétés. Un dossier de propriétés contient des groupes et des propriétés. Une propriété est une paire nom / valeur avec un attribut de type de données facultatif. Un groupe contient des propriétés et d'autres groupes.

En termes d'arbres, les propriétés sont des noeuds feuilles et les groupes sont des noeuds internes. Un noeud interne et le dossier de propriétés, qui est le noeud racine, peuvent contenir un mélange de noeuds internes et de noeuds feuilles. Un noeud ne peut pas être à la fois un noeud interne et un noeud feuille ; voir Figure 2, à la page 513.

Propriété

Une propriété de message est une paire nom / valeur dans un dossier de propriétés. Il peut éventuellement inclure un attribut de type de données et un attribut de propriété ; pour un exemple, voir Figure 7, à la page 520. Si l'attribut de type de données est omis, le type de propriété est string.

```
<pf><p1 dt='i8' >value</p1></pf>
```

Figure 7. Type de données, attribut

Le nom d'une propriété de message est son nom de chemin complet, avec la syntaxe <> de type XML, remplacée par des points. Par exemple, myPropertyFolder1.myGroup1.myGroup2.myProperty1 est mappé à une chaîne *NameValueData* dans Figure 8, à la page 521. La chaîne est formatée pour faciliter la lecture.

```
<myPropertyFolder1>
    <myGroup1>
        <myGroup2>
            <myProperty1>value</myProperty1>
        </myGroup2>
    </myGroup1>
</myPropertyFolder1>
```

Figure 8. Mappage de nom de propriété unique

Un dossier de propriétés peut contenir plusieurs propriétés. Par exemple, les propriétés dans Figure 9, à la page 521 sont mappées au dossier de propriétés dans Figure 10, à la page 521

```
myPropertyFolder1.myProperty4
myPropertyFolder1.myGroup1.myGroup2.myProperty1
myPropertyFolder1.myGroup1.myGroup2.myProperty2
myPropertyFolder1.myGroup1.myProperty3
```

Figure 9. Plusieurs propriétés avec le même nom de racine

```
<myPropertyFolder1>
    <myProperty4>value</myProperty4>
    <myGroup1>
        <myGroup2>
            <myProperty1>value</myProperty1>
            <myProperty2>value</myProperty2>
        </myGroup2>
        <myProperty3>value</myProperty3>
    </myGroup1>
</myPropertyFolder1>
```

Figure 10. Mappage de noms de propriété multiples

Nom

Un nom doit commencer par une *lettre* ou un *soulignement*. Il ne doit pas contenir de *Colon*, ne doit pas se terminer par une *période* et ne doit pas contenir que des *lettres*, des *chiffres*, des *points*, des *Hyphens*et des *points*. Les caractères valides sont définis dans la spécification XML W3C et se composent essentiellement de catégories Unicode `Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm,` et `Nd`.

Le chemin complet d'une propriété ou d'une paire nom-valeur ne doit pas enfreindre la règle décrite dans «Nom de chemin d'accès non valide», à la page 524. Les chemins d'accès sont limités à 4095 octets, ne doivent pas contenir de caractères de compatibilité Unicode et ne doivent pas commencer par la chaîne `XML`.

Nom du groupe

Un nom de groupe a la même syntaxe qu'un nom. Les noms de groupe sont facultatifs. Les propriétés et les paires nom-valeur peuvent être placées à la racine d'un dossier. Utilisez des groupes si cela permet d'organiser les propriétés et les paires nom-valeur.

Nom de l'élément

Un nom d'élément a la même syntaxe qu'un nom.

Valeur d'élément

Une valeur d'élément inclut tous les espaces entre la balise <*Element name*> et </*Element name*>. N'utilisez pas les deux caractères < et & dans une valeur. Remplacez ensuite par < et &.

Propriété, attribut

Les attributs de propriété mappent les zones de descripteur de propriété: Les mappages sont les suivants:

Support

sa
MQPD_SUPPORT_OPTIONAL

sr
MQPD_SUPPORT_REQUIRED

sx
MQPD_SUPPORT_REQUIRED_IF_LOCAL

Contexte

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

N'utilisez pas all en combinaison avec d'autres options.

default
MQPD_COPY_DEFAULT

N'utilisez pas default en combinaison avec d'autres options. default est identique à forward + report + publish

none
MQPD_COPY_NONE

N'utilisez pas none en combinaison avec d'autres options.

Les attributs de la propriété Support ne s'appliquent qu'aux propriétés du dossier mq .

Les attributs de propriété Context et CopyOptions sont applicables à tous les dossiers de propriétés.

Type de données

Les types de données MQRFH2 sont mappés aux types de propriété de message comme suit:

Tableau 539. Mappages de type de données	
Type de données MQRFH2	**Type de propriété de message**
bin.hex	MQBYTE[]
boolean	MQBOOL
i1	MQINT8
i2	MQINT16
i4	MQINT32
i8	MQINT64
r4	MQFLOAT32
r8	MQFLOAT64
string	MQCHAR[]

Tout élément sans type de données est supposé être de type `string`.

Une valeur nulle est indiquée par l'attribut d'élément `xsi:nil='true'`. N'utilisez pas l'attribut `xsi:nil='false'` pour les valeurs non nulles. Par exemple, la propriété suivante a une valeur null:

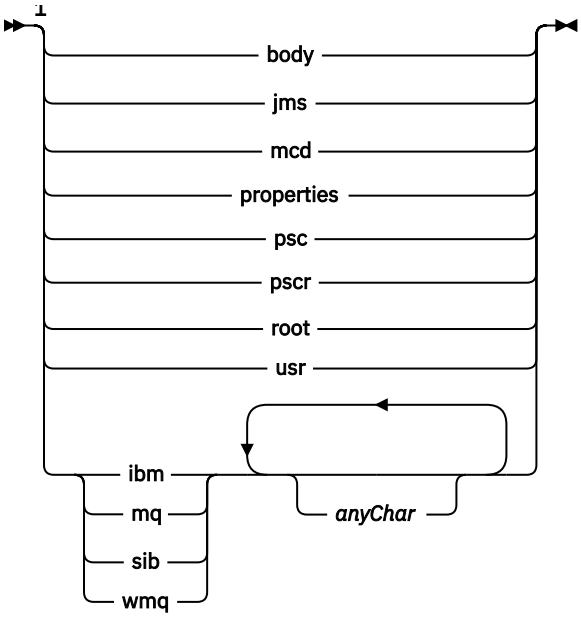
```
<NullProperty
xsi:nil='true'></NullProperty>
```

Une propriété d'octet ou de chaîne de caractères peut avoir une valeur vide. Une valeur vide est représentée par un élément MQRFH2 avec une valeur d'élément de longueur nulle. Par exemple, la propriété suivante a une valeur vide:

```
<EmptyProperty></EmptyProperty>
```

Nom du dossier réservé ou du dossier de propriétés

Limitez le nom d'un dossier ou d'un dossier de propriétés à ne pas commencer par l'une des chaînes suivantes. Les préfixes sont réservés aux noms de dossier ou de propriété créés par IBM.

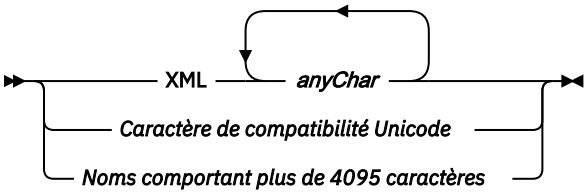


Remarques :

[1] Un nom de dossier ou de propriété réservé contient une combinaison de lettres minuscules et majuscules.

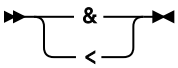
Nom de chemin d'accès non valide

Limitez le chemin complet d'une paire nom-valeur ou d'une propriété pour ne pas inclure l'une des chaînes suivantes.



Caractères non valides

Utilisez toujours les séquences d'échappement & et < à la place des littéraux "&" et "<".

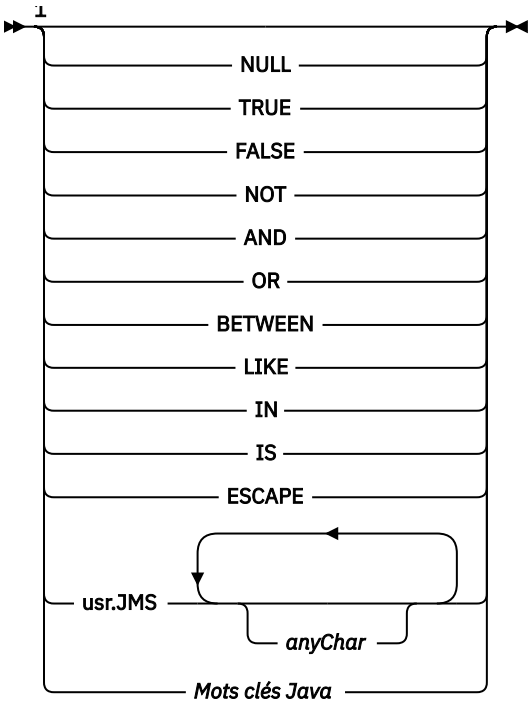


Noms de propriété définis

Les noms de propriété définis sont les noms des propriétés définies par WebSphere MQ, ou d'autres produits, et utilisées par IBM WebSphere MQ et les applications utilisateur. Les propriétés définies existent uniquement dans les dossiers de propriétés définis. Les noms de propriété définis sont décrits dans la description des dossiers de propriétés ; voir Dossiers de propriétés.

Nom de propriété non valide

Ne construisez pas de noms de propriété qui correspondent à la règle suivante. La règle s'applique au chemin d'accès complet à la propriété qui nomme une propriété, et pas seulement au nom de l'élément de propriété.



Remarques :

[1] Un nom de propriété non valide peut contenir n'importe quelle combinaison de majuscules et de minuscules.

NameValueLongueur (MQLONG)
Longueur de la zone `NameValueData` correspondante

Indique la longueur en octets des données de la zone *`NameValueData`* . *`NameValueLength`* doit être un multiple de quatre.

Remarque : Les zones *`NameValueLength`* et *`NameValueData`* sont facultatives, mais si elles sont présentes, elles doivent apparaître sous la forme d'une paire et être adjacentes. La paire de champs peut être répétée autant de fois que nécessaire, par exemple:

```
length1 data1 length2 data2 length3 data3
```

Etant donné que ces zones sont facultatives, elles sont omises dans les déclarations de la structure fournies pour les différents langages de programmation pris en charge.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID MQRFH_STRUC_ID
Identificateur des règles et de la structure d'en-tête de formatage.

Pour le langage de programmation C, la constante MQRFH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQRFH_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQRFH_STRUC_ID.

StrucLength (MQLONG)

Il s'agit de la longueur en octets de la structure MQRFH2 , y compris les zones *NameValueLength* et *NameValueData* à la fin de la structure. Il est possible qu'il y ait plusieurs paires de zones *NameValueLength* et *NameValueData* à la fin de la structure, dans la séquence:

```
length1, data1, length2, data2, ...
```

StrucLength n'inclut *pas* de données utilisateur pouvant suivre la dernière zone *NameValueData* à la fin de la structure.

Pour éviter les problèmes liés à la conversion des données utilisateur dans certains environnements, *StrucLength* doit être un multiple de quatre.

La constante suivante indique la longueur de la partie *fixe* de la structure, c'est-à-dire la longueur à l'exclusion des zones *NameValueLength* et *NameValueData* :

MQRFH_STRUC_LENGTH_FIXED_2
Longueur de la partie fixe de la structure MQRFH2 .

La valeur initiale de cette zone est MQRFH_STRUC_LENGTH_FIXED_2.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQRFH_VERSION_2
Règles Version-2 et structure d'en-tête de formatage.

La valeur initiale de cette zone est MQRFH_VERSION_2.

Valeurs initiales et déclarations de langue pour MQRFH2

Tableau 540. Valeurs initiales des zones dans MQRFH2 pour MQRFH2

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID MQRFH_STRUC_ID	'RFH¬'
Version	MQRFH_VERSION_2	2
StrucLength	MQRFH_STRUC_LENGTH_FIXED_2	36
Encoding	MQENC_NATIVE	Dépend de l'environnement
CodedCharSetId	MQCCSI_HÉRITER	-2
Format	MQFMT_AUCUN	Espaces vides
Flags	MQRFH_AUCUN	0
NameValueCCSID	Aucun	1208

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQRFH2_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQRFH2 MyRFH2 = {MQRFH2_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-En-tête de message de référence

Le tableau suivant récapitule les zones de la structure.

Tableau 541. Zones dans MQRMH

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
StrucLength	Longueur totale de MQRMH, y compris les chaînes à la fin de zones fixes, mais pas les données non formatées	StrucLength
Encoding	Codage numérique des données non formatées	Codage
CodedCharSetId	Identificateur de jeu de caractères des données non formatées	CodedCharSetId
Format	Nom du format des données non formatées	Format
Flags	Identificateurs de message de référence	Indicateurs
ObjectType	Type d'objet	ObjectType
ObjectInstanceId	Identificateur d'instance d'objet	ObjectInstanceID
SrcEnvLength	Longueur des données d'environnement source	SrcEnvLongueur
SrcEnvOffset	Décalage des données d'environnement source	DécalageSrcEnv
SrcNameLength	Longueur du nom d'objet source	SrcNameLongueur
SrcNameOffset	Décalage du nom d'objet source	DécalageSrcName
DestEnvLength	Longueur des données d'environnement de destination	DestEnvLongueur

Tableau 541. Zones dans MQRMH (suite)		
Zone	**Description**	**Topic**
DestEnvOffset	Décalage des données d'environnement de destination	DécalageDestEnv
DestNameLength	Longueur du nom d'objet de destination	DestNameLongueur
DestNameOffset	Décalage du nom d'objet de destination	DestNameDécalage
DataLogicalLength	Longueur des données non formatées	DataLogicalDataLogical
DataLogicalOffset	Décalage faible des données non formatées	DécalageDataLogical
DataLogicalOffset2	Décalage élevé des données non formatées	DataLogicalOffset2

Présentation de MQRMH

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ connectés à ces systèmes.

Objectif: La structure MQRMH définit le format d'un en-tête de message de référence. Cet en-tête est utilisé avec les exits de canal de message écrits par l'utilisateur pour envoyer de très grandes quantités de données (appelées *données non formatées*) d'un gestionnaire de files d'attente à un autre. La différence par rapport à la messagerie normale est que les données non formatées ne sont pas stockées dans une file d'attente ; à la place, seule une *référence* aux données non formatées est stockée dans la file d'attente. Cela réduit la possibilité que les ressources MQ soient épuisées par un petit nombre de messages extrêmement volumineux.

Nom de format: MQFMT_REF_MSG_HEADER.

Jeu de caractères et codage: les données de type caractère dans MQRMH et les chaînes traitées par les zones de décalage doivent figurer dans le jeu de caractères du gestionnaire de files d'attente local ; cet attribut est fourni par l'attribut de gestionnaire de files d'attente *CodedCharSetId* . Les données numériques dans MQRMH doivent être dans le codage machine natif ; elles sont fournies par la valeur de MQENC_NATIVE pour le langage de programmation C.

Définissez le jeu de caractères et le codage du MQRMH dans les zones *CodedCharSetId* et *Encoding* dans:

- MQMD (si la structure MQRMH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQRMH (tous les autres cas).

Utilisation: une application insère un message composé d'un MQRMH, mais en omettant les données non formatées. Lorsqu'un agent MCA lit le message à partir de la file d'attente de transmission, un exit de message fourni par l'utilisateur est appelé pour traiter l'en-tête de message de référence. L'exit peut ajouter au message de référence les données non formatées identifiées par la structure MQRMH, avant que l'agent MCA n'envoie le message via le canal au gestionnaire de files d'attente suivant.

A l'extrémité réceptrice, un exit de message qui attend les messages de référence doit exister. Lorsqu'un message de référence est reçu, l'exit doit créer l'objet à partir des données non formatées qui suivent le MQRMH dans le message, puis transmettre le message de référence sans les données non formatées. Le message de référence peut ensuite être extrait par une application lisant le message de référence (sans les données non formatées) à partir d'une file d'attente.

Normalement, la structure MQRMH est tout ce qui se trouve dans le message. Toutefois, si le message se trouve dans une file d'attente de transmission, un ou plusieurs en-têtes supplémentaires précèdent la structure MQRMH.

Un message de référence peut également être envoyé à une liste de distribution. Dans ce cas, la structure MQDH et ses enregistrements associés précèdent la structure MQRMH lorsque le message se trouve dans une file d'attente de transmission.

Remarque : N'envoyez pas de message de référence en tant que message segmenté, car l'exit de message ne peut pas le traiter correctement.

Conversion de données: à des fins de conversion de données, la conversion de la structure MQRMH inclut la conversion des données d'environnement source, du nom d'objet source, des données d'environnement de destination et du nom d'objet de destination. Tous les autres octets compris dans *StrucLength* octets du début de la structure sont supprimés ou ont des valeurs non définies après la conversion des données. Les données non formatées sont converties à condition que toutes les conditions suivantes soient remplies:

- Les données non formatées sont présentes dans le message lorsque la conversion de données est effectuée.
- La valeur de la zone *Format* dans MQRMH est différente de MQFMT_NONE.
- Un exit de conversion de données écrit par l'utilisateur existe avec le nom de format indiqué.

Sachez toutefois que les données non formatées ne sont généralement *pas* présentes dans le message lorsque le message se trouve dans une file d'attente et que, par conséquent, les données non formatées sont converties par l'option MQGMO_CONVERT.

Zones pour MQRMH

La structure MQRMH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

CodedCharSetId (MQLONG)

Indique l'identificateur de jeu de caractères des données non formatées ; il ne s'applique pas aux données de type caractères de la structure MQRMH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. La valeur spéciale suivante peut être utilisée:

MQCCSI_HÉRITER
Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI_INHERIT n'est pas renvoyée par l'appel MQGET.

N'utilisez pas MQCCSI_INHERIT si la valeur de la zone *PutApplType* dans MQMD est MQAT_BROKER.

Cette valeur est prise en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus les clients WebSphere MQ connectés à ces systèmes.

La valeur initiale de cette zone est MQCCSI_UNDEFINED.

Longueur DataLogical(MQLONG)

La zone *DataLogicalLength* indique la longueur des données non formatées référencées par la structure MQRMH.

Si les données non formatées sont effectivement présentes dans le message, les données commencent à un décalage de *StrucLength* octets à partir du début de la structure MQRMH. La longueur de l'ensemble du message moins *StrucLength* indique la longueur des données non formatées présentes.

Si des données sont présentes dans le message, *DataLogicalLength* indique la quantité de données pertinentes. Dans le cas normal, *DataLogicalLength* doit avoir la même valeur que la longueur des données présentes dans le message.

Si la structure MQRMH représente les données restantes dans l'objet (à partir du décalage logique spécifié), vous pouvez utiliser la valeur zéro pour *DataLogicalLength*, à condition que les données non formatées ne soient pas réellement présentes dans le message.

Si aucune donnée n'est présente, la fin de MQRMH coïncide avec la fin du message.

La valeur initiale de cette zone est 0.

DataLogicalOffset (MQLONG)

Cette zone indique le décalage faible des données non formatées à partir du début de l'objet dont font partie les données non formatées. Le décalage des données non formatées à partir du début de l'objet est appelé *décalage logique*. Il ne s'agit *pas* du décalage physique des données non formatées depuis le début de la structure MQRMH ; ce décalage est fourni par *StrucLength*.

Pour permettre l'envoi d'objets volumineux à l'aide de messages de référence, le décalage logique est divisé en deux zones et le décalage logique réel est donné par la somme de ces deux zones:

- *DataLogicalOffset* représente le reste obtenu lorsque le décalage logique est divisé par 1 000 000 000. Il s'agit donc d'une valeur comprise entre 0 et 999 999 999.
- *DataLogicalOffset2* représente le résultat obtenu lorsque le décalage logique est divisé par 1 000 000 000. C'est donc le nombre de multiples complets de 1 000 000 000 qui existent dans le décalage logique. Le nombre de multiples est compris entre 0 et 999 999 999.

La valeur initiale de cette zone est 0.

DataLogicalOffset2 (MQLONG)

Cette zone indique le décalage élevé des données non formatées à partir du début de l'objet dont font partie les données non formatées. Il s'agit d'une valeur comprise entre 0 et 999 999 999. Voir *DataLogicalOffset* pour des détails.

La valeur initiale de cette zone est 0.

DestEnvLongueur (MQLONG)

Longueur des données de l'environnement de destination. Si cette zone a pour valeur zéro, il n'y a pas de données d'environnement de destination et *DestEnvOffset* est ignoré.

DestEnvDécalage (MQLONG)

Cette zone indique le décalage des données d'environnement de destination à partir du début de la structure MQRMH. Les données d'environnement de destination peuvent être spécifiées par le créateur du message de référence, si ces données sont connues du créateur. Par exemple, sous Windows , les données d'environnement de destination peuvent être le chemin de répertoire de l'objet dans lequel les données non formatées doivent être stockées. Toutefois, si le créateur ne connaît pas les données d'environnement de destination, il incombe à l'exit de message fourni par l'utilisateur de déterminer les informations d'environnement requises.

La longueur des données d'environnement de destination est indiquée par *DestEnvLength*; si cette longueur est égale à zéro, il n'y a pas de données d'environnement de destination et *DestEnvOffset* est ignoré. S'il est présent, les données de l'environnement de destination doivent résider entièrement dans *StrucLength* octets à partir du début de la structure.

Les applications ne doivent pas supposer que les données de l'environnement de destination sont contiguës à l'une des données traitées par les zones *SrcEnvOffset*, *SrcNameOffset* et *DestNameOffset* .

La valeur initiale de cette zone est 0.

DestNameLongueur (MQLONG)

Longueur du nom de l'objet de destination. Si cette zone est à zéro, il n'y a pas de nom d'objet de destination et *DestNameOffset* est ignoré.

DestNameDécalage (MQLONG)

Cette zone indique le décalage du nom d'objet de destination à partir du début de la structure MQRMH. Le nom de l'objet de destination peut être spécifié par le créateur du message de référence, si ces données sont connues du créateur. Toutefois, si le créateur ne connaît pas le nom de l'objet de destination, il incombe à l'exit de message fourni par l'utilisateur d'identifier l'objet à créer ou à modifier.

La longueur du nom d'objet de destination est indiquée par *DestNameLength*; si cette longueur est égale à zéro, il n'y a pas de nom d'objet de destination et *DestNameOffset* est ignoré. S'il est présent, le nom de l'objet de destination doit résider entièrement dans *StrucLength* octets à partir du début de la structure.

Les applications ne doivent pas supposer que le nom de l'objet de destination est contigu à l'une des données traitées par les zones *SrcEnvOffset*, *SrcNameOffset*et *DestEnvOffset* .

La valeur initiale de cette zone est 0.

Codage (MQLONG)

Indique le codage numérique des données non formatées ; il ne s'applique pas aux données numériques de la structure MQRMH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est MQENC_NATIVE.

Indicateurs (MQLONG)

Il s'agit d'indicateurs de message de référence. Les indicateurs suivants sont définis:

MQRMHF_LAST
Cet indicateur indique que le message de référence représente ou contient la dernière partie de l'objet référencé.

MQRMHF_NOT_LAST
Le message de référence ne contient pas ou ne représente pas la dernière partie de l'objet. MQRMHF_NOT_LAST aide la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, cette utilisation ne peut pas être détectée.

La valeur initiale de cette zone est MQRMHF_NOT_LAST.

Format (MQCHAR8)

Indique le nom de format des données non formatées.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La valeur initiale de cette zone est MQFMT_NONE.

ID ObjectInstance(MQBYTE24)

Utilisez cette zone pour identifier une instance spécifique d'un objet. S'il n'est pas nécessaire, définissez-le sur la valeur suivante:

MQOII_AUCUN
Aucun identificateur d'instance d'objet spécifié. La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQOII_NONE_ARRAY est également définie ; elle a la même valeur que MQOII_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ_OBJECT_INSTANCE_ID_LENGTH. La valeur initiale de cette zone est MQOII_NONE.

ObjectType (MQCHAR8)

Il s'agit d'un nom que l'exit de message peut utiliser pour reconnaître les types de message de référence qu'il prend en charge. Le nom doit respecter les mêmes règles que la zone *Format* décrite ci-dessus.

La valeur initiale de cette zone est 8 blancs.

Longueur SrcEnv(MQLONG)

Longueur des données d'environnement source. Si cette zone est définie sur zéro, il n'y a pas de données d'environnement source et *SrcEnvOffset* est ignoré.

La valeur initiale de cette zone est 0.

SrcEnvDécalage (MQLONG)

Cette zone indique le décalage des données d'environnement source à partir du début de la structure MQRMH. Les données d'environnement source peuvent être spécifiées par le créateur du message de référence, si ces données sont connues du créateur. Par exemple, sous Windows , les données d'environnement source peuvent être le chemin de répertoire de l'objet contenant les données non formatées. Toutefois, si le créateur ne connaît pas les données de l'environnement source, l'exit de message fourni par l'utilisateur doit déterminer les informations d'environnement requises.

La longueur des données d'environnement source est indiquée par *SrcEnvLength*; si cette longueur est égale à zéro, il n'y a pas de données d'environnement source et *SrcEnvOffset* est ignoré. Si elles sont présentes, les données de l'environnement source doivent résider entièrement dans *StrucLength* octets à partir du début de la structure.

Les applications ne doivent pas supposer que les données d'environnement démarrent immédiatement après la dernière zone fixe de la structure ou qu'elles sont contiguës à l'une des données traitées par les zones *SrcNameOffset*, *DestEnvOffset* et *DestNameOffset* .

La valeur initiale de cette zone est 0.

SrcNameLongueur (MQLONG)

Longueur du nom de l'objet source. Si cette zone est à zéro, il n'y a pas de nom d'objet source et *SrcNameOffset* est ignoré.

La valeur initiale de cette zone est 0.

SrcNameDécalage (MQLONG)

Cette zone indique le décalage du nom d'objet source à partir du début de la structure MQRMH. Le nom de l'objet source peut être spécifié par le créateur du message de référence, si ces données sont connues du créateur. Toutefois, si le créateur ne connaît pas le nom de l'objet source, l'exit de message fourni par l'utilisateur doit identifier l'objet à accéder.

La longueur du nom de l'objet source est donnée par *SrcNameLength*; si cette longueur est égale à zéro, il n'y a pas de nom d'objet source et *SrcNameOffset* est ignoré. S'il est présent, le nom de l'objet source doit se trouver complètement dans *StrucLength* octets à partir du début de la structure.

Les applications ne doivent pas supposer que le nom de l'objet source est contigu à l'une des données traitées par les zones *SrcEnvOffset*, *DestEnvOffset*et *DestNameOffset* .

La valeur initiale de cette zone est 0.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID_STRUC_MQRMH_
Identificateur de la structure d'en-tête de message de référence.

Pour le langage de programmation C, la constante MQRMH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQRMH_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

La valeur initiale de cette zone est MQRMH_STRUC_ID.

StrucLength (MQLONG)
Longueur totale de MQRMH, y compris les chaînes à la fin des zones fixes, mais pas les données non formatées.

La valeur initiale de cette zone est zéro.

Version (MQLONG)

Numéro de version de la structure. La valeur doit être:

MQRMH_VERSION_1
La structure d'en-tête de message de référence Version-1 .

La constante suivante indique le numéro de version de la version en cours:

MQRMH_VERSION_CURRENT_VERSION
Version actuelle de la structure d'en-tête de message de référence.

La valeur initiale de cette zone est MQRMH_VERSION_1.

Valeurs initiales et déclarations de langage pour MQRMH

Tableau 542. Valeurs initiales des zones dans MQRMH pour MQRMH

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQRMH_	'RMH¬'
Version	MQRMH_VERSION_1	1
StrucLength	Aucun	0
Encoding	MQENC_NATIVE	Dépend de l'environnement
CodedCharSetId	MQCCSI_UNDEFINI	0
Format	MQFMT_AUCUN	Espaces vides
Flags	MQRMHF_NOT_LAST	0
ObjectType	Aucun	Espaces vides
ObjectInstanceId	MQOII_AUCUN	Valeurs NULL
SrcEnvLength	Aucun	0
SrcEnvOffset	Aucun	0
SrcNameLength	Aucun	0
SrcNameOffset	Aucun	0
DestEnvLength	Aucun	0
DestEnvOffset	Aucun	0
DestNameLength	Aucun	0
DestNameOffset	Aucun	0
DataLogicalLength	Aucun	0
DataLogicalOffset	Aucun	0
DataLogicalOffset2	Aucun	0

Tableau 542. Valeurs initiales des zones dans MQRMH pour MQRMH (suite)

Nom de zone	Nom de la constante	Valeur de la constante

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQRMH_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQRMH MyRMH = {MQRMH_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Enregistrement de réponse

Le tableau suivant récapitule les zones de la structure.

Tableau 543. Zones dans MQRR

Zone	Description	Topic
CompCode	Code achèvement de la file d'attente	CompCode
Reason	Code anomalie de la file d'attente	Motif

Présentation de MQRR

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ connectés à ces systèmes.

Objectif: Utilisez la structure MQRR pour recevoir le code achèvement et le code anomalie résultant de l'opération d'ouverture ou d'insertion pour une file d'attente de destination unique, lorsque la destination est une liste de distribution. MQRR est une structure de sortie pour les appels MQOPEN, MQPUT et MQPUT1 .

Jeu de caractères et codage: les données dans MQRR doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Syntaxe: en fournissant un tableau de ces structures sur les appels MQOPEN et MQPUT ou sur l'appel MQPUT1 , vous pouvez déterminer les codes achèvement et les codes raison de toutes les

files d'attente d'une liste de distribution lorsque le résultat de l'appel est mixte, c'est-à-dire lorsque l'appel aboutit pour certaines files d'attente de la liste mais échoue pour d'autres. Le code anomalie MQRC_MULTIPLE_MOTIFS de l'appel indique que les enregistrements de réponse (s'ils sont fournis par l'application) ont été définis par le gestionnaire de files d'attente.

Zones pour MQRR

La structure MQRR contient les zones suivantes ; les zones sont décrites dans l' **ordre alphabétique**:

CompCode (MQLONG)

Il s'agit du code achèvement résultant de l'opération d'ouverture ou d'insertion pour la file d'attente avec le nom indiqué par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 .

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est MQCC_OK.

Motif (MQLONG)

Il s'agit du code anomalie résultant de l'opération d'ouverture ou d'insertion pour la file d'attente avec le nom qui a été indiqué par l'élément correspondant dans le tableau des structures MQOR fournies sur l'appel MQOPEN ou MQPUT1 .

Il s'agit toujours d'une zone de sortie. La valeur initiale de cette zone est MQRC_NONE.

Valeurs initiales et déclarations de langue pour MQRR

Tableau 544. Valeurs initiales des zones dans MQRR pour MQRR

Nom de zone	Nom de la constante	Valeur de la constante
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0

Remarques :

1. Dans le langage de programmation C, la variable macroMQRR_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQRR MyRR = {MQRR_DEFAULT};
```

Déclaration C

```
typedef struct tagMQRR MQRR;
struct tagMQRR {
  MQLONG  CompCode;  /* Completion code for queue */
  MQLONG  Reason;    /* Reason code for queue */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
dcl
 1 MQRR based,
  3 CompCode fixed bin(31), /* Completion code for queue */
  3 Reason   fixed bin(31); /* Reason code for queue */
```

Déclaration Visual Basic

```
Type MQRR
  CompCode As Long 'Completion code for queue'
  Reason   As Long 'Reason code for queue'
End Type
```

MQSCO-Options de configuration SSL

Le tableau suivant récapitule les zones de la structure.

Tableau 545. Champs dans MQSCO.

Liste des zones dans MQSCO, par version, avec des liens vers les rubriques qui décrivent les zones.

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
KeyRepository	Emplacement du référentiel de clés	KeyRepository
CryptoHardware	Détails du matériel de cryptographie	CryptoHardware
AuthInfoRecCount	Nombre d'enregistrements MQAIR présents	AuthInfoRecCount
AuthInfoRecOffset	Décalage du premier enregistrement MQAIR à partir du début de MQSCO	AuthInfoRecOffset
AuthInfoRecPtr	Adresse du premier enregistrement MQAIR	AuthInfoRecPtr
Remarque : Les deux zones suivantes sont ignorées si *Version* est inférieur à MQSCO_VERSION_2.		
KeyResetCount	Nombre de réinitialisations de clé secrète SSL	KeyReset
FipsRequired	Utiliser des algorithmes de cryptographie certifiés FIPS dans WebSphere MQ	«FipsRequired (MQLONG)», à la page 542
Remarque : La zone suivante est ignorée si *Version* est inférieur à MQSCO_VERSION_3.		
EncryptionPolicySuiteB	Utiliser uniquement les algorithmes de cryptographie Suite B	EncryptionPolicySuiteB
Remarque : La zone suivante est ignorée si *Version* est inférieur à MQSCO_VERSION_4.		
CertificateValPolicy	Règle de validation de certificat	PolitiqueCertificateVal

Référence associée

«MQCNO-Options de connexion», à la page 298
Le tableau suivant récapitule les zones de la structure.

«Présentation de MQSCO», à la page 540
Disponibilité: clients AIX, HP-UX, IBM i, Solaris, Linux et Windows .

«Zones pour MQSCO», à la page 540

«Valeurs initiales et déclarations de langue pour MQSCO», à la page 544

Présentation de MQSCO

Disponibilité: clients AIX, HP-UX, IBM i, Solaris, Linux et Windows .

Objectif: La structure MQSCO (conjointement avec les zones SSL de la structure MQCD) permet à une application exécutée en tant que client WebSphere MQ MQI de spécifier des options de configuration qui contrôlent l'utilisation de SSL pour la connexion client lorsque le protocole de canal est TCP/IP. La structure est un paramètre d'entrée dans l'appel MQCONNX.

Si le protocole de canal du canal client n'est pas TCP/IP, la structure MQSCO est ignorée.

Jeu de caractères et codage: Les données de MQSCO mustis sbe se trouvent dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE.

Zones pour MQSCO

La structure MQSCO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

AuthInfoRecCount (MQLONG)

Il s'agit du nombre d'enregistrements d'informations d'authentification (MQAIR) traités par les zones *AuthInfoRecPtr* ou *AuthInfoRecOffset* . Pour plus d'informations, voir «MQAIR-Enregistrement des informations d'authentification», à la page 253. La valeur doit être supérieure ou égale à zéro. Si la valeur n'est pas valide, l'appel échoue avec le code anomalie MQRC_AUTH_INFO_REC_COUNT_ERROR.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

AuthInfoRecOffset (MQLONG)

Décalage en octets du premier enregistrement d'informations d'authentification à partir du début de la structure MQSCO. Le décalage peut être positif ou négatif. La zone est ignorée si *AuthInfoRecCount* a la valeur zéro.

Vous pouvez utiliser *AuthInfoRecOffset* ou *AuthInfoRecPtr* pour spécifier les enregistrements MQAIR, mais pas les deux ; voir la description de la zone *AuthInfoRecPtr* pour plus de détails.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est 0.

AuthInfoRecPtr (PMQAIR)

Il s'agit de l'adresse du premier enregistrement d'informations d'authentification. La zone est ignorée si *AuthInfoRecCount* a la valeur zéro.

Vous pouvez fournir le tableau d'enregistrements MQAIR de l'une des deux manières suivantes:

- A l'aide de la zone de pointeur *AuthInfoRecPtr*

  Dans ce cas, l'application peut déclarer un tableau d'enregistrements MQAIR distinct de la structure MQSCO et définir *AuthInfoRecPtr* sur l'adresse du tableau.

  Envisagez d'utiliser *AuthInfoRecPtr* pour les langages de programmation qui prennent en charge le type de données de pointeur d'une manière portable dans différents environnements (par exemple, le langage de programmation C).

- En utilisant la zone de décalage *AuthInfoRecOffset*

  Dans ce cas, l'application doit déclarer une structure composée contenant un MQSCO suivi du tableau d'enregistrements MQAIR et définir *AuthInfoRecOffset* sur le décalage du premier enregistrement du tableau à partir du début de la structure MQSCO. Vérifiez que cette valeur est correcte et qu'elle peut être prise en compte dans un MQLONG (le langage de programmation le plus restrictif est COBOL, pour lequel la plage valide est comprise entre -999 999 999 et +999 999 999).

  Envisagez d'utiliser *AuthInfoRecOffset* pour les langages de programmation qui ne prennent pas en charge le type de données de pointeur ou qui implémentent le type de données de pointeur

d'une manière qui n'est pas portable dans des environnements différents (par exemple, le langage de programmation COBOL).

Quelle que soit la technique choisie, vous ne pouvez utiliser qu'une seule des options *AuthInfoRecPtr* et *AuthInfoRecOffset* ; l'appel échoue avec le code anomalie MQRC_AUTH_INFO_REC_ERROR si les deux sont différents de zéro.

Il s'agit d'une zone d'entrée. La valeur initiale de cette zone est le pointeur null dans les langages de programmation qui prennent en charge les pointeurs, et une chaîne d'octets entièrement null dans le cas contraire.

Remarque : Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Règle CertificateVal(MQLONG)

Cette zone indique le type de règle de validation de certificat utilisé. La zone peut être définie sur l'une des valeurs suivantes:

MQ_CERT_VAL_POLICY_ANY
Appliquez chacune des règles de validation de certificat prises en charge par la bibliothèque de sockets sécurisés. Acceptez la chaîne de certificats si l'une des politiques considère que la chaîne de certificats est valide.

MQ_CERT_VAL_POLICY_RFC5280
Appliquez uniquement la règle de validation de certificat RFC5280 . Ce paramètre fournit une validation plus stricte que le paramètre ANY, mais rejette certains certificats numériques plus anciens.

La valeur initiale de cette zone est MQ_CERT_VAL_POLICY_ANY

CryptoHardware (MQCHAR256)
Cette zone fournit des détails de configuration pour le matériel cryptographique connecté au système client.

Définissez la zone sur une chaîne au format suivant, ou laissez-la vide ou null:

```
GSK_PKCS11=<the PKCS #11 driver path and file name>;<the PKCS #11
token label>;<the PKCS #11 token password>;<symmetric cipher setting>;
```

Pour utiliser du matériel cryptographique conforme à l'interface PKCS #11 , par exemple, les chaînes de mot de passe IBM 4960 ou IBM 4764, le chemin du pilote PKCS #11 , le libellé de jeton PKCS #11 et le mot de passe de jeton PKCS #11 doivent être spécifiés, chacun se terminant par un point-virgule.

Le chemin du pilote PKCS #11 est un chemin d'accès absolu à la bibliothèque partagée prenant en charge la carte PKCS #11 . Le nom de fichier du pilote PKCS #11 est le nom de la bibliothèque partagée. Voici un exemple de la valeur requise pour le chemin d'accès et le nom de fichier PKCS #11 :

```
/usr/lib/pkcs11/PKCS11_API.so
```

Le libellé de jeton PKCS #11 doit être entièrement en minuscules. Si vous avez configuré votre matériel avec un libellé de jeton en casse mixte ou en majuscule, reconfigurez-le avec ce libellé en minuscule.

Si aucune configuration de matériel de cryptographie n'est requise, définissez la zone sur une valeur vide ou null.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. Si la valeur n'est pas valide ou entraîne un échec lors de l'utilisation de la configuration du matériel de cryptographie, l'appel échoue avec le code anomalie MQRC_CRYPTO_HARDWARE_ERROR.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ_SSL_CRYPTO_HARDWARE_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.

EncryptionPolicySuiteB(MQLONG)

Cette zone indique si la cryptographie compatible Suite B est utilisée et quel niveau de puissance est utilisé. La valeur peut être une ou plusieurs des valeurs suivantes:

- MQ_SUITE_B_NONE

  La cryptographie conforme à la suite B n'est pas utilisée.
- MQ_SUITE_B_128_BIT

  La sécurité de la suite B 128 bits est utilisée.
- MQ_SUITE_B_192_BIT

  La sécurité de la suite B 192 bits est utilisée.

Remarque : L'utilisation de MQ_SUITE_B_NONE avec une autre valeur de cette zone n'est pas valide.

FipsRequired (MQLONG)
WebSphere MQ peut être configuré avec du matériel de cryptographie de sorte que les modules de cryptographie utilisés soient ceux fournis par le produit matériel ; ils peuvent être certifiés FIPS à un niveau particulier en fonction du produit matériel de cryptographie utilisé. Utilisez cette zone pour indiquer que seuls les algorithmes certifiés FIPS sont utilisés si la cryptographie est fournie dans le logiciel fourni par WebSphere MQ.

Lorsque WebSphere MQ est installé, une implémentation de la cryptographie SSL est également installée, qui fournit des modules certifiés FIPS.

Les valeurs possibles sont les suivantes:

MQSSL_FIPS_NO
Il s'agit de la valeur par défaut. Lorsque cette valeur est définie:

- Tout CipherSpec pris en charge sur une plateforme particulière peut être utilisé.
- Si elles sont exécutées sans matériel de cryptographie, les CipherSpecs suivantes sont exécutées à l'aide de la cryptographie certifiée FIPS 140-2 sur les plateformes WebSphere MQ :
  - TLS_RSA_WITH_3DES_EDE_CBC_SHA
  - TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS_RSA_WITH_AES_256_CBC_SHA

MQSSL_FIPS_YES
Lorsque cette valeur est définie, sauf si vous utilisez du matériel de cryptographie pour effectuer la cryptographie, vous pouvez être certain que

- Seuls les algorithmes de cryptographie certifiés FIPS peuvent être utilisés dans le CipherSpec s'appliquant à cette connexion client.
- Les connexions de canal SSL entrantes et sortantes aboutissent uniquement si l'une des spécifications de chiffrement suivantes est utilisée:
  - TLS_RSA_WITH_3DES_EDE_CBC_SHA
  - TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS_RSA_WITH_AES_256_CBC_SHA

Remarques :

1. CipherSpec TLS_RSA_WITH_3DES_EDE_CBC_SHA est obsolète.
2. Dans la mesure du possible, si des CipherSpecs FIPS uniquement sont configurés, le client MQI rejette les connexions qui spécifient un CipherSpec non FIPS avec MQRC_SSL_INITIALIZATION_ERROR. WebSphere MQ ne garantit pas le rejet de toutes ces connexions et il est de votre responsabilité de déterminer si votre configuration WebSphere MQ est conforme à la norme FIPS.

KeyRepository (MQCHAR256)

Cette zone concerne uniquement les clients WebSphere MQ MQI s'exécutant sur des systèmes UNIX, Linuxet Windows . Il indique l'emplacement du fichier de la base de données de clés dans lequel les clés et les certificats sont stockés. Le fichier de la base de données de clés doit avoir un nom de fichier au format `zzz.kdb` , où `zzz` peut être sélectionné par l'utilisateur. La zone *KeyRepository* contient le chemin d'accès à ce fichier, ainsi que le radical du nom de fichier (tous les caractères du nom de fichier jusqu'au `.kdbfinal`, mais non compris). Le suffixe de fichier `.kdb` est ajouté automatiquement.

Chaque fichier de base de données de clés est associé à un *fichier de mot de passe secret*. Contient les mots de passe codés qui sont utilisés pour autoriser l'accès par programmation à la base de données de clés. Le fichier de mot de passe secret doit résider dans le même répertoire et avoir le même radical de fichier que la base de données de clés, et doit se terminer par le suffixe `.sth`.

Par exemple, si la zone *KeyRepository* a la valeur `/xxx/yyy/key` , le fichier de la base de données de clés doit être `/xxx/yyy/key.kdb` et le fichier de mot de passe secret doit être `/xxx/yyy/key.sth` , où `xxx` et `yyy` représentent les noms de répertoire.

Si la valeur est inférieure à la longueur de la zone, terminez la valeur par un caractère NULL ou remplissez la zone avec des blancs. La valeur n'est pas vérifiée ; en cas d'erreur lors de l'accès au référentiel de clés, l'appel échoue avec le code anomalie MQRC_KEY_REPOSITORY_ERROR.

Pour exécuter une connexion SSL à partir d'un client WebSphere MQ MQI, définissez *KeyRepository* sur un nom de fichier de base de données de clés valide.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ_SSL_KEY_REPOSITORY_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.

KeyReset(MQLONG)
Représente le nombre total d'octets non chiffrés envoyés et reçus dans une conversation SSL ou TLS avant la renégociation de la clé secrète.

Le nombre d'octets inclut les informations de contrôle envoyées par l'agent MCA.

Si vous spécifiez un nombre de réinitialisations de clé confidentielle SSL ou TLS compris entre 1 et 32 Ko, les canaux SSL ou TLS utiliseront un nombre de réinitialisations de clé confidentielle de 32 Ko. Cela permet d'éviter le coût de traitement d'un nombre excessif de réinitialisations de clé qui se produiraient pour des valeurs de réinitialisation de clé secrète SSL ou TLS de petite taille.

Il s'agit d'une zone d'entrée. La valeur est un nombre compris entre 0 et 999 999 999, avec une valeur par défaut de 0. Utilisez la valeur 0 pour indiquer que les clés secrètes ne sont jamais renégociées.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID_STRUC_MQSCO_
Identificateur de la structure des options de configuration SSL.

Pour le langage de programmation C, la constante MQSCO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQSCO_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSCO_STRUC_ID.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQSCO_VERSION_1
Structure des options de configuration SSL Version-1 .

MQSCO_VERSION_2
Structure des options de configuration SSL Version-2 .

MQSCO_VERSION_3
Structure des options de configuration SSL Version-3 .

MQSCO_VERSION_4
Structure des options de configuration SSL Version-4 .

La constante suivante indique le numéro de version de la version en cours:

VERSION MQSCO_CURRENT_VERSION
Version actuelle de la structure des options de configuration SSL.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSCO_VERSION_1.

Valeurs initiales et déclarations de langue pour MQSCO

Tableau 546. Valeurs initiales des champs dans MQSCO.

Description des zones dans MQSCO et leurs valeurs initiales

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	MQSCO_STRUC_ID	'SCO¬'
Version	MQSCO_CURRENT_VERSION	1
KeyRepository	Aucun	Chaîne nulle ou blancs
CryptoHardware	Aucun	Chaîne nulle ou blancs
AuthInfoRecCount	Aucun	0
AuthInfoRecOffset	Aucun	0
AuthInfoRecPtr	Aucun	Pointeur null ou octets null
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};
```

Déclaration C

```
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 */
```

Déclaration COBOL

```
**   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
```

Déclaration PL/I

```
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 */
```

Déclaration Visual Basic

```
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 - Descripteur d'abonnement

Le tableau suivant récapitule les zones de la structure.

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options	Options
ObjectName	Nom d'objet	ObjectName
AlternateUserId	Autre ID utilisateur	AlternateUserid
AlternateSecurityId	Autre ID de sécurité	AlternateSecurityId
SubExpiry	Expiration de l'abonnement	SubExpiry
ObjectString	Chaîne d'objet	ObjectString
SubName	Nom abonnement	SubName
SubUserData	Données utilisateur d'abonnement	SubUserData
SubCorrelId	ID corrélation d'abonnement	SubCorrelId
PubPriority	Priorité de publication	PubPriority
PubAccountingToken	Jeton de comptabilité de publication	JetonPubAccounting
PubAppIdentityData	Données d'identité de l'application de publication	PubAppIdentityData
SelectionString	Chaîne fournissant des critères de sélection	SelectionString
SubLevel	Niveau d'abonnement	SubLevel
ResObjectString	Nom d'objet long	ResObjectChaîne

Présentation de MQSD

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OS, plus les clients WebSphere MQ MQI connectés à ces systèmes.

Objectif: La structure MQSD est utilisée pour spécifier les détails de l'abonnement effectué.

La structure est un paramètre d'entrée-sortie dans l'appel MQSUB. Pour plus d'informations, voir les remarques sur l'utilisation de MQSUB.

Abonnements gérés: si une application n'a pas besoin d'utiliser une file d'attente particulière comme destination pour les publications qui correspondent à son abonnement, elle peut utiliser la fonction d'abonnement géré. Si une application choisit d'utiliser un abonnement géré, le gestionnaire de files

d'attente informe l'abonné de la destination où les messages publiés sont envoyés, en fournissant un descripteur d'objet comme sortie de l'appel MQSUB. Pour plus d'informations, voir Hobj (MQHOBJ)-input/output.

Lorsque l'abonnement est supprimé, le gestionnaire de files d'attente s'engage également à nettoyer les messages qui n'ont pas été extraits de la destination gérée, dans les cas suivants:

- Lorsque l'abonnement est supprimé-à l'aide de MQCLOSE avec MQCO_REMOVE_SUB-et que le Hobj géré est fermé.
- Par des moyens implicites lorsque la connexion est perdue à une application à l'aide d'un abonnement non durable (MQSO_NON_DURABLE)
- Par expiration lorsqu'un abonnement est supprimé car il est arrivé à expiration et que le Hobj géré est fermé.

Vous devez utiliser des abonnements gérés avec des abonnements non durables afin que ce nettoyage puisse se produire et que les messages des abonnements non durables fermés n'occupent pas d'espace dans votre gestionnaire de files d'attente. Les abonnements durables peuvent également utiliser des destinations gérées.

Version: la version actuelle de MQSD est MQSD_VERSION_1.

Jeu de caractères et codage: Les données de MQSD doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Zones pour MQSD

La structure MQSD contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

ID AlternateSecurity(MQBYTE40)

Il s'agit d'un identificateur de sécurité qui est transmis avec l'ID AlternateUserau service d'autorisation pour permettre l'exécution des vérifications d'autorisation appropriées.

L'ID AlternateSecurityest utilisé uniquement si MQSO_ALTERNATE_USER_AUTHORITY est spécifié et que la zone d'ID AlternateUsern'est pas entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone.

En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette zone n'est pas modifiée.

Pour plus d'informations, voir la description de «ID AlternateSecurity(MQBYTE40)», à la page 458 dans le type de données MQOD.

ID AlternateUser(MQCHAR12)

Si vous spécifiez MQSO_ALTERNATE_USER_AUTHORITY, cette zone contient un autre identificateur utilisateur qui est utilisé pour vérifier l'autorisation pour l'abonnement et la sortie dans la file d'attente de destination (spécifiée dans le paramètre *Hobj* de l'appel MQSUB), à la place de l'identificateur utilisateur sous lequel l'application est en cours d'exécution.

Si l'opération aboutit, l'ID utilisateur indiqué dans cette zone est enregistré en tant qu'ID utilisateur propriétaire de l'abonnement à la place de l'ID utilisateur sous lequel l'application est en cours d'exécution.

Si MQSO_ALTERNATE_USER_AUTHORITY est spécifié et que cette zone est entièrement vide jusqu'au premier caractère null ou jusqu'à la fin de la zone, l'abonnement peut aboutir uniquement si aucune autorisation utilisateur n'est nécessaire pour s'abonner à cette rubrique avec les options spécifiées ou la file d'attente de destination pour la sortie.

Si MQSO_ALTERNATE_USER_AUTHORITY n'est pas spécifié, cette zone est ignorée.

Les différences suivantes existent dans les environnements indiqués:

- Sous z/OS, seuls les 8 premiers caractères de l'ID AlternateUsersont utilisés pour vérifier l'autorisation de l'abonnement. Toutefois, l'ID utilisateur en cours doit être autorisé à indiquer cet ID utilisateur de remplacement particulier ; les 12 caractères de l'ID utilisateur de remplacement sont utilisés pour cette vérification. L'ID utilisateur ne doit contenir que des caractères autorisés par le gestionnaire de sécurité externe.

En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette zone n'est pas modifiée.

Il s'agit d'une zone d'entrée. La longueur de cette zone est indiquée par MQ_USER_ID_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 12 caractères blancs dans les autres langages de programmation.

ObjectName (MQCHAR48)

Il s'agit du nom de l'objet de rubrique tel qu'il est défini sur le gestionnaire de files d'attente local.

Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (_), pourcentage (%)

Le nom ne doit pas contenir d'espaces de début ou d'espaces imbriqués, mais peut contenir des espaces de fin. Utilisez un caractère null pour indiquer la fin des données significatives dans le nom ; la valeur null et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent dans les environnements indiqués:

- Sur les systèmes qui utilisent EBCDIC Katakana, les caractères minuscules ne peuvent pas être utilisés.
- Sous z/OS:
  - Evitez les noms qui commencent ou se terminent par un trait de soulignement ; ils ne peuvent pas être traités par les panneaux d'opérations et de contrôle.
  - Le caractère de pourcentage a une signification spéciale pour RACF. Si RACF est utilisé comme gestionnaire de sécurité externe, les noms ne doivent pas contenir le pourcentage. Dans ce cas, ces noms ne sont pas inclus dans les contrôles de sécurité lorsque des profils génériques RACF sont utilisés.
- Sous IBM i, les noms contenant des caractères minuscules, des barres obliques ou des pourcentages doivent être placés entre guillemets lorsqu'ils sont spécifiés dans les commandes. Ces guillemets ne doivent pas être indiqués pour les noms qui apparaissent en tant que zones dans les structures ou en tant que paramètres dans les appels.

ObjectName est utilisé pour former le nom complet de la rubrique.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*. Pour plus de détails sur l'utilisation de ces deux zones, voir «Utilisation de chaînes de rubrique», à la page 562.

Si l'objet identifié par la zone *ObjectName* est introuvable, l'appel échoue avec le code anomalie MQRC_UNKNOWN_OBJECT_NAME même si une chaîne est spécifiée dans *ObjectString*.

En cas de retour d'un appel MQSUB à l'aide de l'option MQSO_RESUME, cette zone est inchangée.

La longueur de cette zone est indiquée par MQ_TOPIC_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, le nom de l'objet de rubrique auquel vous êtes abonné ne peut pas être modifié. Cette zone et la zone *ObjectString* peuvent être omises. S'ils sont fournis, ils doivent être résolus avec le même nom de rubrique complet. Si ce n'est pas le cas, l'appel échoue avec MQRC_TOPIC_NOT_ALTERABLE.

ObjectString (MQCHARV)

Il s'agit du nom d'objet long à utiliser.

ObjectString est utilisé pour former le nom de rubrique complet.

Le nom complet de la rubrique peut être généré à partir de deux zones différentes: *ObjectName* et *ObjectString*. Pour plus de détails sur l'utilisation de ces deux zones, voir «Utilisation de chaînes de rubrique», à la page 562.

La longueur maximale de *ObjectString* est 10240.

Si *ObjectString* n'est pas spécifié correctement, conformément à la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC_OBJECT_STRING_ERROR.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

S'il existe des caractères génériques dans *ObjectString* , l'interprétation de ces caractères génériques peut être contrôlée à l'aide des options de caractères génériques spécifiées dans la zone Options du MQSD.

En cas de retour d'un appel MQSUB à l'aide de l'option MQSO_RESUME, cette zone est inchangée. Le nom de rubrique complet utilisé est renvoyé dans la zone *ResObjectString* si une mémoire tampon est fournie.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, le nom long de l'objet de rubrique auquel vous êtes abonné ne peut pas être modifié. Cette zone et la zone *ObjectName* peuvent être omises. S'ils sont fournis, ils doivent être résolus avec le même nom de rubrique complet ou l'appel échoue avec MQRC_TOPIC_NOT_ALTERABLE.

Options (MQLONG)
Fournit des options permettant de contrôler l'action de l'appel MQSUB.

Vous devez spécifier au moins l'une des options suivantes:

- MQSO_ALTER
- MQSO_RESUME
- MQSO_CREER

Les valeurs que vous spécifiez pour les options peuvent être utilisées comme suit:

- Les valeurs peuvent être ajoutées. N'ajoutez pas la même constante plus d'une fois.
- Les valeurs peuvent être combinées à l'aide de l'opération OR bit à bit, si le langage de programmation prend en charge les opérations bit à bit.

Les combinaisons qui ne sont pas valides sont indiquées dans cette rubrique ; les autres combinaisons sont valides.

Options d'accès ou de création: les options d'accès et de création déterminent si un abonnement est créé ou si un abonnement existant est renvoyé ou modifié. Vous devez spécifier au moins l'une de ces options. Le tableau affiche les combinaisons valides d'options d'accès et de création.

Combinaison d'options	**Remarques**
MQSO_CREER	Crée un abonnement s'il n'en existe pas. Cette combinaison échoue si l'abonnement existe.
MQSO_RESUME	Reprend un abonnement existant. Cette combinaison échoue si aucun abonnement n'existe.

Combinaison d'options	Remarques
MQSO_CREATE + MQSO_RESUME	Crée un abonnement s'il n'en existe pas et en reprend un correspondant, s'il existe. Cette combinaison est utile lorsqu'elle est utilisée dans une application qui est exécutée plusieurs fois.
MQSO_ALTER (voir la remarque)	Reprend un abonnement existant, en modifiant toutes les zones pour qu'elles correspondent à celles spécifiées dans le MQSD. Cette combinaison échoue si aucun abonnement n'existe.
MQSO_CREATE + MQSO_ALTER (voir la remarque)	Crée un abonnement s'il n'en existe pas et reprend un abonnement correspondant, s'il existe, en modifiant les zones pour qu'elles correspondent à celles spécifiées dans le MQSD. Cette combinaison est utile lorsqu'elle est utilisée dans une application qui souhaite s'assurer que son abonnement est dans un certain état avant de continuer.
Remarque : Les options spécifiant MQSO_ALTER peuvent également spécifier MQSO_RESUME, mais cette combinaison n'a pas d'effet supplémentaire sur la spécification de MQSO_ALTER uniquement. MQSO_ALTER implique MQSO_RESUME, car l'appel de MQSUB pour modifier un abonnement implique que l'abonnement sera également repris. Le contraire n'est pas vrai, cependant: la reprise d'un abonnement n'implique pas qu'il doit être modifié.	

MQSO_CREER

Créez un nouvel abonnement pour la rubrique spécifiée. Si un abonnement utilisant le même *SubName* existe, l'appel échoue avec MQRC_SUB_ALREADY_EXISTS. Cet échec peut être évité en combinant l'option MQSO_CREATE avec MQSO_RESUME. Le *SubName* n'est pas toujours nécessaire. Pour plus de détails, voir la description de cette zone.

La combinaison de MQSO_CREATE avec MQSO_RESUME renvoie un descripteur à un abonnement préexistant pour le *SubName* spécifié s'il en existe un ; s'il n'existe pas d'abonnement existant, un nouveau descripteur est créé à l'aide de toutes les zones fournies dans le MQSD.

MQSO_CREATE peut également être combiné avec MQSO_ALTER pour obtenir un effet similaire.

MQSO_RESUME

Renvoie un descripteur à un abonnement préexistant qui correspond à celui spécifié par *SubName*. Aucune modification n'est apportée aux attributs d'abonnement correspondants et ils sont renvoyés en sortie dans la structure MQSD. Seules les zones MQSD suivantes sont utilisées: StrucId, Version, Options, AlternateUserId et AlternateSecurityId et SubName.

L'appel échoue avec le code anomalie MQRC_NO_SUBSCRIPTION si un abonnement ne correspond pas au nom d'abonnement complet. Cet échec peut être évité en combinant l'option MQSO_CREATE avec MQSO_RESUME.

L'ID utilisateur de l'abonnement est l'ID utilisateur qui a créé l'abonnement, ou s'il a été modifié ultérieurement par un autre ID utilisateur, il s'agit de l'ID utilisateur de la modification ayant abouti la plus récente. Si un ID AlternateUserest utilisé et que l'utilisation d'ID utilisateur de remplacement est autorisée pour cet utilisateur, l'ID utilisateur de remplacement est enregistré en tant qu'ID utilisateur qui a créé l'abonnement à la place de l'ID utilisateur sous lequel l'abonnement a été effectué.

S'il existe un abonnement correspondant qui a été créé sans l'option MQSO_ANY_USERID et que l'ID utilisateur de l'abonnement est différent de celui de l'application demandant un descripteur à l'abonnement, l'appel échoue avec le code anomalie MQRC_IDENTITY_MISMATCH.

Si un abonnement correspondant existe et est en cours d'utilisation, l'appel échoue avec MQRC_SUBSCRIPTION_IN_USE.

Si l'abonnement nommé dans SubName n'est pas un abonnement valide à reprendre ou à modifier à partir d'une application, l'appel échoue avec MQRC_INVALID_SUBSCRIPTION.

MQSO_RESUME étant implicite dans MQSO_ALTER, vous n'avez pas besoin de le combiner avec cette option. Toutefois, la combinaison des deux options ne génère pas d'erreur.

MQSO_ALTER

Renvoie un descripteur à un abonnement préexistant dont le nom d'abonnement complet correspond à celui spécifié par le nom dans *SubName*. Tous les attributs de l'abonnement qui sont différents de ceux spécifiés dans le MQSD sont modifiés dans l'abonnement sauf si la modification est interdite pour cet attribut. Les détails sont indiqués dans la description de chaque attribut et sont résumés dans le tableau suivant. Si vous tentez de modifier un attribut qui ne peut pas être modifié ou de modifier un abonnement qui a défini l'option MQSO_IMMUTABLE, l'appel échoue avec le code anomalie indiqué dans le tableau suivant.

L'appel échoue avec le code anomalie MQRC_NO_SUBSCRIPTION si un abonnement correspondant au nom d'abonnement complet n'existe pas. Vous pouvez éviter cet échec en combinant l'option MQSO_CREATE avec MQSO_ALTER.

La combinaison de MQSO_CREATE et de MQSO_ALTER renvoie un descripteur à un abonnement préexistant pour le *SubName* spécifié s'il en existe un ; s'il n'existe pas d'abonnement existant, un nouveau descripteur est créé à l'aide de toutes les zones fournies dans le MQSD.

L'ID utilisateur de l'abonnement est l'ID utilisateur qui a créé l'abonnement, ou s'il est modifié ultérieurement par un autre ID utilisateur, il s'agit de l'ID utilisateur de la modification la plus récente ayant abouti. Si un ID AlternateUserest utilisé et que l'utilisation d'ID utilisateur de remplacement est autorisée pour cet utilisateur, l'ID utilisateur de remplacement est enregistré en tant qu'ID utilisateur qui a créé l'abonnement à la place de l'ID utilisateur sous lequel l'abonnement a été effectué.

S'il existe un abonnement correspondant qui a été créé sans l'option MQSO_ANY_USERID et que l'ID utilisateur de l'abonnement est différent de celui de l'application demandant un descripteur à l'abonnement, l'appel échoue avec le code anomalie MQRC_IDENTITY_MISMATCH.

Si un abonnement correspondant existe et est en cours d'utilisation, l'appel échoue avec MQRC_SUBSCRIPTION_IN_USE.

Si l'abonnement nommé dans SubName n'est pas un abonnement valide à reprendre ou à modifier à partir d'une application, l'appel échoue avec MQRC_INVALID_SUBSCRIPTION.

Le tableau suivant montre la capacité de MQSO_ALTER à modifier les valeurs d'attribut dans MQSD et MQSUB.

Tableau 547. Attributs dans MQSD et MQSUB pouvant être modifiés

Descripteur de type de données ou appel de fonction	Nom de zone	Cet attribut peut-il être modifié à l'aide de MQSO_ALTER	Code anomalie
Disque MQSD	Options de durabilité	Non	MQRC_DURATIONITY_NOT_ALTERABLE
Disque MQSD	Options de destination	Oui	Aucun
Disque MQSD	Options d'enregistrement	Oui (voir la remarque «1», à la page 552)	MQRC_GROUPING_NOT_ALTERABLE si vous tentez de modifier MQSO_GROUP_SUB
Disque MQSD	Options de publication	Oui (voir la remarque «2», à la page 552)	Aucun
Disque MQSD	Options de caractère générique	Non	MQRC_TOPIC_NOT_ALTERABLE
Disque MQSD	Autres options	Non (voir la remarque «3», à la page 552)	Aucun
Disque MQSD	ObjectName	Non	MQRC_TOPIC_NOT_ALTERABLE
Disque MQSD	AlternateUserid	Non (voir la remarque «4», à la page 552)	Aucun

Tableau 547. Attributs dans MQSD et MQSUB pouvant être modifiés (suite)

Descripteur de type de données ou appel de fonction	Nom de zone	Cet attribut peut-il être modifié à l'aide de MQSO_ALTER	Code anomalie
Disque MQSD	AlternateSecurityId	Non (voir la remarque «4», à la page 552)	Aucun
Disque MQSD	SubExpiry	Oui	Aucun
Disque MQSD	ObjectString	Non	MQRC_TOPIC_NOT_ALTERABLE
Disque MQSD	SubName	Non (voir la remarque «5», à la page 552)	Aucun
Disque MQSD	SubUserData	Oui	Aucun
Disque MQSD	SubCorrelId	Oui (voir la remarque «6», à la page 552)	MQRC_GROUPING_NOT_ALTERABLE dans un abonnement groupé
Disque MQSD	PubPriority	Oui	Aucun
Disque MQSD	Jeton PubAccounting	Oui	Aucun
Disque MQSD	PubApplIdentityData	Oui	Aucun
Disque MQSD	SubLevel	Non	MQRC_SUBLEVEL_NOT_ALTERABLE
MQSUB	HOBJ	Oui (voir la remarque «6», à la page 552)	MQRC_GROUPING_NOT_ALTERABLE dans un abonnement groupé

Remarques :

1. MQSO_GROUP_SUB ne peut pas être modifié.
2. MQSO_NEW_PUBLICATIONS_ONLY ne peut pas être modifié car il ne fait pas partie de l'abonnement
3. Ces options ne font pas partie de l'abonnement
4. Cet attribut ne fait pas partie de l'abonnement
5. Cet attribut correspond à l'identité de l'abonnement en cours de modification
6. Modifiable sauf lorsqu'il fait partie d'un sous-groupe (MQSO_GROUP_SUB)

Options de durabilité: Les options suivantes contrôlent la durabilité de l'abonnement. Vous ne pouvez spécifier qu'une seule de ces options. Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, vous ne pouvez pas modifier la durabilité de l'abonnement. En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, l'option de durabilité appropriée est définie.

MQSO_DURABLE
Demandez que l'abonnement à cette rubrique soit conservé jusqu'à ce qu'elle soit explicitement supprimée à l'aide de MQCLOSE avec l'option MQCO_REMOVE_SUB. Si cet abonnement n'est pas explicitement supprimé, il sera conservé même après la fermeture de cette connexion d'application au gestionnaire de files d'attente.

Si un abonnement durable est demandé à une rubrique définie comme n'autorisant pas les abonnements durables, l'appel échoue avec MQRC_DURABILITY_NOT_ALLOWED.

MQSO_NON_DURABLE
Demandez la suppression de l'abonnement à cette rubrique lorsque la connexion des applications au gestionnaire de files d'attente est fermée, si elle n'est pas déjà explicitement supprimée. MQSO_NON_DURABLE est l'opposé de l'option MQSO_DURABLE et est défini pour aider la documentation du programme. Il s'agit de la valeur par défaut si aucune n'est spécifiée.

Options de destination: L'option suivante contrôle la destination à laquelle les publications d'une rubrique à laquelle vous avez souscrit sont envoyées. Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, la destination utilisée pour les publications de l'abonnement peut être modifiée. En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette option est définie si nécessaire.

MQSO_GERE

Demande que la destination à laquelle les publications sont envoyées soit gérée par le gestionnaire de files d'attente.

L'identificateur d'objet renvoyé dans *Hobj* représente une file d'attente gérée par le gestionnaire de files d'attente et doit être utilisé avec les appels MQGET, MQCB, MQINQ ou MQCLOSE ultérieurs.

Un descripteur d'objet renvoyé par un appel MQSUB précédent ne peut pas être fourni dans le paramètre *Hobj* lorsque MQSO_MANAGED n'est pas spécifié.

MQSO_NO_MULTICAST

Demandez que la destination à laquelle les publications sont envoyées ne soit pas une adresse de groupe de multidiffusion. Cette option est valide uniquement lorsqu'elle est associée à l'option MQSO_MANAGED. Lorsqu'un descripteur d'une file d'attente est fourni dans le paramètre *Hobj* , la multidiffusion ne peut pas être utilisée pour cet abonnement et l'option n'est pas valide.

Si la rubrique est définie pour n'autoriser que les abonnements multidiffusion, à l'aide du paramètre `MCAST(ONLY)` , l'appel échoue avec le code anomalie MQRC_MULTICAST_REQUIRED.

Option de portée: L'option suivante contrôle la portée de l'abonnement effectué. Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, cette option de portée d'abonnement ne peut pas être modifiée. Lors du retour d'un appel MQSUB à l'aide de MQSO-RESUME, l'option de portée appropriée est définie.

MQSO_SCOPE_QMGR

Cet abonnement est effectué uniquement sur le gestionnaire de files d'attente local. Aucun abonnement de proxy n'est distribué aux autres gestionnaires de files d'attente du réseau. Seules les publications publiées sur ce gestionnaire de files d'attente sont envoyées à cet abonné. Cette option remplace tout comportement défini à l'aide de l'attribut de rubrique SUBSCOPE.

Remarque : S'il n'est pas défini, la portée de l'abonnement est déterminée par l'attribut de rubrique SUBSCOPE.

Options d'enregistrement: Les options suivantes contrôlent les détails de l'enregistrement effectué auprès du gestionnaire de files d'attente pour cet abonnement. Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, ces options d'enregistrement peuvent être modifiées. En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, les options d'enregistrement appropriées sont définies.

SOUS-GROUPE_MQSO

Cet abonnement doit être regroupé avec d'autres abonnements du même SubLevel à l'aide de la même file d'attente et en spécifiant le même ID de corrélation de sorte que toute publication destinée à des rubriques qui entraînerait la fourniture de plusieurs messages de publication au groupe d'abonnements, en raison du chevauchement d'un ensemble de chaînes de rubrique utilisées, entraîne la distribution d'un seul message à la file d'attente. Si cette option n'est pas utilisée, chaque abonnement unique (identifié par SubName) qui correspond est fourni avec une copie de la publication, ce qui peut signifier que plusieurs copies de la publication peuvent être placées dans la file d'attente partagée par un certain nombre d'abonnements.

Seul l'abonnement le plus important du groupe est fourni avec une copie de la publication. L'abonnement le plus significatif est basé sur le nom de rubrique complet jusqu'au point où un caractère générique est trouvé. Si une combinaison de schémas génériques est utilisée dans le groupe, seule la position du caractère générique est importante. Il est conseillé de ne pas combiner différents schémas de caractères génériques dans un groupe d'abonnements partageant la même file d'attente.

Lors de la création d'un abonnement groupé, il doit toujours avoir un SubNameunique, mais s'il correspond au nom de rubrique complet d'un abonnement existant dans le groupe, l'appel échoue avec MQRC_DUPLICATE_GROUP_SUB.

Si l'abonnement le plus significatif dans le groupe spécifie également MQSO_NOT_OWN_PUBS et qu'il s'agit d'une publication provenant de la même application, aucune publication n'est distribuée dans la file d'attente.

Lors de la modification d'un abonnement effectué avec cette option, les zones qui impliquent le regroupement, Hobj sur l'appel MQSUB (représentant le nom de la file d'attente et du gestionnaire de files d'attente) et l'ID SubCorrelne peuvent pas être modifiées. La tentative de modification entraîne l'échec de l'appel avec MQRC_GROUPING_NOT_ALTERABLE.

Cette option doit être combinée avec MQSO_SET_CORREL_ID avec un ID SubCorrelqui n'est pas défini sur MQCI_NONE et ne peut pas être combinée avec MQSO_MANAGED.

ID utilisateur MQSO_ANY_USERID

Lorsque MQSO_ANY_USERID est spécifié, l'identité de l'abonné n'est pas limitée à un seul ID utilisateur. Cela permet à tout utilisateur de modifier ou de reprendre l'abonnement lorsqu'il dispose des droits appropriés. Un seul utilisateur peut disposer de l'abonnement à la fois. Une tentative de reprise de l'utilisation d'un abonnement actuellement utilisé par une autre application entraîne l'échec de l'appel avec MQRC_SUBSCRIPTION_IN_USE.

Pour ajouter cette option à un abonnement existant, l'appel MQSUB (à l'aide de MQSO_ALTER) doit provenir du même ID utilisateur que l'abonnement d'origine lui-même.

Si un appel MQSUB fait référence à un abonnement existant avec MQSO_ANY_USERID défini et que l'ID utilisateur est différent de l'abonnement d'origine, l'appel aboutit uniquement si le nouvel ID utilisateur est autorisé à s'abonner à la rubrique. Une fois l'opération terminée, les futures publications destinées à cet abonné sont placées dans la file d'attente des abonnés avec le nouvel ID utilisateur défini dans le message de publication.

N'indiquez pas à la fois MQSO_ANY_USERID et MQSO_FIXED_USERID. Si aucun des deux n'est spécifié, la valeur par défaut est MQSO_FIXED_USERID.

ID_UTILISATEUR_FIXE_MQSO_FIXE

Lorsque MQSO_FIXED_USERID est spécifié, l'abonnement peut être modifié ou repris uniquement par le dernier ID utilisateur pour modifier l'abonnement. Si l'abonnement n'a pas été modifié, il s'agit de l'ID utilisateur qui a créé l'abonnement.

Si une instruction MQSUB fait référence à un abonnement existant avec MQSO_ANY_USERID défini et modifie l'abonnement à l'aide de MQSO_ALTER pour utiliser l'option MQSO_FIXED_USERID, l'ID utilisateur de l'abonnement est désormais corrigé au niveau de ce nouvel ID utilisateur. L'appel aboutit uniquement si le nouvel ID utilisateur est autorisé à s'abonner à la rubrique.

Si un ID utilisateur autre que celui enregistré comme propriétaire d'un abonnement tente de reprendre ou de modifier un abonnement MQSO_FIXED_USERID, l'appel échoue avec MQRC_IDENTITY_MISMATCH. L'ID utilisateur propriétaire d'un abonnement peut être affiché à l'aide de la commande DISPLAY SBSTATUS.

N'indiquez pas à la fois MQSO_ANY_USERID et MQSO_FIXED_USERID. Si aucun des deux n'est spécifié, la valeur par défaut est MQSO_FIXED_USERID.

Options de publication: Les options suivantes contrôlent la manière dont les publications sont envoyées à cet abonné. Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, vous pouvez modifier ces options de publication.

MQSO_NON_PUBS
Indique au courtier que l'application ne souhaite pas voir ses propres publications. Les publications sont considérées comme provenant de la même application si les descripteurs de connexion sont identiques. En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette option est définie si nécessaire.

MQSO_NEW_PUBLICATIONS_ONLY
Aucune publication actuellement conservée ne doit être envoyée, lorsque cet abonnement est créé, uniquement les nouvelles publications. Cette option s'applique uniquement lorsque MQSO_CREATE est spécifié. Les modifications apportées ultérieurement à un abonnement ne modifient pas le flux des publications et, par conséquent, les publications conservées sur une rubrique ont déjà été envoyées à l'abonné en tant que nouvelles publications.

Si cette option est spécifiée sans MQSO_CREATE, l'appel échoue avec MQRC_OPTIONS_ERROR. En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette option n'est pas définie même si l'abonnement a été créé à l'aide de cette option.

Si cette option n'est pas utilisée, les messages précédemment conservés sont envoyés à la file d'attente de destination fournie. Si cette action échoue en raison d'une erreur, MQRC_RETAINED_MSG_Q_ERROR ou MQRC_RETAINED_NOT_LIVRÉES, la création de l'abonnement échoue.

MQSO_PUBLICATIONS_ON_REQUEST
La définition de cette option indique que l'abonné demandera des informations spécifiquement lorsque cela est nécessaire. Le gestionnaire de files d'attente n'envoie pas de messages non sollicités à l'abonné. La publication conservée (ou éventuellement plusieurs publications si un caractère générique est spécifié dans la rubrique) est envoyée à l'abonné chaque fois qu'un appel MQSUBRQ est effectué à l'aide du descripteur Hsub d'un appel MQSUB précédent. Aucune publication n'est envoyée suite à l'appel MQSUB utilisant cette option. En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette option est définie si nécessaire.

Cette option n'est pas valide avec un SubLevel supérieur à 1.

Options de lecture anticipée: Les options suivantes contrôlent si les messages non persistants sont envoyés à une application avant l'application qui les demande.

MQSO_READ_AHEAD_AS_Q_DEF
Si l'appel MQSUB utilise un descripteur géré, l'attribut de lecture anticipée par défaut de la file d'attente modèle associée à la rubrique à laquelle l'utilisateur est abonné détermine si les messages sont envoyés à l'application avant que l'application ne les demande.

Il s'agit de la valeur par défaut.

MQSO_NO_READ_AHEAD
Si l'appel MQSUB utilise un descripteur géré, les messages ne sont pas envoyés à l'application avant que celle-ci ne les demande.

MQSO_READ_AHEAD
Si l'appel MQSUB utilise un descripteur géré, des messages peuvent être envoyés à l'application avant que celle-ci ne les demande.

Remarque :

Les remarques suivantes s'appliquent aux options de lecture anticipée:

1. Une seule de ces options peut être spécifiée. Si MQOO_READ_AHEAD et MQOO_NO_READ_AHEAD sont spécifiées, le code anomalie MQRC_OPTIONS_ERROR est renvoyé. Ces options ne sont applicables que si MQSO_MANAGED est spécifié.
2. Elles ne s'appliquent pas à MQSUB lorsqu'une file d'attente qui a été précédemment ouverte est transmise. Il se peut que la lecture anticipée ne soit pas activée lorsqu'elle est demandée. Les options MQGET utilisées lors du premier appel MQGET peuvent empêcher l'activation de la lecture anticipée. En outre, la lecture anticipée est désactivée lorsque le client se connecte à un gestionnaire de files d'attente où la lecture anticipée n'est pas prise en charge. Si l'application n'est pas exécutée en tant que client WebSphere MQ , ces options sont ignorées.

Options de caractères génériques: Les options suivantes contrôlent la façon dont les caractères génériques sont interprétés dans la chaîne fournie dans la zone ObjectString du MQSD. Vous ne pouvez spécifier qu'une seule de ces options. Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, ces options génériques ne peuvent pas être modifiées. En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, l'option de caractère générique appropriée est définie.

MQSO_WILDCARD_CHAR
Les caractères génériques ne s'appliquent qu'aux caractères de la chaîne de rubrique.

Le comportement défini par MQSO_WILDCARD_CHAR est illustré dans le tableau suivant.

Caractère spécial	Comportement
la barre oblique (/)	Pas de signification, juste un autre caractère
Astérisque (*)	Caractère générique, zéro ou plusieurs caractères
Point d'interrogation (?)	Caractère générique, 1 caractère

Caractère spécial	Comportement
le symbole de pourcentage (%)	Caractère d'échappement permettant aux caractères (*), (?) ou (%) d'être utilisés dans une chaîne et de ne pas être interprétés comme un caractère spécial, par exemple (% *), (%?) ou (%%).

Par exemple, la publication sur la rubrique suivante:

```
/level0/level1/level2/level3/level4
```

correspond aux abonnés à l'aide des rubriques suivantes:

```
*
/*
/ level0/level1/level2/level3/*
/ level0/level1/*/level3/level4
/ level0/level1/le?el2/level3/level4
```

Remarque : Cette utilisation de caractères génériques fournit exactement la signification fournie dans WebSphere MQ V6 et WebSphere MB V6 lors de l'utilisation de messages formatés MQRFH1 pour la publication / l'abonnement. Il est recommandé de ne pas l'utiliser pour les applications nouvellement écrites et de l'utiliser uniquement pour les applications qui étaient précédemment exécutées sur cette version et qui n'ont pas été modifiées pour utiliser le comportement de caractère générique par défaut, comme décrit dans MQSO_WILDCARD_TOPIC.

MQSO_WILDCARD_TOPIC

Les caractères génériques n'agissent que sur les éléments de rubrique dans la chaîne de rubrique. Il s'agit du comportement par défaut si aucun n'est sélectionné.

Le comportement requis par MQSO_WILDCARD_TOPIC est présenté dans le tableau suivant:

Caractère spécial	Comportement
(/)	Séparateur de niveau de rubrique
Signe dièse (#)	Caractère générique: plusieurs niveaux de rubrique
Signe plus (+)	Caractère générique: niveau de rubrique unique
Remarques : Les caractères (+) et (#) ne sont pas traités comme des caractères génériques s'ils sont mélangés à d'autres caractères (y compris eux-mêmes) dans un niveau de rubrique. Dans la chaîne suivante, les caractères (#) et (+) sont traités comme des caractères ordinaires. `level0/level1/#+/level3/level#`	

Par exemple, la publication sur la rubrique suivante:

```
/level0/level1/level2/level3/level4
```

correspond aux abonnés à l'aide des rubriques suivantes:

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1/+/level3/level4
```

Remarque : Cette utilisation de caractères génériques fournit la signification fournie dans WebSphere Message Broker version 6 lors de l'utilisation de messages formatés MQRFH2 pour la publication / l'abonnement.

Autres options: Les options suivantes contrôlent la manière dont l'appel d'API est émis et non l'abonnement. En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, ces options restent inchangées. Pour plus d'informations, voir «ID AlternateUser(MQCHAR12)», à la page 547.

MQSO_ALTERNATE_USER_AUTHORITY

La zone ID AlternateUsercontient un identificateur utilisateur à utiliser pour valider cet appel MQSUB. L'appel peut aboutir uniquement si cet ID AlternateUserest autorisé à ouvrir l'objet avec les options d'accès spécifiées, que l'ID utilisateur sous lequel l'application s'exécute soit ou non autorisé à le faire.

ID_CORREL_SET_MQSO

L'abonnement doit utiliser l'identificateur de corrélation fourni dans la zone *SubCorrelId* . Si cette option n'est pas spécifiée, un identificateur de corrélation est automatiquement créé par le gestionnaire de files d'attente au moment de l'abonnement et renvoyé à l'application dans la zone *SubCorrelId* . Pour plus d'informations, voir «SubCorrelId (MQBYTE24)», à la page 560 .

Cette option ne peut pas être combinée avec MQSO_MANAGED.

MQSO_SET_IDENTITY_CONTEXT

L'abonnement doit utiliser le jeton de comptabilité et les données d'identité d'application fournies dans les zones *PubAccountingToken* et *PubApplIdentityData* .

Si cette option est spécifiée, la même vérification d'autorisation est effectuée comme si la file d'attente de destination était accessible à l'aide d'un appel MQOPEN avec MQOO_SET_IDENTITY_CONTEXT, sauf dans le cas où l'option MQSO_MANAGED est également utilisée, auquel cas aucune vérification d'autorisation n'est effectuée sur la file d'attente de destination.

Si cette option n'est pas spécifiée, les informations de contexte par défaut sont associées aux publications envoyées à cet abonné comme suit:

Zone dans MQMD	Valeur utilisée
UserIdentifier	ID utilisateur associé à l'abonnement au moment où l'abonnement a été effectué.
AccountingToken	Déterminé à partir de l'environnement si possible ; défini sur MQACT_NONE si ce n'est pas le cas.
ApplIdentityData	Mettre à blanc

Cette option est valide uniquement avec MQSO_CREATE et MQSO_ALTER. Si elle est utilisée avec MQSO_RESUME, les zones *PubAccountingToken* et *PubApplIdentityData* sont ignorées, de sorte que cette option n'a aucun effet.

Si un abonnement est modifié sans utiliser cette option alors que les informations de contexte d'identité fournies par l'abonnement étaient précédemment fournies, des informations de contexte par défaut sont générées pour l'abonnement modifié.

Si un abonnement permettant à différents ID utilisateur de l'utiliser avec l'option MQSO_ANY_USERID, est repris par un autre ID utilisateur, le contexte d'identité par défaut est généré pour le nouvel ID utilisateur propriétaire de l'abonnement et toutes les publications suivantes sont distribuées contenant le nouveau contexte d'identité.

MQSO_FAIL_IF_QUIESCING

L'appel MQSUB échoue si le gestionnaire de files d'attente est à l'état de mise au repos. Sous z/OS, pour une application CICS ou IMS , cette option force également l'appel MQSUB à échouer si la connexion est à l'état de mise au repos.

Jeton PubAccounting(MQBYTE32)

Il s'agit de la valeur qui sera dans la zone *AccountingToken* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. *AccountingToken* fait partie du contexte d'identité du message. Pour plus d'informations sur le contexte de message, voir Contexte de message. Pour plus d'informations sur la zone *AccountingToken* dans le MQMD, voir «AccountingToken (MQBYTE32)», à la page 398

Vous pouvez utiliser la valeur spéciale suivante pour la zone *PubAccountingToken* :

MQACT_AUCUN

Aucun jeton de comptabilité n'est spécifié.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQACT_NONE_ARRAY est également définie ; elle a la même valeur que MQACT_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Si l'option MQSO_SET_IDENTITY_CONTEXT n'est pas spécifiée, le jeton de comptabilité est généré par le gestionnaire de files d'attente en tant qu'informations de contexte par défaut et cette zone est une zone de sortie qui contient le *AccountingToken* qui sera défini dans chaque message publié pour cet abonnement.

Si l'option MQSO_SET_IDENTITY_CONTEXT est spécifiée, le jeton de comptabilité est généré par l'utilisateur et cette zone est une zone d'entrée qui contient le *AccountingToken* à définir dans chaque publication pour cet abonnement.

La longueur de cette zone est indiquée par MQ_ACCOUNTING_TOKEN_LENGTH. La valeur initiale de cette zone est MQACT_NONE.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, la valeur de *AccountingToken* dans les messages de publication ultérieurs peut être modifiée.

En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette zone est définie sur le *AccountingToken* en cours utilisé pour l'abonnement.

PubApplIdentityData (MQCHAR32)

Il s'agit de la valeur qui se trouve dans la zone *ApplIdentityData* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. *ApplIdentityData* fait partie du contexte d'identité du message. Pour plus d'informations sur le contexte de message, voir Contexte de message. Pour plus d'informations sur la zone *ApplIdentityData* dans le MQMD, voir «Données ApplIdentity(MQCHAR32)», à la page 400

Si l'option MQSO_SET_IDENTITY_CONTEXT n'est pas spécifiée, le *ApplIdentityData* défini dans chaque message publié pour cet abonnement est vide, en tant qu'informations de contexte par défaut.

Si l'option MQSO_SET_IDENTITY_CONTEXT est spécifiée, *PubApplIdentityData* est généré par l'utilisateur et cette zone est une zone d'entrée qui contient le *ApplIdentityData* à définir dans chaque publication pour cet abonnement.

La longueur de cette zone est indiquée par MQ_APPL_IDENTITY_DATA_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 32 caractères blancs dans les autres langages de programmation.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, le *ApplIdentityData* des messages de publication ultérieurs peut être modifié.

En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette zone est définie sur le *ApplIdentityData* en cours utilisé pour l'abonnement.

PubPriority (MQLONG)

Il s'agit de la valeur qui sera dans la zone *Priority* du descripteur de message (MQMD) de tous les messages de publication correspondant à cet abonnement. Pour plus d'informations sur la zone *Priority* dans le MQMD, voir «Priorité (MQLONG)», à la page 425.

La valeur doit être supérieure ou égale à zéro ; zéro est la priorité la plus basse. Les valeurs spéciales suivantes peuvent également être utilisées:

MQPRI_PRIORITY_AS_Q_DEF

Lorsqu'une file d'attente d'abonnement est fournie dans la zone *Hobj* de l'appel MQSUB et qu'elle n'est pas un descripteur géré, la priorité du message est extraite de l'attribut *DefPriority* de cette file d'attente. Si la file d'attente est une file d'attente de cluster ou qu'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité est déterminée lorsque le message de publication est inséré dans la file d'attente, comme décrit pour «Priorité (MQLONG)», à la page 425.

Si l'appel MQSUB utilise un descripteur géré, la priorité du message est extraite de l'attribut *DefPriority* de la file d'attente modèle associée à la rubrique à laquelle il est abonné.

MQPRI_PRIORITY_AS_PUBLISHED

La priorité du message est la priorité de la publication d'origine. Il s'agit de la valeur initiale de la zone.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, le *Priority* des messages de publication ultérieurs peut être modifié.

En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette zone est définie sur la priorité en cours utilisée pour l'abonnement.

Chaîne ResObject(MQCHARV)
Il s'agit du nom d'objet long une fois que le gestionnaire de files d'attente a résolu le nom fourni dans *ObjectName*.

Si le nom d'objet long est fourni dans *ObjectString* et que rien n'est fourni dans *ObjectName*, la valeur renvoyée dans cette zone est la même que celle fournie dans *ObjectString*.

Si cette zone est omise (c'est-à-dire que ResObjectString.VSBufSize est égal à zéro), *ResObjectString* n'est pas renvoyé, mais la longueur est renvoyée dans ResObjectString.VSLength. Si la longueur est inférieure à la chaîne ResObjectcomplète, elle est tronquée et renvoie autant de caractères les plus à droite que la longueur indiquée.

Si *ResObjectString* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC_RES_OBJECT_STRING_ERROR.

SelectionString (MQCHARV)
Il s'agit de la chaîne utilisée pour fournir les critères de sélection utilisés lors de l'abonnement aux messages d'une rubrique.

Cette zone de longueur variable est renvoyée en sortie d'un appel MQSUB à l'aide de l'option MQSO_RESUME, si une mémoire tampon est fournie, ainsi qu'une longueur de mémoire tampon positive dans VSBufSize. Si aucune mémoire tampon n'est fournie sur l'appel, seule la longueur de la chaîne de sélection est renvoyée dans la zone VSLength de MQCHARV. Si la mémoire tampon fournie est inférieure à l'espace requis pour renvoyer la zone, seuls les octets VSBufSize sont renvoyés dans la mémoire tampon fournie.

Si *SelectionString* est spécifié de manière incorrecte, conformément à la description du mode d'utilisation de la structure «MQCHARV-Chaîne de longueur variable», à la page 276 , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC_SELECTION_STRING_ERROR.

L'utilisation de SelectionString est décrite dans Sélecteurs .

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID_STRUCD_MQS
Identificateur de la structure du descripteur d'abonnement.

Pour le langage de programmation C, la constante MQSD_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQSD_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSD_STRUC_ID.

SubCorrelId (MQBYTE24)
Cette zone contient un identificateur de corrélation commun à toutes les publications correspondant à cet abonnement.

Avertissement : un identificateur de corrélation ne peut être transmis qu'entre des gestionnaires de files d'attente dans un cluster de publication / abonnement et non une hiérarchie.

Toutes les publications envoyées pour correspondre à cet abonnement contiennent cet identificateur de corrélation dans le descripteur de message. Si plusieurs abonnements obtiennent leurs publications de la même file d'attente, l'utilisation de MQGET par identificateur de corrélation permet d'obtenir uniquement les publications d'un abonnement spécifique. Cet identificateur de corrélation peut être généré par le gestionnaire de files d'attente ou par l'utilisateur.

Si l'option MQSO_SET_CORREL_ID n'est pas spécifiée, l'identificateur de corrélation est généré par le gestionnaire de files d'attente et cette zone est une zone de sortie contenant l'identificateur de corrélation qui sera défini dans chaque message publié pour cet abonnement. L'identificateur de corrélation généré se compose d'un identificateur de produit de 4 octets (AMQX ou CSQM en ASCII ou EBCDIC) suivi d'une implémentation spécifique du produit d'une chaîne unique.

Si l'option MQSO_SET_CORREL_ID est spécifiée, l'identificateur de corrélation est généré par l'utilisateur et cette zone est une zone d'entrée contenant l'identificateur de corrélation à définir dans chaque publication pour cet abonnement. Dans ce cas, si la zone contient MQCI_NONE, l'identificateur de corrélation défini dans chaque message publié pour cet abonnement est l'identificateur de corrélation créé par l'insertion d'origine du message.

Si l'option MQSO_GROUP_SUB est spécifiée et que l'identificateur de corrélation spécifié est identique à un abonnement groupé existant utilisant la même file d'attente et une chaîne de rubrique se chevauchant, seul l'abonnement le plus significatif du groupe est fourni avec une copie de la publication.

La longueur de cette zone est indiquée par MQ_CORREL_ID_LENGTH. La valeur initiale de cette zone est MQCI_NONE.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER et que cette zone est une zone d'entrée, l'identificateur de corrélation d'abonnement peut être modifié, sauf s'il s'agit d'un abonnement groupé, c'est-à-dire qu'il a été créé à l'aide de l'option MQSO_GROUP_SUB, auquel cas l'identificateur de corrélation d'abonnement ne peut pas être modifié.

En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette zone est définie sur l'identificateur de corrélation en cours de l'abonnement.

SubExpiry (MQLONG)

Il s'agit du temps exprimé en dixièmes de seconde après lequel l'abonnement expire. Aucune autre publication ne correspondra à cet abonnement une fois cet intervalle écoulé. Dès qu'un abonnement arrive à expiration, les publications ne sont plus envoyées à la file d'attente. Toutefois, les publications qui existent déjà ne sont en aucun cas affectées. *SubExpiry* n'a aucun effet sur l'expiration de la publication.

La valeur spéciale suivante est reconnue:

MQEI_UNLIMITED
L'abonnement a un délai d'expiration illimité.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, l'expiration de l'abonnement peut être modifiée.

En cas de retour d'un appel MQSUB à l'aide de l'option MQSO_RESUME, cette zone est définie sur l'expiration d'origine de l'abonnement et non sur le délai d'expiration restant.

SubLevel (MQLONG)
Niveau associé à l'abonnement. Les publications ne sont distribuées à cet abonnement que si elles se trouvent dans l'ensemble des abonnements dont la valeur SubLevel la plus élevée est inférieure ou égale à la valeur PubLevel utilisée au moment de la publication. Toutefois, si une publication a été conservée, elle n'est plus disponible pour les abonnés de niveau supérieur car elle est republiée sur le site PubLevel 1.

La valeur doit être comprise entre zéro et 9. Zéro est la valeur la plus faible.

La valeur initiale de cette zone est 1.

Pour plus d'informations, voir Intercepting publications .

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, le SubLevel ne peut pas être modifié.

La combinaison d'un SubLevel avec une valeur supérieure à 1 avec l'option MQSO_PUBLICATIONS_ON_REQUEST n'est pas autorisée.

En cas de retour d'un appel MQSUB à l'aide de MQSO_RESUME, cette zone est définie sur le niveau en cours utilisé pour l'abonnement.

Données SubUser(MQCHARV)

Indique les données utilisateur de l'abonnement. Les données fournies sur l'abonnement dans cette zone seront incluses en tant que propriété de message de données MQSubUserde chaque publication envoyée à cet abonnement.

La longueur maximale de *SubUserData* est 10240.

Si *SubUserData* est spécifié de manière incorrecte, conformément à la description de l'utilisation de la structure MQCHARV , ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC_SUB_USER_DATA_ERROR.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, les données utilisateur de l'abonnement peuvent être modifiées.

Cette zone de longueur variable est renvoyée en sortie d'un appel MQSUB à l'aide de l'option MQSO_RESUME, si une mémoire tampon est fournie et qu'il existe une longueur de mémoire tampon positive dans *VSBufLen*. Si aucune mémoire tampon n'est fournie sur l'appel, seule la longueur de la date utilisateur de l'abonnement est renvoyée dans la zone *VSLength* de MQCHARV. Si la mémoire tampon fournie est inférieure à l'espace requis pour renvoyer la zone, seuls *VSBufLen* octets sont renvoyés dans la mémoire tampon fournie.

SubName (MQCHARV)

Indique le nom de l'abonnement. Cette zone est requise uniquement si *Options* spécifie l'option MQSO_DURABLE, mais si elle est fournie, elle sera également utilisée par le gestionnaire de files d'attente pour MQSO_NON_DURABLE.

S'il est spécifié, *SubName* doit être unique dans le gestionnaire de files d'attente, car il s'agit de la méthode utilisée pour identifier l'abonnement.

La longueur maximale de *SubName* est 10240.

Cette zone a deux objectifs. Pour un abonnement MQSO_DURABLE, vous utilisez cette zone pour identifier un abonnement afin de pouvoir le reprendre après sa création si vous avez fermé l'identificateur de l'abonnement (à l'aide de l'option MQCO_KEEP_SUB) ou si vous avez été déconnecté du gestionnaire de files d'attente. Cette opération est effectuée à l'aide de l'appel MQSUB avec l'option MQSO_RESUME. Il est également affiché dans la vue d'administration des abonnements dans la zone SUBNAME de DISPLAY SBSTATUS.

Si *SubName* est spécifié de manière incorrecte, selon la description de la façon d'utiliser la structure MQCHARV , elle est laissée de côté lorsqu'elle est requise (c'est-à-dire *SubName.VSLength* est égal à zéro), ou s'il dépasse la longueur maximale, l'appel échoue avec le code anomalie MQRC_SUB_NAME_ERROR.

Il s'agit d'une zone d'entrée. Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQCHARV.

Si vous modifiez un abonnement existant à l'aide de l'option MQSO_ALTER, le nom de l'abonnement ne peut pas être modifié car il s'agit de la zone d'identification utilisée pour rechercher l'abonnement référencé. Il n'est pas modifié dans la sortie d'un appel MQSUB avec l'option MQSO_RESUME.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQSD_VERSION_1
Structure de descripteur d'abonnement Version-1 .

La constante suivante indique le numéro de version de la version en cours:

MQSD_CURRENT_VERSION
Version actuelle de la structure du descripteur d'abonnement.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSD_VERSION_1.

Utilisation de chaînes de rubrique

Une rubrique est construite à partir de la sous-rubrique identifiée dans un objet de rubrique et d'une sous-rubrique fournie par une application. Vous pouvez utiliser l'une ou l'autre des sous-rubriques comme nom de rubrique ou les combiner pour former un nouveau nom de rubrique.

Dans un programme MQI, le nom de rubrique complet est créé par MQOPEN. Il est composé de deux zones utilisées dans des appels MQI de publication/abonnement, dans l'ordre indiqué :

1. L'attribut **TOPICSTR** de l'objet de rubrique, nommé dans la zone **ObjectName**.
2. Le paramètre **ObjectString** définissant la sous-rubrique fournie par l'application.

La chaîne de rubrique résultante est renvoyée dans le paramètre **ResObjectString**.

Ces zones sont considérées comme étant présentes si le premier caractère de chaque zone n'est pas vide ou un caractère nul et si la longueur de la zone est supérieure à zéro. Si seule une zone est présente, elle est utilisée telle quelle comme nom de rubrique. Si aucune des zones n'a de valeur, l'appel échoue avec le code anomalie MQRC_UNKNOWN_OBJECT_NAME ou MQRC_TOPIC_STRING_ERROR si le nom complet de la rubrique n'est pas valide.

Si les deux zones sont présentes, un caractère'/'est inséré entre les deux éléments du nom de rubrique combiné résultant.

Le Tableau 548, à la page 562 illustre des exemples de concaténation de chaînes de rubrique :

Tableau 548. Exemples de concaténation de chaînes de rubrique

TOPICSTR	**ObjectString**	**Nom complet de la rubrique**	**Commentaire**
Football/Scores	' '	Football/Scores	Le TOPICSTR est utilisé seul

Tableau 548. Exemples de concaténation de chaînes de rubrique (suite)			
TOPICSTR	**ObjectString**	**Nom complet de la rubrique**	**Commentaire**
' '	Football/Scores	Football/Scores	ObjectString est utilisé seul
Football	Scores	Football/Scores	Un caractère'/' est ajouté au point de concaténation
Football	/Scores	Football//Scores	Un'noeud vide'est produit entre les deux chaînes
/Football	Scores	/Football/Scores	La rubrique commence par un'noeud vide'

Le caractère'/'est considéré comme un caractère spécial, fournissant une structure au nom de rubrique complet dans Arborescences de rubriques , et ne doit pas être utilisé pour une autre raison car la structure de l'arborescence de rubriques est affectée. La rubrique "`/Football`" n'est pas la même que la rubrique "`Football`".

Les caractères génériques suivants sont des caractères spéciaux :

- signe plus'+'
- signe dièse'#'
- astérisque'*'
- point d'interrogation'?'

Ces caractères ne sont pas considérés comme non valides, mais vous devez vous assurer de comprendre comment ils sont utilisés. Vous pouvez préférer ne pas utiliser ces caractères dans vos chaînes de rubrique lors de la publication. La publication sur une chaîne de rubrique avec'#'ou'+'mélangée avec d'autres caractères (y compris eux-mêmes) au sein d'un niveau de rubrique, peut être souscrite avec l'un ou l'autre des schémas de caractères génériques. La publication sur une chaîne de rubrique avec'#'ou'+'comme seul caractère entre deux caractères'/'génère une chaîne de rubrique à laquelle une application ne peut pas s'abonner explicitement à l'aide du schéma de caractères génériques MQSO_WILDCARD_TOPIC. Il en résulte que l'application reçoit davantage de publications que prévu.

Exemple de fragment de code

Ce fragment de code, extrait de l'exemple de programme Exemple 2: Diffuseur de publications vers une rubrique variable, combine un objet de rubrique avec une chaîne de rubrique variable.

```
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);
```

Valeurs initiales et déclarations de langue pour MQSD

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUCD_MQS	'SD¬¬'
Version	MQSD_VERSION_1	1

Nom de zone	Nom de la constante	Valeur de la constante
Options	MQSO_NON_DURABLE	0
ObjectName	Aucun	Chaîne nulle ou blancs
AlternateUserId	Aucun	Chaîne nulle ou blancs
AlternateSecurityId	MQSID_AUCUN	Valeurs NULL
SubExpiry	MQEI_UNLIMITED	-1
ObjectString	Aucun	Noms et valeurs définis pour MQCHARV
SubName	Aucun	Noms et valeurs définis pour MQCHARV
SubUserData	Aucun	Noms et valeurs définis pour MQCHARV
SubCorrelId	MQCI_NONE	Valeurs NULL
PubPriority	MQPRI_PRIORITY_AS_Q_DEF	-3
PubAccountingToken	MQACT_AUCUN	Valeurs NULL
PubApplIdentityData	Aucun	Chaîne nulle ou blancs
Selection String	Aucun	Noms et valeurs définis pour MQCHARV
SubLevel	Aucun	1
ResObjectString	Aucun	Noms et valeurs définis pour MQCHARV

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQSD_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQSD MySD = {MQSD_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
** 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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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-Définition des options de propriété de message

Le tableau suivant récapitule les zones de la structure.

Tableau 549. Zones dans MQSMPO

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options	Options
ValueEncoding	Codage de la valeur de propriété	ValueEncoding
ValueCCSID	Jeu de caractères de la valeur de propriété	ValueCCSID

Présentation de MQSMPO

Disponibilité: Tous les systèmes WebSphere MQ et les clients WebSphere MQ .

Objectif: La structure **MQSMPO** permet aux applications de spécifier des options qui contrôlent la manière dont les propriétés des messages sont définies. La structure est un paramètre d'entrée dans l'appel **MQSETMP** .

Jeu de caractères et codage: les données dans **MQSMPO** doivent être dans le jeu de caractères de l'application et le codage de l'application (**MQENC_NATIVE**).

Zones pour MQSMPO

La structure MQSMPO contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

Options (MQLONG)

Options d'emplacement: Les options suivantes se rapportent à l'emplacement relatif de la propriété par rapport au curseur de propriété:

MQSMPO_SET_FIRST
Définit la valeur de la première propriété qui correspond au nom spécifié ou, si elle n'existe pas, ajoute une nouvelle propriété après toutes les autres propriétés avec une hiérarchie correspondante.

MQSMPO_SET_PROP_UNDER_CURSOR
Définit la valeur de la propriété indiquée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois à l'aide de l'option MQIMPO_INQ_FIRST ou MQIMPO_INQ_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé sur un appel MQGET ou lorsque le descripteur de message est spécifié dans la zone *MsgHandle* de la structure MQGMO ou MQPMO sur un appel MQPUT.

Si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi ou si le pointeur de propriété du curseur de propriété a été supprimé, l'appel échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_PROPERTY_NOT_AVAILABLE.

MQSMPO_SET_PROP_BEFORE_CURSOR
Définit une nouvelle propriété avant la propriété pointée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois à l'aide de l'option MQIMPO_INQ_FIRST ou MQIMPO_INQ_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé sur un appel MQGET ou lorsque le descripteur de message est spécifié dans la zone *MsgHandle* de la structure MQGMO ou MQPMO sur un appel MQPUT.

Si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi ou si le pointeur de propriété du curseur de propriété a été supprimé, l'appel échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_PROPERTY_NOT_AVAILABLE.

MQSMPO_SET_PROP_AFTER_CURSOR
Définit une nouvelle propriété après la propriété pointée par le curseur de propriété. La propriété indiquée par le curseur de propriété est celle qui a été interrogée pour la dernière fois à l'aide de l'option MQIMPO_INQ_FIRST ou MQIMPO_INQ_NEXT.

Le curseur de propriété est réinitialisé lorsque le descripteur de message est réutilisé sur un appel MQGET ou lorsque le descripteur de message est spécifié dans la zone *MsgHandle* de la structure MQGMO ou MQPMO sur un appel MQPUT.

Si cette option est utilisée alors que le curseur de propriété n'a pas encore été établi ou si le pointeur de propriété du curseur de propriété a été supprimé, l'appel échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_PROPERTY_NOT_AVAILABLE.

MQSMPO_XX_ENCODE_CASE_ONE annexes
Entraîne l'ajout d'une nouvelle propriété après toutes les autres propriétés avec une hiérarchie correspondante. S'il existe au moins une propriété correspondant au nom spécifié, une nouvelle propriété est ajoutée à la fin après la fin de cette liste de propriétés.

Cette option permet de créer une liste de propriétés portant le même nom.

Si vous n'avez besoin d'aucune des options décrites, utilisez l'option suivante:

MQSMPO_NONE
Aucune option spécifiée.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSMPO_SET_FIRST.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID_STRUC_MQSMPO_

Identificateur de la structure des options de définition des propriétés de message.

Pour le langage de programmation C, la constante **MQSMPO_STRUC_ID_ARRAY** est également définie ; elle a la même valeur que **MQSMPO_STRUC_ID**, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQSMPO_STRUC_ID**.

ValueCCSID (MQLONG)

Jeu de caractères de la valeur de propriété à définir si la valeur est une chaîne de caractères.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQCCSI_APPL**.

ValueEncoding (MQLONG)

Codage de la valeur de propriété à définir si la valeur est numérique.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQENC_NATIVE**.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQSMPO_VERSION_1
Version-1 définit la structure des options de propriété de message.

La constante suivante indique le numéro de version de la version en cours:

MQSMPO_CURRENT_VERSION
Version actuelle de la structure des options de définition des propriétés de message.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est **MQSMPO_VERSION_1**.

Valeurs initiales et déclarations de langage pour MQSMPO

Tableau 550. Valeurs initiales des zones dans MQSMPO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQSMPO_	'SMPO'
Version	MQSMPO_VERSION_1	1
Options	MQSMPO_NONE	0
ValueEncoding	MQENC_NATIVE	Dépend de l'environnement
ValueCCSID	MQCCSI_APPL	-3

Remarques :

1. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
2. Dans le langage de programmation C, la variable macroMQSMPO_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQSMPO MySMPO = {MQSMPO_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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-Options de demande d'abonnement

Cette section décrit les options de demande d'abonnement, les zones qu'elle contient et les valeurs initiales de ces zones.

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
Options	Options	Options
NumPubs	Nombre de publications	NumPubs

Présentation de MQSRO

Disponibilité: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OS plus WebSphere MQ clients MQI connectés à ces systèmes.

Objectif: La structure MQSRO permet à l'application de spécifier des options qui contrôlent la manière dont une demande d'abonnement est effectuée. La structure est un paramètre d'entrée-sortie dans l'appel MQSUBRQ.

Version: la version actuelle de MQSRO est MQSRO_VERSION_1.

Jeu de caractères et codage: les données dans MQSRO doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE. Toutefois, si l'application s'exécute en tant que client MQ MQI, la structure doit être dans le jeu de caractères et le codage du client.

Zones pour MQSRO

La structure MQSRO contient les zones suivantes ; les zones sont décrites par ordre alphabétique:

NumPubs (MQLONG)

Il s'agit d'une zone de sortie renvoyée à l'application pour indiquer le nombre de publications envoyées à la file d'attente d'abonnement à la suite de cet appel. Bien que ce nombre de publications ait été envoyé à la suite de cet appel, rien ne garantit que ce nombre de messages sera disponible pour l'application, en particulier s'il s'agit de messages non persistants.

Il peut y avoir plusieurs publications si la rubrique abonnée contient un caractère générique. Si aucun caractère générique n'était présent dans la chaîne de rubrique lors de la création de l'abonnement représenté par *Hsub* , une seule publication au maximum est envoyée suite à cet appel.

Options (MQLONG)

L'une des options suivantes doit être spécifiée. Vous ne pouvez indiquer qu'une seule option.

MQSRO_FAIL_IF_QUIESCING

L'appel MQSUBRQ échoue si le gestionnaire de files d'attente est à l'état de mise au repos. Sous z/OS, pour une application CICS ou IMS , cette option force également l'échec de l'appel MQSUBRQ si la connexion est à l'état de mise au repos.

Option par défaut: Si l'option décrite ci-dessus n'est pas requise, l'option suivante doit être utilisée:

MQSRO_AUCUN

Utilisez cette valeur pour indiquer qu'aucune autre option n'a été spécifiée. Toutes les options sont définies sur leur valeur par défaut.

MQSRO_NONE aide à la documentation du programme. Bien que cette option ne soit pas destinée à être utilisée avec une autre option, car sa valeur est zéro, cette utilisation ne peut pas être détectée.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure ; la valeur doit être:

ID_STRUC_MQS
Identificateur de la structure des options de demande d'abonnement.

Pour le langage de programmation C, la constante MQSRO_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQSRO_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSRO_STRUC_ID.

Version (MQLONG)

Il s'agit du numéro de version de la structure ; la valeur doit être:

MQSRO_VERSION_1
Structure des options de demande d'abonnement Version-1 .

La constante suivante indique le numéro de version de la version en cours:

VERSION_MQSRO_CURRENT_VERSION
Version actuelle de la structure des options de demande d'abonnement.

Il s'agit toujours d'une zone d'entrée. La valeur initiale de cette zone est MQSRO_VERSION_1.

Valeurs initiales et déclarations de langage pour MQSRO

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQS	'SRO¬'
Version	MQSRO_VERSION_1	1
Options	MQSRO_AUCUN	0
NumPubs	Aucun	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQSRO_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQSRO MySRO = {MQSRO_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
** 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.
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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-Structure de génération de rapports de statut

Le tableau suivant récapitule les zones de la structure.

Tableau 551. Zones dans MQSTS

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
CompCode	Code achèvement de la première erreur	CompCode
Reason	Code anomalie de la première erreur	Motif
PutSuccessCount	Nombre d'appels d'insertion asynchrone réussis	SuccessCount
PutWarningCount	Nombre d'appels d'insertion asynchrones ayant généré des avertissements	WarningCount
PutFailureCount	Nombre d'appels d'insertion asynchrones ayant échoué	FailureCount
ObjectType	Type de l'objet défaillant	ObjectType
ObjectName	Nom de l'objet défaillant	ObjectName
ObjectQMgrName	Nom du gestionnaire de files d'attente propriétaire de l'objet défaillant	ObjectQMgrName
ResolvedObjectName	Nom résolu de la file d'attente de destination	ResolvedObjectNom
ResolvedQMgrName	Nom résolu du gestionnaire de files d'attente de destination	ResolvedQMgrNom
Remarque : Les zones restantes sont ignorées si la version est inférieure à MQSTS_VERSION_2.		
ObjectString	Nom d'objet long de l'objet défaillant	ObjectString
SubName	Nom de l'abonnement défaillant	SubName
OpenOptions	Options d'ouverture associées à l'échec	OpenOptions
SubOptions	Options d'abonnement associées à l'échec	SubOptions

Présentation de MQSTS

Objectif: La structure MQSTS est un paramètre de sortie de la commande MQSTAT.

Jeu de caractères et codage: les données de type caractère dans MQSTS se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; cet attribut est fourni par l'attribut de gestionnaire de files d'attente *CodedCharSetId* . Les données numériques dans MQSTS sont dans le codage machine natif ; elles sont fournies par *Encoding*.

Syntaxe: La commande MQSTAT permet d'extraire des informations de statut. Ces informations sont renvoyées dans une structure MQSTS. Pour plus d'informations sur MQSTAT, voir «MQSTAT-Extraction des informations de statut», à la page 773.

Zones pour MQSTS

La structure MQSTS contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

CompCode (MQLONG)
Code achèvement de l'opération faisant l'objet d'un rapport.

L'interprétation de CompCode dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Il s'agit du code achèvement résultant d'une opération d'insertion asynchrone précédente sur l'objet spécifié dans ObjectName.

MQSTAT_TYPE_RECONNEXION

Si la connexion se reconnecte ou n'a pas pu se reconnecter, il s'agit du code achèvement qui a provoqué le début de la reconnexion.

Si la connexion est actuellement connectée, la valeur est MQCC_OK.

MQSTAT_TYPE_RECONNECTION_ERROR

Si la reconnexion a échoué, il s'agit du code achèvement qui a provoqué l'échec de la reconnexion.

Si la connexion est actuellement connectée ou reconnectée, la valeur est MQCC_OK.

CompCode est toujours un champ de sortie. Sa valeur initiale est MQCC_OK.

ObjectName (MQCHAR48)
Nom de l'objet faisant l'objet du rapport.

L'interprétation de ObjectName dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Il s'agit du nom de la file d'attente ou de la rubrique utilisée dans l'opération d'insertion, dont l'échec est signalé dans les zones *CompCode* et *Reason* de la structure MQSTS .

MQSTAT_TYPE_RECONNEXION

Si la connexion est reconnectée, il s'agit du nom du gestionnaire de files d'attente associé à la connexion.

MQSTAT_TYPE_RECONNECTION_ERROR

Si la reconnexion a échoué, il s'agit du nom de l'objet qui a provoqué l'échec de la reconnexion. La raison de l'échec est indiquée dans les zones *CompCode* et *Reason* de la structure MQSTS .

ObjectName est un champ de sortie. Sa valeur initiale est la chaîne nulle en C et 48 caractères blancs dans les autres langages de programmation.

ObjectQMgrNom (MQCHAR48)
Nom du gestionnaire de files d'attente faisant l'objet d'un rapport.

L'interprétation de ObjectQMgrName dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Il s'agit du nom du gestionnaire de files d'attente sur lequel l'objet *ObjectName* est défini. Un nom entièrement vide jusqu'au premier caractère NULL ou jusqu'à la fin de la zone indique le gestionnaire de files d'attente auquel l'application est connectée (le gestionnaire de files d'attente local).

MQSTAT_TYPE_RECONNEXION

Non renseigné.

MQSTAT_TYPE_RECONNECTION_ERROR

Si la reconnexion a échoué, il s'agit du nom de l'objet qui a provoqué l'échec de la reconnexion. La raison de l'échec est indiquée dans les zones *CompCode* et *Reason* de la structure MQSTS .

ObjectQMgrName est un champ de sortie. Sa valeur est la chaîne nulle en C et 48 caractères blancs dans les autres langages de programmation.

ObjectString (MQCHARV)
Nom d'objet long de l'objet défaillant signalé. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

L'interprétation de ObjectString dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Il s'agit du nom d'objet long de la file d'attente ou de la rubrique utilisée dans l'opération MQPUT , qui a échoué.

MQSTAT_TYPE_RECONNEXION

Chaîne de longueur zéro

MQSTAT_TYPE_RECONNECTION_ERROR

Il s'agit du nom d'objet long de l'objet qui a provoqué l'échec de la reconnexion.

ObjectString est un champ de sortie. Sa valeur initiale est une chaîne de longueur nulle.

ObjectType (MQLONG)
Type de l'objet nommé dans *ObjectName* faisant l'objet d'un rapport.

Les valeurs possibles de ObjectType sont répertoriées dans «MQOT_ * (types d'objet et types d'objet étendu)», à la page 147.

ObjectType est un champ de sortie. Sa valeur initiale est MQOT_Q.

OpenOptions (MQLONG)
OpenOptions utilisé pour ouvrir l'objet faisant l'objet d'un rapport. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

La valeur de OpenOptions dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

zéro.

MQSTAT_TYPE_RECONNEXION

zéro.

MQSTAT_TYPE_RECONNECTION_ERROR

OpenOptions utilisé lorsque l'incident s'est produit. La raison de l'échec est indiquée dans les zones *CompCode* et *Reason* de la structure MQSTS .

OpenOptions est un champ de sortie. Sa valeur initiale est zéro.

PutFailureNombre (MQLONG)
Nombre d'opérations d'insertion asynchrones ayant échoué.

La valeur de PutFailureCount dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Nombre d'opérations d'insertion asynchrone dans l'objet nommé dans la structure MQSTS qui s'est terminée avec MQCC_FAILED.

MQSTAT_TYPE_RECONNEXION

zéro.

MQSTAT_TYPE_RECONNECTION_ERROR

zéro.

PutFailureCount est un champ de sortie. Sa valeur initiale est zéro.

PutSuccessNombre (MQLONG)
Nombre d'opérations d'insertion asynchrones ayant abouti.

La valeur de PutSuccessCount dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Nombre d'opérations d'insertion asynchrone dans l'objet nommé dans la structure MQSTS qui s'est terminée avec MQCC_OK.

MQSTAT_TYPE_RECONNEXION

zéro.

MQSTAT_TYPE_RECONNECTION_ERROR

zéro.

PutSuccessCount est un champ de sortie. Sa valeur initiale est zéro.

Nombre PutWarning(MQLONG)
Nombre d'opérations d'insertion asynchrones qui se sont terminées par un avertissement.

La valeur de PutWarningCount dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Nombre d'opérations d'insertion asynchrone dans l'objet nommé dans la structure MQSTS qui s'est terminée avec MQCC_WARNING.

MQSTAT_TYPE_RECONNEXION

zéro.

MQSTAT_TYPE_RECONNECTION_ERROR

zéro.

PutWarningCount est un champ de sortie. Sa valeur initiale est zéro.

SubName (MQCHARV)
Nom de l'abonnement défaillant. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

L'interprétation de SubName dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Chaîne de longueur zéro.

MQSTAT_TYPE_RECONNEXION

Chaîne de longueur zéro.

MQSTAT_TYPE_RECONNECTION_ERROR

Nom de l'abonnement qui a provoqué l'échec de la reconnexion. Si aucun nom d'abonnement n'est disponible ou si l'échec n'est pas lié à un abonnement, il s'agit d'une chaîne de longueur nulle.

SubName est un champ de sortie. Sa valeur initiale est une chaîne de longueur nulle.

SubOptions (MQLONG)
SubOptions utilisé pour ouvrir l'abonnement défaillant. Présent uniquement dans la version 2 de MQSTS ou ultérieure.

L'interprétation de SubOptions dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

zéro.

MQSTAT_TYPE_RECONNEXION

zéro.

MQSTAT_TYPE_RECONNECTION_ERROR

SubOptions utilisé lorsque l'incident s'est produit. Si l'échec n'est pas lié à l'abonnement à une rubrique, la valeur renvoyée est zéro.

SubOptions est un champ de sortie. Sa valeur initiale est zéro.

Motif (MQLONG)
Code anomalie de l'opération faisant l'objet d'un rapport.

L'interprétation de Reason dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

Il s'agit du code raison résultant d'une opération d'insertion asynchrone précédente sur l'objet spécifié dans ObjectName.

MQSTAT_TYPE_RECONNEXION

Si la connexion est en cours de reconnexion ou n'a pas pu être reconnectée, il s'agit du code raison qui a provoqué le début de la reconnexion.

Si la connexion est actuellement connectée, la valeur est MQRC_NONE.

MQSTAT_TYPE_RECONNECTION_ERROR

Si la reconnexion a échoué, il s'agit du code raison qui a provoqué l'échec de la reconnexion.

Si la connexion est actuellement connectée ou reconnectée, la valeur est MQRC_NONE.

Reason est un champ de sortie. Sa valeur initiale est MQRC_NONE.

ResolvedObjectNom (MQCHAR48)
Le nom de l'objet nommé dans *ObjectName* après que le gestionnaire de files d'attente local a résolu le nom.

L'interprétation de ResolvedObjectName dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

ResolvedObjectName est le nom de l'objet nommé dans *ObjectName* après que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom d'un objet qui existe sur le gestionnaire de files d'attente identifié par *ResolvedQMgrName*.

MQSTAT_TYPE_RECONNEXION

Non renseigné.

MQSTAT_TYPE_RECONNECTION_ERROR

Non renseigné.

ResolvedObjectName est un champ de sortie. Sa valeur initiale est la chaîne nulle en C et 48 caractères blancs dans les autres langages de programmation.

ResolvedQMgr(MQCHAR48)
Nom du gestionnaire de files d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom.

L'interprétation de ResolvedQMgrName dépend de la valeur du paramètre MQSTAT Type .

MQSTAT_TYPE_ASYNC_ERROR

ResolvedQMgrName est le nom du gestionnaire de files d'attente de destination une fois que le gestionnaire de files d'attente local a résolu le nom. Le nom renvoyé est le nom du gestionnaire de files d'attente qui possède l'objet identifié par *ResolvedObjectName*. *ResolvedQMgrName* peut être le nom du gestionnaire de files d'attente local.

MQSTAT_TYPE_RECONNEXION

Non renseigné.

MQSTAT_TYPE_RECONNECTION_ERROR

Non renseigné.

ResolvedQMgrName est toujours un champ de sortie. Sa valeur initiale est la chaîne nulle en C et 48 caractères blancs dans les autres langages de programmation.

StrucId (MQCHAR4)
Identificateur de la structure de génération de rapports de statut, MQSTS.

StrucId est l'identificateur de structure. La valeur doit être:

ID_STRUCTE_MQST
Identificateur de la structure de génération de rapports de statut.

Pour le langage de programmation C, la constante MQSTS_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQSTS_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

StrucId est toujours une zone d'entrée. Sa valeur initiale est MQSTS_STRUC_ID.

Version (MQLONG)
Numéro de version de la structure.

La valeur doit être:

MQSTS_VERSION_1
Structure de génération de rapports de statut de la version 1.

MQSTS_VERSION_2
Structure de génération de rapports d'état de la version 2.

La constante suivante indique le numéro de version de la version en cours:

MQSTS_CURRENT_VERSION
Version actuelle de la structure de génération de rapports de statut. La version actuelle est MQSTS_VERSION_2.

Version est toujours une zone d'entrée. Sa valeur initiale est MQSTS_VERSION_1.

Valeurs initiales et déclarations de langue pour MQSTS

Tableau 552. Valeurs initiales des zones dans MQSTS

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUCTE_MQST	'STAT¬'
Version	MQSTS_VERSION_1	1
CompCode	MQCC_OK	0

Tableau 552. Valeurs initiales des zones dans MQSTS (suite)

Nom de zone	Nom de la constante	Valeur de la constante
Reason	MQRC_NONE	0
PutSuccessCount	Aucun	0
PutWarningCount	Aucun	0
PutFailureCount	Aucun	0
ObjectType	MQOT_Q	1
ObjectName	Aucun	Chaîne nulle ou blancs
ObjectQMgrName	Aucun	Chaîne nulle ou blancs
ResolvedObjectName	Aucun	Chaîne nulle ou blancs
ResolvedQMgrName	Aucun	Chaîne nulle ou blancs
ObjectString	MQCHARV_VALEUR PAR DEFAUT	{NULL,0,0,0,-3}
SubName	MQCHARV_VALEUR PAR DEFAUT	{NULL,0,0,0,-3}
OpenOptions	Aucun	0
SubOptions	Aucun	0

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable de macro MQSTS_DEFAULT contient les valeurs répertoriées ci-dessus. Il peut être utilisé de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQSTS MySTS = {MQSTS_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
      ** 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 **
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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-Message de déclenchement

Le tableau suivant récapitule les zones de la structure.

Tableau 553. Zones dans MQTM

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
QName	Nom de la file d'attente déclenchée	nomQ
ProcessName	Nom de l'objet processus	ProcessName
TriggerData	Données de déclenchement	TriggerData
ApplType	Type d'application	ApplType
ApplId	Identificateur d'application	ApplId

Tableau 553. Zones dans MQTM (suite)		
Zone	**Description**	**Topic**
EnvData	Données d'environnement	EnvData
UserData	Données utilisateur	UserData

Présentation de MQTM

Objectif: La structure MQTM décrit les données du message de déclenchement envoyé par le gestionnaire de files d'attente à une application de moniteur de déclenchement lorsqu'un événement de déclenchement se produit pour une file d'attente.

Cette structure fait partie de l'interface TMI (Trigger Monitor Interface) WebSphere MQ , qui est l'une des interfaces de l'infrastructure WebSphere MQ .

Nom de format: MQFMT_TRIGGER.

Jeu de caractères et codage: les données de type caractère dans MQTM se trouvent dans le jeu de caractères du gestionnaire de files d'attente qui génère le MQTM. Les données numériques dans MQTM se trouvent dans le codage machine du gestionnaire de files d'attente qui génère le MQTM.

Le jeu de caractères et le codage du MQTM sont fournis par les zones *CodedCharSetId* et *Encoding* dans:

- MQMD (si la structure MQTM est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQTM (tous les autres cas).

Utilisation: Une application de moniteur de déclenchement peut avoir besoin de transmettre une partie ou la totalité des informations du message de déclencheur à l'application démarrée par l'application de moniteur de déclenchement. Les informations qui peuvent être requises par l'application démarrée incluent *QName*, *TriggerData*et *UserData*. L'application de moniteur de déclenchement peut transmettre la structure MQTM directement à l'application démarrée ou transmettre une structure MQTMC2 à la place, en fonction de ce qui est autorisé par l'environnement et pratique pour l'application démarrée. Pour plus d'informations sur MQTMC2, voir «MQTMC2 -Message de déclenchement 2 (format de caractère)», à la page 588.

- Sous z/OS, pour une application MQAT_CICS démarrée à l'aide de la transaction CKTI, l'intégralité de la structure de message de déclenchement MQTM est mise à la disposition de la transaction démarrée ; les informations peuvent être extraites à l'aide de la commande EXEC CICS RETRIEVE.
- Sous IBM i, l'application de moniteur de déclenchement fournie avec WebSphere MQ transmet une structure MQTMC2 à l'application démarrée.

Pour plus d'informations sur l'utilisation de déclencheurs, voir Démarrage d'applications WebSphere MQ à l'aide de déclencheurs .

MQMD pour un message de déclenchement: les zones du MQMD d'un message de déclenchement généré par le gestionnaire de files d'attente sont définies comme suit:

Zone dans MQMD	**Valeur utilisée**
StrucId	ID_STRUCD_MQM
Version	MQMD_VERSION_1
Report	MQRO_AUCUN
MsgType	MQMT_DATAGRAM
Expiry	MQEI_UNLIMITED
Feedback	MQFB_AUCUN
Encoding	MQENC_NATIVE
CodedCharSetId	Attribut *CodedCharSetId* du gestionnaire de files d'attente

Zone dans MQMD	Valeur utilisée
Format	MQFMT_TRIGGER
Priority	Attribut *DefPriority* de la file d'attente d'initialisation
Persistence	MQPER_NON_PERSISTENT
MsgId	Une valeur unique
CorrelId	MQCI_NONE
BackoutCount	0
ReplyToQ	Espaces vides
ReplyToQMgr	Nom du gestionnaire de files d'attente
UserIdentifier	Espaces vides
AccountingToken	MQACT_AUCUN
ApplIdentityData	Espaces vides
PutApplType	MQAT_QMGR, ou selon le cas, pour l'agent MCA
PutApplName	28 premiers octets du nom du gestionnaire de files d'attente
PutDate	Date d'envoi du message de déclenchement
PutTime	Heure d'envoi du message de déclenchement
ApplOriginData	Espaces vides

Une application qui génère un message de déclenchement est recommandée pour définir des valeurs similaires, à l'exception des suivantes:

- La zone *Priority* peut être définie sur MQPRI_PRIORITY_AS_Q_DEF (le gestionnaire de files d'attente remplace la priorité par défaut de la file d'attente d'initialisation lorsque le message est inséré).
- La zone *ReplyToQMgr* peut être mise à blanc (le gestionnaire de files d'attente change le nom du gestionnaire de files d'attente local lors de l'insertion du message).
- Définissez les zones de contexte appropriées pour l'application.

Zones pour MQTM

La structure MQTM contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

ApplId (MQCHAR256)

Il s'agit d'une chaîne de caractères qui identifie l'application à démarrer et qui est utilisée par l'application trigger-monitor qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut *ApplId* de l'objet de processus identifié par la zone *ProcessName* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus», à la page 859 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

La signification de *ApplId* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par WebSphere MQ requiert que *ApplId* soit le nom d'un programme exécutable. Les remarques suivantes s'appliquent aux environnements indiqués:

- Sous z/OS, *ApplId* est:
  - Un identificateur de transaction CICS , pour les applications démarrées à l'aide de la transaction CKTI du moniteur de déclenchement CICS
  - Un identificateur de transaction IMS , pour les applications démarrées à l'aide du moniteur de déclenchement IMS CSQQTRMN
- Sur les systèmes Windows , le nom du programme peut être précédé d'un lecteur et d'un chemin de répertoire.

- Sous IBM i, le nom de programme peut être précédé d'un nom de bibliothèque et d'un caractère /.
- Sur les systèmes UNIX , le nom du programme peut être précédé d'un chemin de répertoire.

La longueur de cette zone est indiquée par MQ_PROCESS_APPL_ID_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 256 caractères blancs dans les autres langages de programmation.

ApplType (MQLONG)

Identifie la nature du programme à démarrer et est utilisé par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut *ApplType* de l'objet de processus identifié par la zone *ProcessName* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus», à la page 859 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

ApplType peut avoir l'une des valeurs standard suivantes. Les types définis par l'utilisateur peuvent également être utilisés, mais ils doivent être limités aux valeurs comprises entre MQAT_USER_FIRST et MQAT_USER_LAST:

MQAT_AIX
Application AIX (même valeur que MQAT_UNIX).

MQAT_BATCH
application de traitement par lots

MQAT_COURTIER
Application de courtier

MQAT_CICS
Transaction CICS .

MQAT_CICS_BRIDGE
Application de pont CICS .

MQAT_CICS_VSE
Transaction CICS/VSE .

MQAT_DOS
WebSphere MQ Application client MQI sur PC DOS.

MQAT_IMS
Application IMS .

MQAT_IMS_BRIDGE
Application de passerelle IMS .

MQAT_JAVA
Application Java.

MQAT_MVS
Application MVS ou TSO (même valeur que MQAT_ZOS).

MQAT_NOTES_AGENT
Lotus Notes Application Agent.

MQAT_NSK
Application HP Integrity NonStop Server .

MQAT_OS390
Application OS/390 (même valeur que MQAT_ZOS).

MQAT_OS400
Application IBM i .

MQAT_RRS_BATCH
Application par lots RRS.

MQAT_UNIX
Application UNIX .

MQAT_INCONNU
Application de type inconnu.

Utilisateur_MQAT
Type d'application défini par l'utilisateur.

MQAT_VOS
Application Stratus VOS.

MQAT_WINDOWS
Application Windows 16 bits.

MQAT_WINDOWS_NT
Application Windows 32 bits.

MQAT_WLM
Application de gestionnaire de charge de travail z/OS .

MQAT_XCF
XCF.

MQAT_ZOS
Application z/OS .

MQAT_USER_FIRST
Valeur la plus faible pour le type d'application défini par l'utilisateur.

MQAT_USER_LAST
Valeur la plus élevée pour le type d'application défini par l'utilisateur.

La valeur initiale de cette zone est `0`.

EnvData (MQCHAR128)

Il s'agit d'une chaîne de caractères qui contient des informations liées à l'environnement concernant l'application à démarrer et qui est utilisée par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut *EnvData* de l'objet de processus identifié par la zone *ProcessName* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus», à la page 859 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

Sous z/OS, pour une application CICS démarrée à l'aide de la transaction CKTI ou une application IMS à démarrer à l'aide de la transaction CSQQTRMN, ces informations ne sont pas utilisées.

La longueur de cette zone est indiquée par MQ_PROCESS_ENV_DATA_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 128 caractères blancs dans les autres langages de programmation.

ProcessName (MQCHAR48)

Il s'agit du nom de l'objet de processus de gestionnaire de files d'attente spécifié pour la file d'attente déclenchée. Il peut être utilisé par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut *ProcessName* de la file d'attente identifiée par la zone *QName* ; pour plus de détails sur cet attribut, voir «Attributs des files d'attente», à la page 824 .

Les noms inférieurs à la longueur définie de la zone sont toujours complétés à droite par des blancs ; ils ne sont pas terminés prématurément par un caractère NULL.

La longueur de cette zone est indiquée par MQ_PROCESS_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

QName (MQCHAR48)

Il s'agit du nom de la file d'attente pour laquelle un événement déclencheur s'est produit. Il est utilisé par l'application démarrée par l'application du moniteur de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut *QName* de la file d'attente de déclenchement ; pour plus de détails sur cet attribut, voir «Attributs des files d'attente», à la page 824 .

Les noms inférieurs à la longueur définie de la zone sont complétés à droite par des blancs ; ils ne sont pas terminés prématurément par un caractère NULL.

La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure. La valeur doit être:

ID_STRUC_MQTM
Identificateur de la structure de message de déclenchement.

Pour le langage de programmation C, la constante MQTM_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQTM_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQTM_STRUC_ID.

TriggerData (MQCHAR64)

Il s'agit de données au format libre à utiliser par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut *TriggerData* de la file d'attente identifiée par la zone *QName* ; pour plus de détails sur cet attribut, voir «Attributs des files d'attente», à la page 824 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

Sous z/OS, pour une application CICS démarrée à l'aide de la transaction CKTI, ces informations ne sont pas utilisées.

La longueur de cette zone est indiquée par MQ_TRIGGER_DATA_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 64 caractères blancs dans les autres langages de programmation.

UserData (MQCHAR128)

Il s'agit d'une chaîne de caractères qui contient des informations utilisateur relatives à l'application à démarrer et qui est utilisée par l'application du moniteur de déclenchement qui reçoit le message de déclenchement. Le gestionnaire de files d'attente initialise cette zone avec la valeur de l'attribut *UserData* de l'objet de processus identifié par la zone *ProcessName* ; pour plus de détails sur cet attribut, voir «Attributs des définitions de processus», à la page 859 . Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente.

Pour Microsoft Windows, la chaîne de caractères ne doit pas contenir de guillemets si la définition de processus doit être transmise à **runmqtrm**.

La longueur de cette zone est indiquée par MQ_PROCESS_USER_DATA_LENGTH. La valeur initiale de cette zone est la chaîne nulle en C et 128 caractères blancs dans les autres langages de programmation.

Version (MQLONG)

Il s'agit du numéro de version de la structure. La valeur doit être:

MQTM_VERSION_1
Numéro de version de la structure de message de déclenchement.

La constante suivante indique le numéro de version de la version en cours:

MQTM_CURRENT_VERSION
Version actuelle de la structure de message de déclenchement.

La valeur initiale de cette zone est MQTM_VERSION_1.

Valeurs initiales et déclarations de langue pour MQTM

Tableau 554. Valeurs initiales des zones dans MQTM pour MQTM

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQTM	'TM¬¬'
Version	MQTM_VERSION_1	1
QName	Aucun	Chaîne nulle ou blancs
ProcessName	Aucun	Chaîne nulle ou blancs
TriggerData	Aucun	Chaîne nulle ou blancs
ApplType	Aucun	0
ApplId	Aucun	Chaîne nulle ou blancs
EnvData	Aucun	Chaîne nulle ou blancs
UserData	Aucun	Chaîne nulle ou blancs

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQTM_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQTM MyTM = {MQTM_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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).
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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 -Message de déclenchement 2 (format de caractère)

Le tableau suivant récapitule les zones de la structure.

Tableau 555. Zones dans MQTMC2

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
QName	Nom de la file d'attente déclenchée	nomQ
ProcessName	Nom de l'objet processus	ProcessName
TriggerData	Données de déclenchement	TriggerData
ApplType	Type d'application	ApplType

Tableau 555. Zones dans MQTMC2 (suite)		
Zone	**Description**	**Topic**
ApplId	Identificateur d'application	ApplId
EnvData	Données d'environnement	EnvData
UserData	Données utilisateur	UserData
QMgrName	Nom gest. de files	QMgrName

Présentation de MQTMC2

Objectif: Lorsqu'une application de moniteur de déclenchement extrait un message de déclenchement (MQTM) d'une file d'attente d'initialisation, le moniteur de déclenchement peut avoir besoin de transmettre tout ou partie des informations du message de déclenchement à l'application démarrée par le moniteur de déclenchement.

Les informations dont l'application démarrée peut avoir besoin incluent *QName*, *TriggerData* et *UserData*. L'application du moniteur de déclenchement peut transmettre la structure MQTM directement à l'application démarrée ou transmettre une structure MQTMC2 à la place, en fonction de ce qui est autorisé par l'environnement et pratique pour l'application démarrée.

Cette structure fait partie de l'interface TMI (Trigger Monitor Interface) WebSphere MQ , qui est l'une des interfaces de l'infrastructure WebSphere MQ .

Jeu de caractères et codage: les données de type caractère dans MQTMC2 se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente *CodedCharSetId* .

Syntaxe: la structure MQTMC2 est très similaire au format de la structure MQTM. La différence réside dans le fait que les zones non alphanumériques de MQTM sont remplacées dans MQTMC2 par des zones alphanumériques de même longueur et que le nom du gestionnaire de files d'attente est ajouté à la fin de la structure.

- Sous z/OS, pour une application MQAT_IMS démarrée à l'aide de l'application CSQQTRMN, une structure MQTMC2 est mise à la disposition de l'application démarrée.
- Sous IBM i, l'application du moniteur de déclenchement fournie avec WebSphere MQ transmet une structure MQTMC2 à l'application démarrée.

Zones pour MQTMC2

La structure MQTMC2 contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

ApplId (MQCHAR256)

Identificateur d'application.

Voir la zone *ApplId* dans la structure MQTM.

ApplType (MQCHAR4)
Type d'application.

Cette zone contient toujours des blancs, quelle que soit la valeur de la zone *ApplType* dans la structure MQTM du message de déclenchement d'origine.

EnvData (MQCHAR128)
Données d'environnement.

Voir la zone *EnvData* dans la structure MQTM.

ProcessName (MQCHAR48)
Nom de l'objet de processus.

Voir la zone *ProcessName* dans la structure MQTM.

QMgrName (MQCHAR48)
Nom du gestionnaire de files d'attente

Il s'agit du nom du gestionnaire de files d'attente dans lequel l'événement déclencheur s'est produit.

QName (MQCHAR48)
Nom de la file d'attente déclenchée.

Voir la zone *QName* dans la structure MQTM.

StrucId (MQCHAR4)
Identificateur de structure.

La valeur doit être:

ID MQTMC_STRUC_ID
Identificateur de la structure du message de déclenchement (format de caractères).

Pour le langage de programmation C, la constante MQTMC_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQTMC_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

TriggerData (MQCHAR64)
Données de déclenchement.

Voir la zone *TriggerData* dans la structure MQTM.

UserData (MQCHAR128)
Données utilisateur.

Voir la zone *UserData* dans la structure MQTM.

Version (MQCHAR4)
Numéro de version de la structure.

La valeur doit être:

MQTMC_VERSION_2
Structure du message de déclenchement version 2 (format de caractère).

Pour le langage de programmation C, la constante MQTMC_VERSION_2_ARRAY est également définie ; elle a la même valeur que MQTMC_VERSION_2, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La constante suivante indique le numéro de version de la version en cours:

MQTMC_CURRENT_VERSION
Version en cours de la structure du message de déclenchement (format de caractères).

Valeurs initiales et déclarations de langue pour MQTMC2

Tableau 556. Valeurs initiales des zones dans MQTMC2 pour MQTMC2

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID MQTMC_STRUC_ID	'TMC¬'
Version	MQTMC_VERSION_2	'¬¬¬2'
QName	Aucun	Chaîne nulle ou blancs
ProcessName	Aucun	Chaîne nulle ou blancs

Tableau 556. Valeurs initiales des zones dans MQTMC2 pour MQTMC2 (suite)		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
TriggerData	Aucun	Chaîne nulle ou blancs
ApplType	Aucun	Espaces vides
ApplId	Aucun	Chaîne nulle ou blancs
EnvData	Aucun	Chaîne nulle ou blancs
UserData	Aucun	Chaîne nulle ou blancs
QMgrName	Aucun	Chaîne nulle ou blancs

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQTMC2_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQTMC2 MyTMC = {MQTMC2_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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).
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-En-tête d'informations de travail

Le tableau suivant récapitule les zones de la structure.

Tableau 557. Zones dans MQWIH

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
StrucLength	Longueur de la structure MQWIH	StrucLength
Encoding	Codage numérique des données qui suit MQWIH	Codage
CodedCharSetId	Identificateur de jeu de caractères des données qui suit MQWIH	CodedCharSetId

Tableau 557. Zones dans MQWIH (suite)		
Zone	**Description**	**Topic**
Format	Nom du format des données qui suit MQWIH	Format
Flags	Indicateurs	Indicateurs
ServiceName	Nom du service	ServiceName
ServiceStep	Nom d'étape du service	ServiceStep
MsgToken	Jeton de message	MsgToken
Reserved	Réservé	Reserved

Présentation de MQWIH

Disponibilité: Tous les systèmes WebSphere MQ , plus les clients WebSphere MQ connectés à ces systèmes.

Objectif: La structure MQWIH décrit les informations qui doivent être présentes au début d'un message qui doit être géré par le gestionnaire de charge de travail z/OS .

Nom de format: MQFMT_WORK_INFO_HEADER.

Jeu de caractères et codage: les zones de la structure MQWIH se trouvent dans le jeu de caractères et le codage donnés par les zones *CodedCharSetId* et *Encoding* de la structure d'en-tête qui précède MQWIH, ou par les zones de la structure MQMD si MQWIH se trouve au début des données de message d'application.

Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms de file d'attente.

Utilisation: si un message doit être traité par le gestionnaire de charge de travail z/OS , il doit commencer par une structure MQWIH.

Zones pour MQWIH

La structure MQWIH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

CodedCharSetId (MQLONG)

Indique l'identificateur de jeu de caractères des données qui suivent la structure MQWIH ; il ne s'applique pas aux données de type caractères de la structure MQWIH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Vous pouvez utiliser la valeur spéciale suivante:

MQCCSI_HÉRITER
Les données de type caractères dans les données *suivant* cette structure se trouvent dans le même jeu de caractères que cette structure.

Le gestionnaire de files d'attente remplace cette valeur dans la structure envoyée dans le message par l'identificateur de jeu de caractères réel de la structure. Si aucune erreur ne se produit, la valeur MQCCSI_INHERIT n'est pas renvoyée par l'appel MQGET.

MQCCSI_INHERIT ne peut pas être utilisé si la valeur de la zone *PutApplType* dans MQMD est MQAT_BROKER.

La valeur initiale de cette zone est MQCCSI_UNDEFINED.

Codage (MQLONG)

Indique le codage numérique des données qui suivent la structure MQWIH ; il ne s'applique pas aux données numériques de la structure MQWIH elle-même.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données.

La valeur initiale de cette zone est 0.

Indicateurs (MQLONG)

La valeur doit être:

MQWIH_AUCUN
Aucun indicateur.

La valeur initiale de cette zone est MQWIH_NONE.

Format (MQCHAR8)

Indique le nom de format des données qui suivent la structure MQWIH.

Dans l'appel MQPUT ou MQPUT1 , l'application doit définir cette zone sur la valeur appropriée aux données. Les règles de codage de cette zone sont les mêmes que celles de la zone *Format* dans MQMD.

La longueur de cette zone est indiquée par MQ_FORMAT_LENGTH. La valeur initiale de cette zone est MQFMT_NONE.

MsgToken (MQBYTE16)

Il s'agit d'un jeton de message qui identifie le message de manière unique.

Pour les appels MQPUT et MQPUT1 , cette zone est ignorée. La longueur de cette zone est indiquée par MQ_MSG_TOKEN_LENGTH. La valeur initiale de cette zone est MQMTOK_NONE.

Réservé (MQCHAR32)

Il s'agit d'une zone réservée ; elle doit être vide.

ServiceName (MQCHAR32)

Il s'agit du nom du service qui doit traiter le message.

La longueur de cette zone est indiquée par MQ_SERVICE_NAME_LENGTH. La valeur initiale de cette zone est de 32 caractères blancs.

ServiceStep (MQCHAR8)

Il s'agit du nom de l'étape de *ServiceName* à laquelle le message est lié.

La longueur de cette zone est indiquée par MQ_SERVICE_STEP_LENGTH. La valeur initiale de cette zone est de 8 caractères blancs.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure. La valeur doit être:

ID_STRUC_MQWIH
Identificateur de la structure d'en-tête des informations de travail.

Pour le langage de programmation C, la constante MQWIH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQWIH_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

La valeur initiale de cette zone est MQWIH_STRUC_ID.

StrucLength (MQLONG)

Longueur de la structure MQWIH. La valeur doit être:

MQWIH_LENGTH_1
Longueur de la structure d'en-tête des informations de travail version-1 .

La constante suivante indique la longueur de la version en cours:

MQWIH_LONGUEUR_EN_COURS
Longueur de la version actuelle de la structure d'en-tête des informations de travail.

La valeur initiale de cette zone est MQWIH_LENGTH_1.

Version (MQLONG)

Il s'agit du numéro de version de la structure. La valeur doit être:

MQWIH_VERSION_1
Structure d'en-tête des informations de travail Version-1 .

La constante suivante indique le numéro de version de la version en cours:

MQWIH_CURRENT_VERSION
Version actuelle de la structure d'en-tête des informations de travail.

La valeur initiale de cette zone est MQWIH_VERSION_1.

Valeurs initiales et déclarations de langage pour MQWIH

Tableau 558. Valeurs initiales des zones dans MQWIH pour MQWIH

Nom de zone	Nom de la constante	Valeur de la constante
StrucId	ID_STRUC_MQWIH	'WIH¬'
Version	MQWIH_VERSION_1	1
StrucLength	MQWIH_LENGTH_1	120
Encoding	Aucun	0
CodedCharSetId	MQCCSI_UNDEFINI	0
Format	MQFMT_AUCUN	Espaces vides
Flags	MQWIH_AUCUN	0
ServiceName	Aucun	Espaces vides
ServiceStep	Aucun	Espaces vides
MsgToken	MQMTOK_NONE	Valeurs NULL
Reserved	Aucun	Espaces vides

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. Dans le langage de programmation C, la variable macroMQWIH_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQWIH MyWIH = {MQWIH_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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).
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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-Bloc de paramètres d'exit

Le tableau suivant récapitule les zones de la structure.

Tableau 559. Zones dans MQXP

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
ExitId	Identificateur d'exit	ExitId
ExitReason	Raison de l'appel d'exit	ExitReason
ExitResponse	Réponse de l'exit	ExitResponse
ExitCommand	Code d'appel d'API	ExitCommand
ExitParmCount	Nombre de paramètres	ExitParmNombre
ExitUserArea	Zone utilisateur	ZoneExitUser

Présentation de MQXP

Disponibilité: z/OS.

Objectif: La structure MQXP est utilisée en tant que paramètre d'entrée/sortie de l'exit de croisement d'API. Pour plus d'informations sur cet exit, voir L'exit de croisement d'API.

Jeu de caractères et codage: les données de type caractère dans MQXP se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente *CodedCharSetId* . Les données numériques dans MQXP sont dans le codage de la machine native ; elles sont fournies par MQENC_NATIVE.

Zones pour MQXP

La structure MQXP contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

ExitCommand (MQLONG)

Cette zone est définie à l'entrée de la routine d'exit. Il identifie l'appel d'API qui a provoqué l'appel de l'exit:

MQXC_RAPPEL
Appel CALLBACK.

MQXC_MQBACK
Appel MQBACK.

MQXC_MQCB
Appel MQCB.

MQXC_MQCLOSE
Appel MQCLOSE.

MQXC_MQCMIT
Appel MQCMIT.

MQXC_MQCTL
Appel MQCTL.

MQXC_MQGET
Appel MQGET.

MQXC_MQINQ
Appel MQINQ.

MQXC_MQOPEN
Appel MQOPEN.

MQXC_MQPUT
Appel MQPUT.

MQXC_MQPUT1
L'appel MQPUT1 .

MQXC_MQSET
Appel MQSET.

MQXC_MQSTAT
Appel MQSTAT.

MQXC_MQSUB
Appel MQSUB.

MQXC_MQSUBRQ
Appel MQSUBRQ.

Il s'agit d'une zone d'entrée de l'exit.

ExitId (MQLONG)

Cette valeur est définie à l'entrée de la routine d'exit et indique le type d'exit:

EXIT MQXT_API_CROSSING_EXIT
Exit de croisement d'API pour CICS.

Il s'agit d'une zone d'entrée de l'exit.

Nombre ExitParm(MQLONG)

Cette zone est définie à l'entrée de la routine d'exit. Il contient le nombre de paramètres pris par l'appel MQ . Il s'agit des fonctions suivantes :

Nom de l'appel	Nombre de paramètres
MQBACK	3
MQCLOSE	5
MQCMIT	3
MQGET	9
MQINQ	10
MQOPEN	6
MQPUT	8

Nom de l'appel	Nombre de paramètres
MQPUT1	8
MQSET	10

Il s'agit d'une zone d'entrée de l'exit.

ExitReason (MQLONG)

Ce paramètre est défini à l'entrée de la routine d'exit. Pour l'exit de croisement d'API, il indique si la routine est appelée avant ou après l'exécution de l'appel d'API:

MQXR_AVANT
Avant l'exécution de l'API.

MQXR_APRES
Après l'exécution de l'API.

Il s'agit d'une zone d'entrée de l'exit.

ExitResponse (MQLONG)

La valeur est définie par l'exit pour communiquer avec l'appelant. Les valeurs suivantes sont définies :

MQXCC_OK
La sortie a abouti.

MQXCC_SUPPRESS_FUNCTION
Supprimer la fonction.

Lorsque cette valeur est définie par un exit de croisement d'API appelé *avant* l'appel d'API, l'appel d'API n'est pas effectué. Le paramètre *CompCode* de l'appel est défini sur MQCC_FAILED, le paramètre *Reason* est défini sur MQRC_SUPPRESSED_BY_EXIT et tous les autres paramètres restent tels que l'exit les a laissés.

Lorsque cette valeur est définie par un exit de croisement d'API appelé *après* l'appel d'API, elle est ignorée par le gestionnaire de files d'attente.

MQXCC_LISTE_FONCTION
Ignorer la fonction.

Lorsque cette valeur est définie par un exit de croisement d'API appelé *avant* l'appel d'API, l'appel d'API n'est pas effectué ; les paramètres *CompCode* et *Reason* et tous les autres paramètres restent tels que l'exit les a laissés.

Lorsque cette valeur est définie par un exit de croisement d'API appelé *après* l'appel d'API, elle est ignorée par le gestionnaire de files d'attente.

Il s'agit d'une zone de sortie de l'exit.

Zone ExitUser(MQBYTE16)

Il s'agit d'une zone disponible que l'exit peut utiliser. Il est initialisé à zéro binaire pour la longueur de la zone avant le premier appel de l'exit pour la tâche, puis toutes les modifications apportées à cette zone par l'exit sont conservées lors des appels de l'exit. La valeur suivante est définie:

MQXUA_NONE
Aucune information utilisateur.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQXUA_NONE_ARRAY est également définie ; elle a la même valeur que MQXUA_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ_EXIT_USER_AREA_LENGTH. Il s'agit d'une zone d'entrée-sortie pour l'exit.

Réservé (MQLONG)

Il s'agit d'une zone réservée. Sa valeur n'est pas significative pour l'exit.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure. La valeur doit être:

ID_STRUCT_MQXP
Identificateur de la structure des paramètres d'exit.

Pour le langage de programmation C, la constante MQXP_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQXP_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit d'une zone d'entrée de l'exit.

Version (MQLONG)

Il s'agit du numéro de version de la structure. La valeur doit être:

MQXP_VERSION_1
Numéro de version de la structure de bloc de paramètres d'exit.

Remarque : Lorsqu'une nouvelle version de cette structure est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit donc vérifier que le numéro de version est égal ou supérieur à la version la plus basse qui contient les zones que l'exit doit utiliser.

Il s'agit d'une zone d'entrée de l'exit.

Déclarations de langue

Cette structure est prise en charge dans les langages de programmation suivants.

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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).
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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-en-tête de file d'attente de transmission

Le tableau suivant récapitule les zones de la structure.

Tableau 560. Zones dans MQXQH		
Zone	**Description**	**Topic**
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
RemoteQName	Nom de la file d'attente de destination	RemoteQName
RemoteQMgrName	Nom du gestionnaire de files d'attente de destination	RemoteQMgrName
MsgDesc	Descripteur de message original	MsgDesc

Présentation de MQXQH

Disponibilité: Tous les systèmes WebSphere MQ et les clients WebSphere MQ .

Objectif: La structure MQXQH décrit les informations qui sont préfixées aux données de message d'application des messages lorsqu'ils se trouvent dans des files d'attente de transmission. Une file d'attente de transmission est un type spécial de file d'attente locale qui contient temporairement des messages destinés à des files d'attente éloignées (c'est-à-dire destinés à des files d'attente qui n'appartiennent pas au gestionnaire de files d'attente local). Une file d'attente de transmission est désignée par l'attribut de file d'attente *Usage* ayant la valeur MQUS_TRANSMISSION.

Nom de format: MQFMT_XMIT_Q_HEADER.

Jeu de caractères et codage: les données de MQXOH doivent être dans le jeu de caractères indiqué par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et dans le codage du gestionnaire de files d'attente local indiqué par MQENC_NATIVE.

Définissez le jeu de caractères et le codage du MQXQH dans les zones *CodedCharSetId* et *Encoding* dans:

- Le MQMD distinct (si la structure MQXQH est au début des données de message), ou
- Structure d'en-tête qui précède la structure MQXQH (tous les autres cas).

Utilisation: Un message qui se trouve dans une file d'attente de transmission comporte *deux* descripteurs de message:

- Un descripteur de message est stocké séparément des données de message ; il est appelé *descripteur de message distinct*et est généré par le gestionnaire de files d'attente lorsque le message est placé dans la file d'attente de transmission. Certaines zones du descripteur de message distinct sont copiées à partir du descripteur de message fourni par l'application sur l'appel MQPUT ou MQPUT1 .

  Le descripteur de message distinct est celui qui est renvoyé à l'application dans le paramètre *MsgDesc* de l'appel MQGET lorsque le message est supprimé de la file d'attente de transmission.
- Un deuxième descripteur de message est stocké dans la structure MQXQH dans le cadre des données de message ; il s'agit du *descripteur de message intégré*et il s'agit d'une copie du descripteur de message fourni par l'application sur l'appel MQPUT ou MQPUT1 (avec des variations mineures).

  Le descripteur de message imbriqué est toujours un MQMD version-1 . Si le message inséré par l'application comporte des valeurs autres que celles par défaut pour une ou plusieurs des zones version-2 du MQMD, une structure MQMDE suit le MQXQH et est à son tour suivie des données de message d'application (le cas échéant). Le MQMDE est l'un des suivants:

  – Généré par le gestionnaire de files d'attente (si l'application utilise un MQMD version-2 pour insérer le message), ou
  – Déjà présent au début des données de message d'application (si l'application utilise un MQMD version-1 pour insérer le message).

  Le descripteur de message imbriqué est celui qui est renvoyé à l'application dans le paramètre *MsgDesc* de l'appel MQGET lorsque le message est supprimé de la file d'attente de destination finale.

Zones du descripteur de message distinct: les zones du descripteur de message distinct sont définies par le gestionnaire de files d'attente comme indiqué. Si le gestionnaire de files d'attente ne prend pas en charge le MQMD version-2 , un MQMD version-1 est utilisé sans perte de fonction.

Zone dans MQMD distinct	Valeur utilisée
StrucId	ID_STRUCD_MQM
Version	MQMD_VERSION_2
Report	Copié à partir du descripteur de message imbriqué, mais avec les bits identifiés par MQRO_ACCEPT_UNSUP_IF_XMIT_MASK définis sur zéro. (Cela empêche la génération d'un message de rapport COA ou COD lorsqu'un message est placé dans ou supprimé d'une file d'attente de transmission.)
MsgType	Copié à partir du descripteur de message imbriqué.
Expiry	Copié à partir du descripteur de message imbriqué.
Feedback	Copié à partir du descripteur de message imbriqué.
Encoding	MQENC_NATIVE (voir la remarque)
CodedCharSetId	Attribut *CodedCharSetId* du gestionnaire de files d'attente.
Format	MQFMT_XMIT_Q_HEADER
Priority	Copié à partir du descripteur de message imbriqué.

Zone dans MQMD distinct	Valeur utilisée
Persistence	Copié à partir du descripteur de message imbriqué.
MsgId	Une nouvelle valeur est générée par le gestionnaire de files d'attente. Cet identificateur de message est différent du *MsgId* que le gestionnaire de files d'attente a peut-être généré pour le descripteur de message imbriqué (voir ci-dessus).
CorrelId	*MsgId* à partir du descripteur de message imbriqué. Pour les messages insérés dans SYSTEM.CLUSTER.TRANSMIT.QUEUE, *CorrelId* est réservé à une utilisation interne.
BackoutCount	0
ReplyToQ	Copié à partir du descripteur de message imbriqué.
ReplyToQMgr	Copié à partir du descripteur de message imbriqué.
UserIdentifier	Copié à partir du descripteur de message imbriqué.
AccountingToken	Copié à partir du descripteur de message imbriqué. Pour les messages insérés dans SYSTEM.CLUSTER.TRANSMIT.QUEUE, *AccountingToken* est réservé à une utilisation interne.
ApplIdentityData	Copié à partir du descripteur de message imbriqué.
PutApplType	MQAT_QMGR
PutApplName	28 premiers octets du nom du gestionnaire de files d'attente.
PutDate	Date à laquelle le message a été inséré dans la file d'attente de transmission.
PutTime	Heure à laquelle le message a été inséré dans la file d'attente de transmission.
ApplOriginData	Espaces vides
GroupId	MQGI_AUCUN
MsgSeqNumber	1
Offset	0
MsgFlags	MQMF_AUCUN
OriginalLength	MQOL_UNDEFINI

- Sous Windows, la valeur de MQENC_NATIVE for Micro Focus COBOL diffère de la valeur de C. La valeur de la zone *Encoding* dans le descripteur de message distinct est toujours la valeur de C dans ces environnements ; cette valeur est 546 en décimal. De plus, les zones entières de la structure MQXQH sont dans le codage qui correspond à cette valeur (codage Intel natif).

Zones du descripteur de message imbriqué: les zones du descripteur de message imbriqué ont les mêmes valeurs que celles du paramètre *MsgDesc* de l'appel MQPUT ou MQPUT1 , à l'exception de ce qui suit:

- La zone *Version* a toujours la valeur MQMD_VERSION_1.
- Si la zone *Priority* a la valeur MQPRI_PRIORITY_AS_Q_DEF, elle est remplacée par la valeur de l'attribut *DefPriority* de la file d'attente.
- Si la zone *Persistence* a la valeur MQPER_PERSISTENCE_AS_Q_DEF, elle est remplacée par la valeur de l'attribut *DefPersistence* de la file d'attente.
- Si la zone *MsgId* a la valeur MQMI_NONE, que l'option MQPMO_NEW_MSG_ID a été spécifiée ou que le message est un message de liste de distribution, *MsgId* est remplacé par un nouvel identificateur de message généré par le gestionnaire de files d'attente.

Lorsqu'un message de liste de distribution est divisé en messages de liste de distribution plus petits placés dans des files d'attente de transmission différentes, la zone *MsgId* de chacun des nouveaux descripteurs de message imbriqués est identique à celle du message de liste de distribution d'origine.

- Si l'option MQPMO_NEW_CORREL_ID a été spécifiée, *CorrelId* est remplacé par un nouvel identificateur de corrélation généré par le gestionnaire de files d'attente.
- Les zones de contexte sont définies comme indiqué par les options MQPMO_ * _CONTEXT spécifiées dans le paramètre *PutMsgOpts* ; les zones de contexte sont les suivantes:
  - *AccountingToken*
  - *ApplIdentityData*
  - *ApplOriginData*
  - *PutApplName*
  - *PutApplType*
  - *PutDate*
  - *PutTime*
  - *UserIdentifier*
- Les zones version-2 (si elles étaient présentes) sont supprimées de MQMD et déplacées dans une structure MQMDE, si une ou plusieurs des zones version-2 ont une valeur autre que la valeur par défaut.

Insertion de messages dans des files d'attente éloignées: lorsqu'une application insère un message dans une file d'attente éloignée (en spécifiant directement le nom de la file d'attente éloignée ou en utilisant une définition locale de la file d'attente éloignée), le gestionnaire de files d'attente local:

- Crée une structure MQXQH contenant le descripteur de message imbriqué
- Ajoute un MQMDE si nécessaire et qu'il n'est pas déjà présent
- Ajoute les données de message d'application
- Place le message dans une file d'attente de transmission appropriée

Insertion de messages directement dans des files d'attente de transmission: une application peut également insérer un message directement dans une file d'attente de transmission. Dans ce cas, l'application doit préfixer les données de message d'application avec une structure MQXQH et initialiser les zones avec les valeurs appropriées. De plus, la zone *Format* du paramètre *MsgDesc* de l'appel MQPUT ou MQPUT1 doit avoir la valeur MQFMT_XMIT_Q_HEADER.

Les données de type caractère dans la structure MQXQH créée par l'application doivent être dans le jeu de caractères du gestionnaire de files d'attente local (défini par l'attribut de gestionnaire de files d'attente *CodedCharSetId*) et les données de type entier doivent être dans le codage machine natif. En outre, les données de type caractère dans la structure MQXQH doivent être complétées par des blancs à la longueur définie de la zone ; les données ne doivent pas être arrêtées prématurément à l'aide d'un caractère null, car le gestionnaire de files d'attente ne convertit pas les caractères null et suivants en blancs dans la structure MQXQH.

Toutefois, le gestionnaire de files d'attente ne vérifie pas qu'une structure MQXQH est présente ou que des valeurs valides ont été spécifiées pour les zones.

Les applications ne doivent pas placer leurs messages directement dans SYSTEM.CLUSTER.TRANSMIT.QUEUE.

Obtention de messages à partir de files d'attente de transmission: les applications qui extraient des messages à partir d'une file d'attente de transmission doivent traiter les informations dans la structure MQXQH de manière appropriée. La présence de la structure MQXQH au début des données de message d'application est indiquée par la valeur MQFMT_XMIT_Q_HEADER renvoyée dans la zone *Format* du paramètre *MsgDesc* de l'appel MQGET. Les valeurs renvoyées dans les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* indiquent le jeu de caractères et le codage des données de type caractère et entier dans la structure MQXQH. Le jeu de caractères et le codage des données de message d'application sont définis par les zones *CodedCharSetId* et *Encoding* dans le descripteur de message imbriqué.

Zones pour MQXQH

La structure MQXQH contient les zones suivantes ; elles sont décrites dans l' **ordre alphabétique**:

MsgDesc (MQMD1)

Il s'agit du descripteur de message intégré. Il s'agit d'une copie fermée du descripteur de message MQMD qui a été spécifié en tant que paramètre *MsgDesc* dans l'appel MQPUT ou MQPUT1 lors de l'insertion initiale du message dans la file d'attente éloignée.

Remarque : Il s'agit d'un MQMD version-1 .

Les valeurs initiales des zones de cette structure sont identiques à celles de la structure MQMD.

RemoteQMgrNom (MQCHAR48)

Il s'agit du nom du gestionnaire de files d'attente ou du groupe de partage de files d'attente propriétaire de la file d'attente qui est la destination finale apparente du message.

Si le message est un message de liste de distribution, *RemoteQMgrName* est vide.

La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

RemoteQName (MQCHAR48)

Il s'agit du nom de la file d'attente de messages correspondant à la destination finale apparente du message (cette dernière peut ne pas être la destination finale si, par exemple, cette file d'attente est définie dans *RemoteQMgrName* comme étant une définition locale d'une autre file d'attente éloignée).

Si le message est un message de liste de distribution (c'est-à-dire que la zone *Format* du descripteur de message imbriqué est MQFMT_DIST_HEADER), *RemoteQName* est vide.

La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH. La valeur initiale de cette zone correspond à la chaîne nulle en C et à 48 caractères blancs dans les autres langages de programmation.

StrucId (MQCHAR4)

Il s'agit de l'identificateur de structure. La valeur doit être:

ID_STRUC_MQXQH_
Identificateur de la structure d'en-tête de file d'attente de transmission.

Pour le langage de programmation C, la constante MQXQH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQXQH_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

La valeur initiale de cette zone est MQXQH_STRUC_ID.

Version (MQLONG)

Il s'agit du numéro de version de la structure. La valeur doit être:

MQXQH_VERSION_1
Numéro de version de la structure d'en-tête de file d'attente de transmission.

La constante suivante indique le numéro de version de la version en cours:

MQXQH_CURRENT_VERSION
Version actuelle de la structure d'en-tête de file d'attente de transmission.

La valeur initiale de cette zone est MQXQH_VERSION_1.

Valeurs initiales et déclarations de langage pour MQXQH

Tableau 561. Valeurs initiales des zones dans MQXQH pour MQXQH		
Nom de zone	**Nom de la constante**	**Valeur de la constante**
StrucId	ID_STRUC_MQXQH_	'XQH¬'
Version	MQXQH_VERSION_1	1
RemoteQName	Aucun	Chaîne nulle ou blancs
RemoteQMgrName	Aucun	Chaîne nulle ou blancs
MsgDesc	Mêmes noms et valeurs que MQMD ; voir Tableau 515, à la page 443	-

Remarques :

1. Le symbole ¬ représente un caractère blanc unique.
2. La valeur Chaîne nulle ou blanc indique la chaîne nulle en C, et les caractères blancs dans les autres langages de programmation.
3. Dans le langage de programmation C, la variable macroMQXQH_DEFAULT contient les valeurs répertoriées ci-dessus. Utilisez-la de la manière suivante pour fournir des valeurs initiales pour les zones de la structure:

```
MQXQH MyXQH = {MQXQH_DEFAULT};
```

Déclaration C

```
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 */
};
```

Déclaration COBOL

```
**   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).
```

Déclaration PL/I

```
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 */
```

Déclaration High Level Assembler

```
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)
```

Déclaration Visual Basic

```
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
```

Appels de fonctions

Cette section fournit des informations sur tous les appels MQI possibles. Les descriptions, la syntaxe, les informations de paramètre, les remarques d'utilisation et les appels de langue pour chaque langue possible sont indiqués pour chacun des différents appels.

Descriptions des appels

Cette section décrit les appels MQI.

- «MQCMIT-Validation des modifications», à la page 642
- «MQCONN-Connexion du gestionnaire de files d'attente», à la page 646
- «MQCONNX - Connexion du gestionnaire de files d'attente (étendue)», à la page 654
- «MQCRTMH-Création d'un descripteur de message», à la page 660
- «MQCTL-Rappels de contrôle», à la page 663
- «MQDISC-Déconnexion du gestionnaire de files d'attente», à la page 669
- «MQDLTMH-Suppression du descripteur de message», à la page 673
- «MQDLTMP-Propriété de message Delete», à la page 676
- «Message d'extraction MQGET», à la page 678
- «MQINQ-Attributs d'objet Inquire», à la page 691
- «MQINQMP-Propriété de message d'interrogation», à la page 710
- «MQMHBUF-Conversion du descripteur de message en mémoire tampon», à la page 715
- «MQOPEN-Objet ouvert», à la page 719
- «MQPUT-Insertion d'un message», à la page 738
- «MQPUT1 -Message d'insertion unique», à la page 752
- «MQSET-Définition des attributs d'objet», à la page 762
- «MQSETMP-Définition de la propriété de message», à la page 769
- «MQSTAT-Extraction des informations de statut», à la page 773
- «MQMHBUF-Conversion du descripteur de message en mémoire tampon», à la page 715
- «MQSUB-Enregistrement d'abonnement», à la page 777
- «MQSUBRQ-Demande d'abonnement», à la page 784

L'aide en ligne sur les plateformes UNIX , sous la forme de pages *man* , est disponible pour ces appels.

Remarque : Les appels associés à la conversion de données, MQXCNVC et MQ_DATA_CONV_EXIT, se trouvent dans «Exit de conversion de données», à la page 898.

Conventions utilisées dans les descriptions d'appel

Pour chaque appel, cette collection de rubriques fournit une description des paramètres et de l'utilisation de l'appel dans un format indépendant du langage de programmation. Elle est suivie d'appels typiques de l'appel et de déclarations typiques de ses paramètres, dans chacun des langages de programmation pris en charge.

Important : Lors du codage des appels d'API WebSphere MQ , vous devez vous assurer que tous les paramètres pertinents (comme décrit dans les sections suivantes) sont fournis. Si vous ne le faites pas, vous risquez d'obtenir des résultats imprévisibles.

La description de chaque appel contient les sections suivantes:

Nom de l'appel
Nom de l'appel, suivi d'une brève description de l'objet de l'appel.

Paramètres
Pour chaque paramètre, le nom est suivi de son type de données entre parenthèses () et l'un des éléments suivants:

entrée
Vous fournissez des informations dans le paramètre lorsque vous effectuez l'appel.

sortie
Le gestionnaire de files d'attente renvoie des informations dans le paramètre lorsque l'appel aboutit ou échoue.

entrée-sortie
Vous fournissez des informations dans le paramètre lorsque vous effectuez l'appel et le gestionnaire de files d'attente modifie les informations lorsque l'appel se termine ou échoue.

Exemple :

Compcode (MQLONG)-sortie

Dans certains cas, le type de données est une structure. Dans tous les cas, vous trouverez plus d'informations sur le type de données ou la structure dans «Types de données élémentaires», à la page 218.

Les deux derniers paramètres de chaque appel sont un code achèvement et un code anomalie. Le code achèvement indique si l'appel a abouti, partiellement ou pas du tout. Des informations supplémentaires sur la réussite partielle ou l'échec de l'appel sont fournies dans le code anomalie. Pour plus d'informations sur chaque code achèvement et anomalie, voir «Codes retour», à la page 863.

Notes d'utilisation
Informations supplémentaires sur l'appel, décrivant la façon de l'utiliser et les éventuelles restrictions liées à son utilisation.

Appel du langage assembleur
Appel standard de l'appel et déclaration de ses paramètres, en langage assembleur.

Appel C
Appel standard de l'appel et déclaration de ses paramètres, en C.

Appel COBOL
Appel standard de l'appel et déclaration de ses paramètres en COBOL.

Appel PL/I
Appel typique de l'appel, et déclaration de ses paramètres, en PL/I.

Tous les paramètres sont transmis par référence.

Appel Visual Basic
Appel standard de l'appel et déclaration de ses paramètres dans Visual Basic.

Les autres conventions de notation sont les suivantes:

Constantes
Les noms des constantes sont affichés en majuscules ; par exemple, MQOO_OUTPUT. Un ensemble de constantes ayant le même préfixe est présenté comme suit: MQIA_ *. Pour la valeur d'une constante, voir «Constantes», à la page 50 .

Tableaux
Dans certains appels, les paramètres sont des tableaux de chaînes de caractères qui n'ont pas de taille fixe. Dans les descriptions de ces paramètres, une minuscule n représente une constante numérique. Lorsque vous codez la déclaration pour ce paramètre, remplacez le n par la valeur numérique requise.

Utilisation des appels en langage C

Les paramètres qui sont *uniquement en entrée* et de type MQHCONN, MQHOBJ, MQHMSG ou MQLONG sont transmis par valeur. Pour tous les autres paramètres, l' *adresse* du paramètre est transmise par valeur.

Il n'est pas nécessaire de spécifier tous les paramètres transmis par adresse chaque fois que vous appelez une fonction. Si vous n'avez pas besoin d'un paramètre particulier, spécifiez un pointeur null comme paramètre sur l'appel de fonction, à la place de l'adresse des données de paramètre. Les paramètres pour lesquels cela est possible sont identifiés dans les descriptions d'appel.

Aucun paramètre n'est renvoyé comme valeur de l'appel ; dans la terminologie C, cela signifie que tous les appels renvoient `void`.

Déclaration du paramètre Buffer

Les appels MQGET, MQPUTet MQPUT1 comportent chacun un paramètre dont le type de données n'est pas défini: le paramètre *Buffer* . Utilisez ce paramètre pour envoyer et recevoir les données de message de l'application.

Les paramètres de ce type sont présentés dans les exemples C sous la forme de tableaux de MQBYTE. Vous pouvez déclarer les paramètres de cette manière, mais il est généralement plus pratique de les déclarer comme la structure particulière qui décrit la présentation des données dans le message. Le prototype de fonction déclare le paramètre en tant que pointeur vers vide, de sorte que vous pouvez spécifier l'adresse de n'importe quel type de données en tant que paramètre sur l'appel d'appel.

Pointer-to-void est un pointeur vers des données de format non défini. Il est défini comme suit:

```
typedef void *PMQVOID;
```

MQBACK-Annulation des modifications

L'appel MQBACK indique au gestionnaire de files d'attente que tous les extractions et insertions de messages qui se sont produites depuis le dernier point de synchronisation doivent être annulées.

Les messages insérés dans une unité d'oeuvre sont supprimés ; les messages extraits dans une unité d'oeuvre sont réintégrés dans la file d'attente.

- Sous z/OS, cet appel est utilisé uniquement par les programmes batch (y compris les programmes batch DL/I IMS).
- Sous IBM i, cet appel n'est pas pris en charge pour les applications s'exécutant en mode compatibilité.

Syntaxe

MQBACK (*Hconn*, *Code achèvement*, *Raison*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Appel non valide dans l'environnement.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

MQRC_OUTCOME_MIXTE
(2123, X'84B') Le résultat de l'opération de validation ou d'exclusion est mélangé.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir Codes anomalie .

Notes d'utilisation

1. Vous pouvez utiliser cet appel uniquement lorsque le gestionnaire de files d'attente lui-même coordonne l'unité d'oeuvre. Ce nom peut être :
   - Unité d'oeuvre locale dans laquelle les modifications affectent uniquement les ressources MQ .
   - Unité de travail globale dans laquelle les modifications peuvent affecter les ressources appartenant à d'autres gestionnaires de ressources, ainsi que les ressources MQ .

   Pour plus de détails sur les unités de travail locales et globales, voir «MQBEGIN-Début de l'unité de travail», à la page 615.
2. Dans les environnements où le gestionnaire de files d'attente ne coordonne pas l'unité d'oeuvre, utilisez l'appel d'annulation approprié à la place de MQBACK. L'environnement peut également prendre en charge une interruption implicite provoquée par l'arrêt anormal de l'application.
   - Sous z/OS, utilisez les appels suivants:
     - Les programmes batch (y compris les programmes DL/I de lot IMS) peuvent utiliser l'appel MQBACK si l'unité de travail affecte uniquement les ressources MQ . Toutefois, si l'unité de travail affecte à la fois les ressources MQ et les ressources appartenant à d'autres gestionnaires de ressources (par exemple, DB2), utilisez l'appel SRRBACK fourni par le service RRS (z/OS Recoverable Resource Service). L'appel SRRBACK annule les modifications apportées aux ressources appartenant aux gestionnaires de ressources qui ont été activés pour la coordination RRS.
     - Les applications CICS doivent utiliser la commande `EXEC CICS SYNCPOINT ROLLBACK` pour rétablir l'unité de travail. N'utilisez pas l'appel MQBACK pour les applications CICS.

  - Les applications IMS (autres que les programmes batch DL/I) doivent utiliser des appels IMS tels que ROLB pour rétablir l'unité de travail. N'utilisez pas l'appel MQBACK pour les applications IMS (autres que les programmes DL/I de traitement par lots).
- Sous IBM i, utilisez cet appel pour les unités d'oeuvre locales coordonnées par le gestionnaire de files d'attente. Cela signifie qu'une définition de validation ne doit pas exister au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre CMTSCOPE(*JOB) ne doit pas avoir été émise pour le travail.

3. Si une application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans «MQDISC-Déconnexion du gestionnaire de files d'attente», à la page 669 .
4. Lorsqu'une application insère ou extrait des messages dans des groupes ou des segments de messages logiques, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages et au message logique pour les derniers appels MQPUT et MQGET ayant abouti. Ces informations sont associées à l'identificateur de file d'attente et incluent les éléments suivants:
   - Valeurs des zones *GroupId*, *MsgSeqNumber*, *Offset*et *MsgFlags* dans MQMD.
   - Indique si le message fait partie d'une unité de travail.
   - Pour l'appel MQPUT: indique si le message est persistant ou non persistant.

   Le gestionnaire de files d'attente conserve *trois* ensembles d'informations de groupe et de segment, un ensemble pour chacun des éléments suivants:
   - Dernier appel MQPUT réussi (qui peut faire partie d'une unité de travail).
   - Dernier appel MQGET réussi qui a supprimé un message de la file d'attente (il peut faire partie d'une unité d'oeuvre).
   - Dernier appel MQGET ayant réussi à parcourir un message dans la file d'attente (ce *ne peut pas* faire partie d'une unité d'oeuvre).

5. Les informations associées à l'appel MQGET sont restaurées à la valeur qu'elles avaient avant le premier appel MQGET réussi pour ce descripteur de file d'attente dans l'unité d'oeuvre en cours.

   Les files d'attente qui ont été mises à jour par l'application après le démarrage de l'unité de travail, mais en dehors de la portée de l'unité de travail, ne voient pas leurs informations de groupe et de segment restaurées si l'unité de travail est annulée.

   La restauration des informations de groupe et de segment à sa valeur précédente lors de l'annulation d'une unité de travail permet à l'application de répartir un grand groupe de messages ou un grand message logique composé de plusieurs segments sur plusieurs unités de travail, et de redémarrer au point approprié du groupe de messages ou du message logique si l'une des unités de travail échoue.

   L'utilisation de plusieurs unités d'oeuvre peut s'avérer avantageuse si le gestionnaire de files d'attente local ne dispose que d'une mémoire de file d'attente limitée. Toutefois, l'application doit conserver suffisamment d'informations pour pouvoir redémarrer l'insertion ou l'obtention de messages au point approprié en cas de défaillance du système.

   Pour plus de détails sur le redémarrage au point approprié après une défaillance du système, voir l'option MQPMO_LOGICAL_ORDER décrite dans «MQPMO-Options d'insertion de message», à la page 477et l'option MQGMO_LOGICAL_ORDER décrite dans «Options MQGMO-Get-message», à la page 345.

   Les autres remarques d'utilisation s'appliquent uniquement lorsque le gestionnaire de files d'attente coordonne les unités de travail.
6. Une unité de travail a la même portée qu'un descripteur de connexion. Tous les appels MQ qui affectent une unité de travail particulière doivent être exécutés à l'aide du même descripteur de connexion. Les appels émis à l'aide d'un descripteur de connexion différent (par exemple, les appels émis par une autre application) affectent une unité d'oeuvre différente. Pour plus d'informations sur la portée des descripteurs de connexion, voir le paramètre *Hconn* décrit dans «MQCONN-Connexion du gestionnaire de files d'attente», à la page 646 .

7. Seuls les messages insérés ou extraits dans le cadre de l'unité d'oeuvre en cours sont affectés par cet appel.
8. Une application à exécution longue qui émet des appels MQGET, MQPUT ou MQPUT1 dans une unité de travail, mais qui n'émet jamais d'appel de validation ou d'annulation, peut remplir les files d'attente avec des messages qui ne sont pas disponibles pour d'autres applications. Pour éviter cette possibilité, l'administrateur doit définir l'attribut de gestionnaire de files d'attente *MaxUncommittedMsgs* sur une valeur suffisamment faible pour empêcher les applications en boucle indéfinie de remplir les files d'attente, mais suffisamment élevée pour permettre aux applications de messagerie attendues de fonctionner correctement.

Appel C

```
MQBACK (Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

Appel COBOL

```
CALL 'MQBACK' USING HCONN, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQBACK (Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn     fixed bin(31);  /* Connection handle */
dcl CompCode  fixed bin(31);  /* Completion code */
dcl Reason    fixed bin(31);  /* Reason code qualifying CompCode */
```

Appel d'assembleur de haut niveau

```
CALL MQBACK,(HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Appel Visual Basic

```
MQBACK Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQBEGIN-Début de l'unité de travail

L'appel MQBEGIN démarre une unité de travail coordonnée par le gestionnaire de files d'attente et qui peut impliquer des gestionnaires de ressources externes.

Syntaxe

MQBEGIN (*Hconn*, *BeginOptions*, *Code achèvement*, *Motif*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Hconn doit être un descripteur de connexion non partagé. Si un descripteur de connexion partagée est spécifié, l'appel échoue avec le code anomalie MQRC_HCONN_ERROR. Pour plus d'informations sur les descripteurs partagés et non partagés, voir la description des options MQCNO_HANDLE_SHARE_ * dans «MQCNO-Options de connexion», à la page 298 .

BeginOptions
Type: MQBO-entrée / sortie

Il s'agit d'options qui contrôlent l'action de MQBEGIN, comme décrit dans «MQBO-Options de début», à la page 259.

Si aucune option n'est requise, les programmes écrits en assembleur C ou S/390 peuvent spécifier une adresse de paramètre null au lieu de spécifier l'adresse d'une structure MQBO.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_NO_EXTERNAL_PARTICIPANTS
(2121, X'849') Aucun gestionnaire de ressources participant enregistré.

MQRC_PARTICIPANT_NOT_XX_ENCODE_CASE_ONE disponible
(2122, X'84A') Gestionnaire de ressources participant non disponible.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_BO_ERREUR
(2134, X'856') Structure des options de début incorrecte.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Appel non valide dans l'environnement.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

MQRC_UOW_EN_COURS
(2128, X'850') Unité de travail déjà démarrée.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie.

Notes d'utilisation

1. Utilisez l'appel MQBEGIN pour démarrer une unité de travail coordonnée par le gestionnaire de files d'attente et qui peut impliquer des modifications apportées aux ressources appartenant à d'autres gestionnaires de ressources. Le gestionnaire de files d'attente prend en charge trois types d'unité de travail:
   - **Unité de travail locale coordonnée par le gestionnaire de files d'attente**: unité de travail dans laquelle le gestionnaire de files d'attente est le seul gestionnaire de ressources participant, de sorte que le gestionnaire de files d'attente joue le rôle de coordinateur d'unité de travail.
     - Pour démarrer ce type d'unité de travail, spécifiez l'option MQPMO_SYNCPOINT ou MQGMO_SYNCPOINT sur le premier appel MQPUT, MQPUT1ou MQGET dans l'unité de travail.
     - Pour valider ou rétablir ce type d'unité de travail, utilisez l'appel MQCMIT ou MQBACK.
   - **Unité de travail globale coordonnée par le gestionnaire de files d'attente**: unité de travail dans laquelle le gestionnaire de files d'attente agit en tant que coordinateur d'unité de travail, à la fois pour les ressources MQ *et* pour les ressources appartenant à d'autres gestionnaires de ressources. Ces gestionnaires de ressources coopèrent avec le gestionnaire de files d'attente pour s'assurer que toutes les modifications apportées aux ressources de l'unité d'oeuvre sont validées ou annulées ensemble.

  - Pour démarrer ce type d'unité de travail, utilisez l'appel MQBEGIN.
  - Pour valider ou rétablir ce type d'unité d'oeuvre, utilisez les appels MQCMIT et MQBACK.
- **Unité de travail globale coordonnée en externe**: unité de travail dans laquelle le gestionnaire de files d'attente est un participant, mais le gestionnaire de files d'attente n'agit pas en tant que coordinateur d'unité de travail. A la place, il existe un coordinateur d'unité de travail externe avec lequel le gestionnaire de files d'attente coopère.
  - Pour démarrer ce type d'unité de travail, utilisez l'appel approprié fourni par le coordinateur de l'unité de travail externe.

    Si l'appel MQBEGIN est utilisé pour tenter de démarrer l'unité de travail, l'appel échoue avec le code anomalie MQRC_ENVIRONMENT_ERROR.
  - Pour valider ou exécuter ce type d'unité de travail, utilisez les appels de validation et d'exécution fournis par le coordinateur d'unité de travail externe.

    Si vous utilisez l'appel MQCMIT ou MQBACK pour valider ou rétablir l'unité d'oeuvre, l'appel échoue avec le code anomalie MQRC_ENVIRONMENT_ERROR.

2. Si l'application se termine avec des modifications non validées dans une unité de travail, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus de détails, voir les remarques sur l'utilisation dans «MQDISC-Déconnexion du gestionnaire de files d'attente», à la page 669 .
3. Une application ne peut participer qu'à une seule unité de travail à la fois. L'appel MQBEGIN échoue avec le code anomalie MQRC_UOW_IN_PROGRESS s'il existe déjà une unité de travail pour l'application, quel que soit le type d'unité de travail.
4. L'appel MQBEGIN n'est pas valide dans un environnement client MQ MQI. Une tentative d'utilisation de l'appel échoue avec le code anomalie MQRC_ENVIRONMENT_ERROR.
5. Lorsque le gestionnaire de files d'attente joue le rôle de coordinateur d'unité de travail pour les unités de travail globales, les gestionnaires de ressources pouvant participer à l'unité de travail sont définis dans le fichier de configuration du gestionnaire de files d'attente.
6. Sous IBM i, les trois types d'unité de travail sont pris en charge comme suit:
   - **Unité de travail locale coordonnée par le gestionnaire de files d'attente** peut être utilisée uniquement lorsqu'une définition de validation n'existe pas au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre CMTSCOPE(*JOB) ne doit pas avoir été émise pour le travail.
   - **L'unité d'oeuvre globale coordonnée par le gestionnaire de files d'attente** n'est pas prise en charge.
   - L' **unité d'oeuvre globale coordonnée en externe** ne peut être utilisée que lorsqu'une définition de validation existe au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre CMTSCOPE(*JOB) doit avoir été émise pour le travail. Si tel est le cas, les opérations IBM i COMMIT et ROLLBACK s'appliquent aux ressources MQ ainsi qu'aux ressources appartenant à d'autres gestionnaires de ressources participants.

Appel C

```
MQBEGIN (Hconn, &BeginOptions, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;         /* Connection handle */
MQBO     BeginOptions;  /* Options that control the action of MQBEGIN */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

Appel COBOL

```
     CALL 'MQBEGIN' USING HCONN, BEGINOPTIONS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQBEGIN (Hconn, BeginOptions, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel Visual Basic

```
MQBEGIN Hconn, BeginOptions, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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-Conversion de la mémoire tampon en descripteur de message

L'appel de fonction MQBUFMH convertit une mémoire tampon en descripteur de message et est l'inverse de l'appel MQMHBUF.

Cet appel prend un descripteur de message et des propriétés MQRFH2 dans la mémoire tampon et les rend disponibles via un descripteur de message. Les propriétés MQRFH2 des données de message sont éventuellement supprimées. Les zones *Encoding*, *CodedCharSetId*et *Format* du descripteur de message sont mises à jour, si nécessaire, pour décrire correctement le contenu de la mémoire tampon après la suppression des propriétés.

Syntaxe

MQBUFMH (*Hconn*, *Hmsg*, *BufMsgHOpts* , *MsgDesc*, *Buffer*, *BufferLength*, *DataLength*, *Compcode* , *Cause*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre *Hmsg* .

Si le descripteur de message a été créé à l'aide de MQHC_UNASSOCIATED_HCONN, une connexion valide doit être établie sur l'unité d'exécution qui convertit une mémoire tampon en descripteur de message. Si aucune connexion valide n'est établie, l'appel échoue avec MQRC_CONNECTION_BROKEN.

Msg

Type: MQHMQSG-entrée

Il s'agit du descripteur de message pour lequel une mémoire tampon est requise. La valeur a été renvoyée par un appel MQCRTMH précédent.

BufMsgHOpts

Type: MQBMHO-entrée

La structure MQBMHO permet aux applications de spécifier des options qui contrôlent la façon dont les descripteurs de message sont produits à partir des mémoires tampon.

Voir «MQBMHO-Options de traitement de la mémoire tampon vers le message», à la page 257 pour plus de détails.

MsgDesc

Type : MQMD - entrée/sortie

La structure *MsgDesc* contient les propriétés de descripteur de message et décrit le contenu de la zone tampon.

En sortie de l'appel, les propriétés sont éventuellement supprimées de la zone tampon et, dans ce cas, le descripteur de message est mis à jour pour décrire correctement la zone tampon.

Les données de cette structure doivent être dans le jeu de caractères et le codage de l'application.

BufferLength

Type : MQLONG - entrée

BufferLength est la longueur de la zone tampon, en octets.

Une valeur *BufferLength* de zéro octet est valide et indique que la zone tampon ne contient aucune donnée.

Tampon

Type: MQBYTExBufferLength-input/output

Il s'agit d'options qui contrôlent l'action de MQBEGIN, comme décrit dans «MQBEGIN-Début de l'unité de travail», à la page 615.

Buffer définit la zone contenant la mémoire tampon de messages. Pour la plupart des données, vous devez aligner la mémoire tampon sur une limite de 4 octets.

Si *Buffer* contient des données de type caractère ou numérique, définissez les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* sur les valeurs appropriées aux données ; cela permet la conversion des données, si nécessaire.

Si des propriétés sont trouvées dans la mémoire tampon de messages, elles sont éventuellement supprimées ; elles sont ensuite disponibles à partir du descripteur de message lors du retour de l'appel.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide, ce qui signifie que l'adresse de tout type de données peut être spécifiée comme paramètre.

Si le paramètre *BufferLength* a pour valeur zéro, il n'est pas fait référence à *Buffer* ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

DataLength
Type : MQLONG - sortie

Longueur, en octets, de la mémoire tampon dont les propriétés peuvent être supprimées.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Carte non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BMHO_ERREUR
(2489, X'09B9') La structure des options de la mémoire tampon vers le descripteur de message n'est pas valide.

MQRC_BUFFER_ERROR
(2004, X'07D4') Paramètre de mémoire tampon non valide.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Paramètre de longueur de mémoire tampon non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

ERREUR MQRC_HMSG_ERROR
(2460, X'099C') Descripteur de message non valide.

MQRC_MD_ERROR
(2026, X'07EA') Descripteur de message non valide.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Descripteur de message déjà utilisé.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Options non valides ou non cohérentes.

MQRC_RFH_ERREUR
(2334, X'091E') Structure MQRFH2 non valide.

MQRC_RFH_FORMAT_ERREUR
(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

Les appels MQBUFMH ne peuvent pas être interceptés par des exits API-une mémoire tampon est convertie en descripteur de message dans l'espace d'application ; l'appel n'atteint pas le gestionnaire de files d'attente.

Appel C

```
MQBUFMH (Hconn, Hmsg, &BufMsgHOpts, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQBUFMH' USING HCONN, HMSG, BUFMSGHOPTS, MSGDESC, BUFFERLENGTH,
                           BUFFER, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQBUFMH (Hconn, Hmsg, BufMsgHOpts, MsgDesc, BufferLength, Buffer,
              DataLength, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQBUFMH,(HCONN,HMSG,BUFMSGHOPTS,MSGDESC,BUFFERLENGTH,BUFFER,
               DATALENGTH,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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-Gestion des rappels

L'appel MQCB enregistre un rappel pour le descripteur d'objet spécifié et contrôle l'activation et les modifications apportées au rappel.

Un rappel est un élément de code (spécifié comme le nom d'une fonction qui peut être liée dynamiquement ou comme pointeur de fonction) qui est appelé par IBM WebSphere MQ lorsque certains événements se produisent.

Pour utiliser MQCB et MQCTL sur un client V7 , vous devez être connecté à un serveur V7 et le paramètre **SHARECNV** du canal doit avoir une valeur différente de zéro.

Les types de rappel pouvant être définis sont les suivants:

Consommateur de message
Une fonction de rappel de consommateur de message est appelée lorsqu'un message répondant aux critères de sélection spécifiés est disponible sur un descripteur d'objet.

Une seule fonction de rappel peut être enregistrée pour chaque descripteur d'objet. Si une seule file d'attente doit être lue avec plusieurs critères de sélection, la file d'attente doit être ouverte plusieurs fois et une fonction de consommateur doit être enregistrée sur chaque descripteur.

Gestionnaire d'événements
Le gestionnaire d'événements est appelé pour les conditions qui affectent l'ensemble de l'environnement de rappel.

La fonction est appelée lorsqu'une condition d'événement se produit, par exemple, l'arrêt ou la mise au repos d'un gestionnaire de files d'attente ou d'une connexion.

La fonction n'est pas appelée pour des conditions spécifiques à un consommateur de message unique, par exemple MQRC_GET_INHIBITED; elle est toutefois appelée si une fonction de rappel ne se termine pas normalement.

Syntaxe

MQCB (*Hconn*, *Opération*, *CallbackDesc*, *Hobj*, *MsgDesc* , *GetMsgOpts*, *CompCode*, *Motif*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications qui s'exécutent en mode compatibilité, vous pouvez spécifier la valeur spéciale suivante pour *MQHC_DEF_HCONN* afin d'utiliser le descripteur de connexion associé à cette unité d'exécution.

Opération
Type : MQLONG - entrée

Opération en cours de traitement sur le rappel défini pour le descripteur d'objet spécifié. Vous devez spécifier l'une des options suivantes ; si plusieurs options sont requises, les valeurs peuvent être:

- ajoutées les unes aux autres (n'ajoutez pas la même constante plusieurs fois), ou
- combinées par le biais de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations par bit).

MQOP_REGISTER
Définissez la fonction de rappel pour le descripteur d'objet spécifié. Cette opération définit la fonction à appeler et les critères de sélection à utiliser.

Si une fonction de rappel est déjà définie pour le descripteur d'objet, la définition est remplacée. Si une erreur est détectée lors du remplacement du rappel, la fonction est désenregistrée.

Si un rappel est enregistré dans la même fonction de rappel dans laquelle il a été précédemment désenregistré, il est traité comme une opération de remplacement ; les appels initiaux ou finaux ne sont pas appelés.

Vous pouvez utiliser MQOP_REGISTER avec MQOP_SUSPEND ou MQOP_RESUME.

MQOP_DÉSENREGISTREMENT
Arrête la consommation de messages pour le descripteur d'objet et supprime le descripteur de ceux qui sont éligibles pour un rappel.

Un rappel est automatiquement désenregistré si le descripteur associé est fermé.

Si MQOP_REGISISTER est appelé à partir d'un consommateur et qu'un appel d'arrêt est défini pour le rappel, il est appelé lors du retour du consommateur.

Si cette opération est émise sur un *Hobj* sans consommateur enregistré, l'appel est renvoyé avec MQRC_CALLBACK_NOT_REGISTERED.

MQOP_SUSPENSION
Interrompt la consommation de messages pour le descripteur d'objet.

Si cette opération est appliquée à un gestionnaire d'événements, le gestionnaire d'événements n'obtient pas d'événements lorsqu'il est suspendu et les événements manqués lorsqu'il est à l'état suspendu ne sont pas fournis à l'opération lors de sa reprise.

En cas d'interruption, la fonction de consommateur continue d'obtenir les rappels de type de contrôle.

MQOP_RESUME
Reprenez la consommation de messages pour le descripteur d'objet.

Si cette opération est appliquée à un gestionnaire d'événements, le gestionnaire d'événements n'obtient pas d'événements lorsqu'il est suspendu et les événements manqués lorsqu'il est à l'état suspendu ne sont pas fournis à l'opération lors de sa reprise.

CallbackDesc
Type: MQCBD-entrée

Il s'agit d'une structure qui identifie la fonction de rappel qui est enregistrée par l'application et les options utilisées lors de son enregistrement.

Pour plus de détails sur la structure, voir MQCBD .

Le descripteur de rappel est requis uniquement pour l'option MQOP_REGISTER ; si le descripteur n'est pas requis, l'adresse de paramètre transmise peut être null.

Hobj
Type : MQHOBJ - entrée

Ce descripteur représente l'accès qui a été établi à l'objet à partir duquel un message doit être consommé. Il s'agit d'un descripteur qui a été renvoyé à partir d'un précédent appel MQOPEN ou MQSUB (dans le paramètre *Hobj*).

Hobj n'est pas requis lors de la définition d'une routine de gestionnaire d'événements (MQCBT_EVENT_HANDLER) et doit être spécifié en tant que MQHO_NONE.

Si *Hobj* a été renvoyé par un appel MQOPEN, la file d'attente doit avoir été ouverte avec une ou plusieurs des options suivantes:

- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE
- MQOO_INPUT_AS_Q_DEF
- MQOO_BROWSE

MsgDesc
Type: MQMD-entrée

Cette structure décrit les attributs du message requis ainsi que ceux du message récupéré.

Le paramètre *MsgDesc* définit les attributs des messages requis par le consommateur et la version du MQMD à transmettre au consommateur de message.

MsgId, *CorrelId*, *GroupId* , *MsgSeqNumber* et *Offset* dans le MQMD sont utilisés pour la sélection des messages, en fonction des options spécifiées dans le paramètre *GetMsgOpts* .

Encoding et *CodedCharSetId* sont utilisés pour la conversion de message si vous spécifiez l'option MQGMO_CONVERT.

Pour plus de détails, voir MQMD .

MsgDesc est utilisé pour MQOP_REGISTER et si vous avez besoin de valeurs autres que celles par défaut pour les zones. *MsgDesc* n'est pas utilisé pour un gestionnaire d'événements.

Si le descripteur n'est pas requis, l'adresse de paramètre transmise peut être null.

Notez que si plusieurs consommateurs sont enregistrés dans la même file d'attente avec des sélecteurs qui se chevauchent, le consommateur choisi pour chaque message n'est pas défini.

GetMsgOpts
Type: MQGMO-input

Le paramètre *GetMsgOpts* contrôle la façon dont le consommateur de message obtient les messages. Toutes les options de ce paramètre ont la signification décrite dans «Options MQGMO-Get-message», à la page 345, lorsqu'elles sont utilisées sur un appel MQGET, sauf:

MQGMO_SET_SIGNAL
Cette option n'est pas autorisée.

MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXT, MQGMO_MARK_ *
L'ordre des messages distribués à un consommateur de navigation est dicté par les combinaisons de ces options. Les combinaisons significatives sont les suivantes:

MQGMO_BROWSE_FIRST
Le premier message de la file d'attente est distribué de manière répétée au consommateur. Cela est utile lorsque le consommateur consomme de façon destructive le message dans le rappel. Utilisez cette option avec précaution.

MQGMO_BROWSE_NEXT
Le destinataire reçoit chaque message de la file d'attente, de la position actuelle du curseur jusqu'à la fin de la file d'attente.

MQGMO_BROWSE_FIRST + MQGMO_BROWSE_NEXT
Le curseur est réinitialisé au début de la file d'attente. Le destinataire reçoit ensuite chaque message jusqu'à ce que le curseur atteigne la fin de la file d'attente.

MQGMO_BROWSE_FIRST + MQGMO_MARK_ *
A partir du début de la file d'attente, le consommateur reçoit le premier message non marqué de la file d'attente, qui est ensuite marqué pour ce consommateur. Cette combinaison permet de s'assurer que le consommateur peut recevoir de nouveaux messages ajoutés derrière le point de curseur en cours.

MQGMO_BROWSE_NEXT + MQGMO_MARK_ *
A partir de la position du curseur, le consommateur reçoit le prochain message non marqué dans la file d'attente, qui est ensuite marqué pour ce consommateur. Utilisez cette combinaison avec précaution car des messages peuvent être ajoutés à la file d'attente derrière la position actuelle du curseur.

MQGMO_BROWSE_FIRST + MQGMO_BROWSE_NEXT + MQGMO_MARK_ *
Cette combinaison n'est pas autorisée. S'il est utilisé, l'appel renvoie MQRC_OPTIONS_ERROR.

MQGMO_NO_WAIT, MQGMO_WAIT et WaitInterval
Ces options contrôlent la façon dont le consommateur est appelé.

MQGMO_NO_WAIT
Le consommateur n'est jamais appelé avec MQRC_NO_MSG_AVAILABLE. Le consommateur n'est appelé que pour les messages et les événements.

MQGMO_WAIT avec la valeur zéro WaitInterval
Le code MQRC_NO_MSG_AVAILABLE est transmis au destinataire lorsqu'aucun message n'est disponible et que le destinataire a été démarré ou qu'il a reçu au moins un message depuis le dernier code anomalie "aucun message".

Cela empêche le consommateur de s'interroger dans une boucle occupée lorsqu'un intervalle d'attente de zéro est spécifié.

MQGMO_WAIT et un WaitInterval positif
Le consommateur est appelé après l'intervalle d'attente spécifié avec le code anomalie MQRC_NO_MSG_AVAILABLE. Cet appel est effectué indépendamment du fait que des messages aient été distribués ou non au consommateur. Cela permet à l'utilisateur d'effectuer un traitement de signal de présence ou de type de lot.

MQGMO_WAIT et WaitInterval de MQWI_UNLIMITED
Indique une attente infinie avant de renvoyer MQRC_NO_MSG_AVAILABLE. Le consommateur n'est jamais appelé avec MQRC_NO_MSG_AVAILABLE.

GetMsgOpts est utilisé uniquement pour MQOP_REGISTER et si vous avez besoin de valeurs autres que la valeur par défaut pour les zones. *GetMsgOpts* n'est pas utilisé pour un gestionnaire d'événements.

Si les *GetMsgOpts* ne sont pas obligatoires, l'adresse de paramètre transmise peut être null. L'utilisation de ce paramètre revient à spécifier MQGMO_DEFAULT avec MQGMO_FAIL_IF_QUIESCING.

Si un descripteur de propriétés de message est fourni dans la structure MQGMO, une copie est fournie dans la structure MQGMO qui est transmise au rappel de consommateur. En cas de retour de l'appel MQCB, l'application peut supprimer le descripteur de propriétés de message.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Les codes anomalie de la liste suivante sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre *Reason* .

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855') Impossible de charger les modules de service de conversion de données.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CALLBACK_LINK_ERROR
(2487, X'9B7') Zone de type de rappel incorrecte.

MQRC_CALLBACK_NOT_REGISTERED
(2448, X' 990') Impossible de désenregistrer, de suspendre ou de reprendre car il n'y a pas de rappel enregistré.

MQRC_CALLBACK_ROUTINE_ERROR
(2486, X'9B6') *CallbackFunction* ou *CallbackName* doit être spécifié, mais pas les deux.

MQRC_CALLBACK_TYPE_ERREUR
(2483, X'9B3') Zone de type de rappel incorrecte.

MQRC_CBD_OPTIONS_ERREUR
(2484, X'9B4') Zone d'options MQCBD incorrecte.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Connexion en cours de mise au repos.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_CORREL_ID_ERROR
(2207, X'89F') Erreur d'identificateur de corrélation.

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Paramètre de longueur des données non valide.

MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

MQRC_GET_INHIBITED
(2016, X'7E0') Se bloque pour la file d'attente.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Conflit entre les unités d'oeuvre globales.

MQRC_GMO_ERROR
(2186, X'88A') Structure des options de message Get non valide.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HOBJ_ERROR
(2019, X'7E3') Descripteur d'objet non valide.

MQRC_INCONSISTENT_BROWSE
(2259, X'8D3') Spécification de navigation incohérente.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

MQRC_INVALID_MSG_UNDER_CURSOR
(2246, X'8C6') Message sous le curseur non valide pour la récupération.

MQRC_LOCAL_UOW_CONFLICT
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Options de correspondance non valides.

MQRC_MAX_MSG_LENGTH_ERROR
(2485, X'9B4') Zone *MaxMsgLength* incorrecte.

MQRC_MD_ERROR
(2026, X'7EA') Descripteur de message non valide.

MQRC_MODULE_ENTRY_NOT_FOUND
(2497, X'9C1') Le point d'entrée de fonction spécifié est introuvable dans le module.

MQRC_MODULE_INVALID
(2496, X'9C0') Le module a été trouvé, mais son type est incorrect ; il ne s'agit pas d'un module 32 bits, 64 bits ou d'une bibliothèque de liens dynamiques valide.

MQRC_MODULE_NOT_FOUND
(2495, X'9BF') Module introuvable dans le chemin de recherche ou non autorisé à être chargé.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Numéro de séquence de message non valide.

MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Utilisation du jeton de message non valide.

MQRC_NO_MSG_AVAILABLE
(2033, X'7F1') Aucun message disponible.

MQRC_NO_MSG_UNDER_CURSOR
(2034, X'7F2') Curseur d'exploration non positionné sur le message.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') File d'attente non ouverte pour exploration.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') File d'attente non ouverte pour saisie.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

ERREUR D'OPERATION_MQRC
(2206, X'89E') Code d'opération incorrect sur l'appel API.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_Q_DELETED
(2052, X'804') File d'attente supprimée.

MQRC_Q_INDEX_TYPE_ERROR
(2394, X'95A') Type d'index incorrect pour la file d'attente.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_QUIESCING
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SIGNAL_OUTSTANDING
(2069, X'815') Signal en attente pour ce descripteur.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_SYNCPOINT_LIMIT_REACHED
(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818') Prise en charge du point de synchronisation non disponible.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Intervalle d'attente non valide dans MQGMO.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Version MQGMO fournie non valide.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Version MQMD fournie non valide.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. MQCB permet de définir l'action à appeler pour chaque message, correspondant aux critères spécifiés, disponible dans la file d'attente. Lorsque l'action est traitée, le message est supprimé de la file d'attente et transmis au consommateur de message défini, ou un jeton de message est fourni, qui est utilisé pour extraire le message.
2. MQCB peut être utilisé pour définir des routines de rappel avant de démarrer la consommation avec MQCTL ou à partir d'une routine de rappel.
3. Pour utiliser MQCB en dehors d'une routine de rappel, vous devez d'abord interrompre la consommation des messages à l'aide de MQCTL et reprendre la consommation par la suite.
4. MQCB n'est pas pris en charge dans l'adaptateur IMS .

Séquence de rappel de consommateur de message

Vous pouvez configurer un destinataire pour qu'il appelle le rappel à des points clés au cours du cycle de vie du destinataire. Exemple :

- lorsque le consommateur est enregistré pour la première fois,
- lorsque la connexion est démarrée,
- lorsque la connexion est arrêtée et
- lorsque le consommateur est désenregistré, soit explicitement, soit implicitement par un MQCLOSE.

Tableau 562. Définitions d'instruction MQCTL

Verbe	**Explication**
MQCTL (DEMARRAGE)	Appel MQCTL à l'aide de l'opération MQOP_START
MQCTL (STOP)	Appel MQCTL à l'aide de l'opération MQOP_STOP
MQCTL (en attente)	Appel MQCTL à l'aide de l'opération MQOP_START_WAIT

Cela permet au consommateur de conserver l'état associé au consommateur. Lorsqu'un rappel est demandé par une application, les règles d'appel du consommateur sont les suivantes:

ENREGISTRER
Est toujours le premier type d'appel du rappel.

Est toujours appelé sur la même unité d'exécution que l'appel MQCB (REGISTER).

DEBUT
Est toujours appelé de manière synchrone avec l'instruction MQCTL (START).

- Tous les rappels START sont terminés avant que l'instruction MQCTL (START) ne soit renvoyée.

Se trouve sur la même unité d'exécution que la distribution des messages si THREAD_AFFINITY est demandé.

L'appel avec démarrage n'est pas garanti si, par exemple, un rappel précédent émet MQCTL (STOP) pendant MQCTL (START).

ARRETER
Aucun autre message ou événement n'est distribué après cet appel tant que la connexion n'est pas redémarrée.

Une commande STOP est garantie si l'application a été précédemment appelée pour la commande START, un message ou un événement.

ANNULER l'enregistrement
Est toujours le dernier type d'appel du rappel.

Assurez-vous que votre application effectue une initialisation et un nettoyage basés sur les unités d'exécution dans les rappels START et STOP. Vous pouvez effectuer une initialisation et un nettoyage non basés sur les unités d'exécution avec les rappels REGISTER et REGISISTER.

Ne faites aucune hypothèse sur la durée de vie et la disponibilité de l'unité d'exécution autre que celle qui est indiquée. Par exemple, ne vous fiez pas à une unité d'exécution qui reste active après le dernier appel à REGISTERISTER. De même, lorsque vous avez choisi de ne pas utiliser THREAD_AFFINITY, ne supposez pas que l'unité d'exécution existe à chaque fois que la connexion est démarrée.

Si votre application a des exigences particulières en matière de caractéristiques d'unité d'exécution, elle peut toujours créer une unité d'exécution en conséquence, puis utiliser MQCTL (WAIT). Cela a pour effet de'donner'l'unité d'exécution à IBM WebSphere MQ pour la distribution asynchrone des messages.

Utilisation de la connexion de consommateur de message

Vous pouvez configurer un destinataire pour qu'il appelle le rappel à des points clés au cours du cycle de vie du destinataire. Exemple :

- lorsque le consommateur est enregistré pour la première fois,
- lorsque la connexion est démarrée,
- lorsque la connexion est arrêtée et
- lorsque le consommateur est désenregistré, soit explicitement, soit implicitement par un MQCLOSE.

Tableau 563. Définitions d'instruction MQCTL

Verbe	**Explication**
MQCTL (DEMARRAGE)	Appel MQCTL à l'aide de l'opération MQOP_START
MQCTL (STOP)	Appel MQCTL à l'aide de l'opération MQOP_STOP
MQCTL (en attente)	Appel MQCTL à l'aide de l'opération MQOP_START_WAIT

Cela permet au consommateur de conserver l'état associé au consommateur. Lorsqu'un rappel est demandé par une application, les règles d'appel du consommateur sont les suivantes:

ENREGISTRER
Est toujours le premier type d'appel du rappel.

Est toujours appelé sur la même unité d'exécution que l'appel MQCB (REGISTER).

DEBUT
Est toujours appelé de manière synchrone avec l'instruction MQCTL (START).

- Tous les rappels START sont terminés avant que l'instruction MQCTL (START) ne soit renvoyée.

Se trouve sur la même unité d'exécution que la distribution des messages si THREAD_AFFINITY est demandé.

L'appel avec démarrage n'est pas garanti si, par exemple, un rappel précédent émet MQCTL (STOP) pendant MQCTL (START).

ARRETER
Aucun autre message ou événement n'est distribué après cet appel tant que la connexion n'est pas redémarrée.

Une commande STOP est garantie si l'application a été précédemment appelée pour la commande START, un message ou un événement.

ANNULER l'enregistrement
Est toujours le dernier type d'appel du rappel.

Assurez-vous que votre application effectue une initialisation et un nettoyage basés sur les unités d'exécution dans les rappels START et STOP. Vous pouvez effectuer une initialisation et un nettoyage non basés sur les unités d'exécution avec les rappels REGISTER et REGISISTER.

Ne faites aucune hypothèse sur la durée de vie et la disponibilité de l'unité d'exécution autre que celle qui est indiquée. Par exemple, ne vous fiez pas à une unité d'exécution qui reste active après le dernier appel à REGISTERISTER. De même, lorsque vous avez choisi de ne pas utiliser THREAD_AFFINITY, ne supposez pas que l'unité d'exécution existe à chaque fois que la connexion est démarrée.

Si votre application a des exigences particulières en matière de caractéristiques d'unité d'exécution, elle peut toujours créer une unité d'exécution en conséquence, puis utiliser MQCTL (WAIT). Cela a pour effet de'donner'l'unité d'exécution à IBM WebSphere MQ pour la distribution asynchrone des messages.

Appel C

```
MQCB (Hconn, Operation, CallbackDesc, Hobj, MsgDesc,
GetMsgOpts, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQCB' USING HCONN, OPERATION, CBDESC, HOBJ, MSGDESC,
            GETMSGOPTS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQCB(Hconn, Operation, CallbackDesc, Hobj, MsgDesc, GetMsgOpts,
          CompCode, Reason)
```

Déclarez les paramètres comme suit :

```
dcl Hconn        fixed bin(31); /* Connection handle */
dcl Operation    fixed bin(31); /* Operation */
dcl CallbackDesc like MQCBD;    /* Callback Descriptor */
dcl Hobj         fixed bin(31); /* Object Handle */
dcl MsgDesc      like MQMD;     /* Message Descriptor */
dcl GetMsgOpts   like MQGMO;    /* Get Message Options */
dcl CompCode     fixed bin(31); /* Completion code */
dcl Reason       fixed bin(31); /* Reason code qualifying CompCode */
```

MQCB_FUNCTION-Fonction de rappel

L'appel de fonction MQCB_FUNCTION est la fonction de rappel pour la gestion des événements et la consommation des messages asynchrones.

La définition d'appel MQCB_FUNCTION est fournie uniquement pour décrire les paramètres qui sont transmis à la fonction de rappel. Aucun point d'entrée appelé MQCB_FUNCTION n'est fourni par le gestionnaire de files d'attente.

La spécification de la fonction réelle à appeler est une entrée de l'appel MQCB et est transmise via la structure MQCBD .

Syntaxe

MQCB_FUNCTION (*Hconn*, *MsgDesc*, *GetMsgOpts*, *Buffer*, *Contexte*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent. Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour Hconn:

MQHC_DEF_CONN

Descripteur de connexion par défaut.

MsgDesc
Type: MQMD-entrée

Cette structure décrit les attributs du message extrait.

Voir «MQMD-Descripteur de message», à la page 394 pour plus de détails.

La version de MQMD transmise est la même que celle transmise sur l'appel MQCB qui a défini la fonction de consommateur.

L'adresse du MQMD est transmise sous forme de caractères nuls si un MQGMO version 4 a été utilisé pour demander qu'un descripteur de message soit renvoyé à la place d'un MQMD.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

GetMsgOpts
Type: MQGMO-input

Options utilisées pour contrôler les actions du consommateur de message. Ce paramètre contient également des informations supplémentaires sur le message renvoyé.

Pour plus d'informations, voir MQGMO .

La version de MQGMO transmise est la version la plus récente prise en charge.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

Tampon
Type: MQBYTExBufferLength-input

Il s'agit de la zone contenant les données de message.

Si aucun message n'est disponible pour cet appel ou si le message ne contient aucune donnée de message, l'adresse du *Buffer* est transmise sous la forme de valeurs NULL.

Il s'agit d'une zone d'entrée de la fonction de consommateur de message ; elle n'est pas pertinente pour une fonction de gestionnaire d'événements.

Contexte
Type: MQCBC-entrée / sortie

Cette structure fournit des informations de contexte aux fonctions de rappel. Voir «MQCBC-Contexte de rappel», à la page 261 pour plus de détails.

Notes d'utilisation

1. Sachez que si vos routines de rappel utilisent des services qui peuvent retarder ou bloquer l'unité d'exécution, par exemple, MQGET avec attente, peut retarder la répartition d'autres rappels.
2. Une unité de travail distincte n'est pas automatiquement établie pour chaque appel d'une routine de rappel. Par conséquent, les routines peuvent émettre un appel de validation ou différer la validation jusqu'à ce qu'un lot logique de travail ait été traité. Lorsque le lot de travaux est validé, il valide les messages pour toutes les fonctions de rappel qui ont été appelées depuis le dernier point de synchronisation.
3. Les programmes appelés par les paramètres d'extraction CICS LINK ou CICS START à l'aide des services CICS via des objets nommés appelés conteneurs de canaux. Les noms de conteneur sont identiques aux noms de paramètre. Pour plus d'informations, voir la documentation CICS .
4. Les routines de rappel peuvent émettre un appel MQDISC, mais pas pour leur propre connexion. Par exemple, si une routine de rappel a créé une connexion, elle peut également la déconnecter.
5. En règle générale, une routine de rappel ne doit pas être appelée à chaque fois à partir de la même unité d'exécution. Si nécessaire, utilisez MQCTLO_THREAD_AFFINITY lorsque la connexion est démarrée.
6. Lorsqu'une routine de rappel reçoit un code anomalie différent de zéro, elle doit effectuer l'action appropriée.
7. MQCB_FUNCTION n'est pas pris en charge dans l'adaptateur IMS .

MQCLOSE-Fermer l'objet

L'appel MQCLOSE abandonne l'accès à un objet et est l'inverse des appels MQOPEN et MQSUB.

Syntaxe

MQCLOSE (*Hconn*, *Hobj*, *Options*, *CompCode*, *Raison*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, vous pouvez omettre l'appel MQCONN et spécifier la valeur suivante pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

Hobj
Type: MQHOBJ-entrée / sortie

Ce descripteur représente l'objet en cours de fermeture. L'objet peut être de n'importe quel type. La valeur de *Hobj* a été renvoyée par un appel MQOPEN précédent.

Une fois l'appel terminé, le gestionnaire de files d'attente définit ce paramètre sur une valeur qui n'est pas un descripteur valide pour l'environnement. Cette valeur est:

MQHO_UNUSABLE_HOBJ
Descripteur d'objet inutilisable.

Sous z/OS, *Hobj* est défini sur une valeur non définie.

OPTIONS
Type : MQLONG - entrée

Ce paramètre contrôle la façon dont l'objet est fermé.

Seules les files d'attente dynamiques permanentes et les abonnements peuvent être fermés de plusieurs manières, car ils doivent être conservés ou supprimés ; il s'agit des files d'attente avec l'attribut *DefinitionType* dont la valeur est MQQDT_PERMANENT_DYNAMIC (voir l'attribut *DefinitionType* décrit dans «Attributs des files d'attente», à la page 824). Les options de fermeture sont résumées dans cette rubrique.

Les abonnements durables peuvent être conservés ou supprimés ; ils sont créés à l'aide de l'appel MQSUB avec l'option MQSO_DURABLE.

Lors de la fermeture de l'indicateur sur une destination gérée (c'est-à-dire le paramètre *Hobj* renvoyé sur un appel MQSUB qui utilisait l'option MQSO_MANAGED), le gestionnaire de files d'attente nettoie toutes les publications qui n'ont pas été extraites lorsque l'abonnement associé a également été supprimé. L'abonnement est supprimé à l'aide de l'option MQCO_REMOVE_SUB du paramètre *Hsub* renvoyé dans un appel MQSUB. Notez que MQCO_REMOVE_SUB est le comportement par défaut sur MQCLOSE pour un abonnement non durable.

Lors de la fermeture d'un descripteur vers une destination non gérée, vous êtes responsable du nettoyage de la file d'attente dans laquelle les publications sont envoyées. Fermez d'abord l'abonnement à l'aide de MQCO_REMOVE_SUB, puis traitez les messages hors de la file d'attente jusqu'à ce qu'il n'en reste plus.

Vous ne devez spécifier qu'une seule option parmi les suivantes:

Options de file d'attente dynamique: Ces options contrôlent la manière dont les files d'attente dynamiques permanentes sont fermées.

MQCO_DELETE
La file d'attente est supprimée si l'une des conditions suivantes est vérifiée:

- Il s'agit d'une file d'attente dynamique permanente, créée par un précédent appel MQOPEN, et il n'y a aucun message dans la file d'attente et aucune demande d'extraction ou d'insertion non validée en attente pour la file d'attente (pour la tâche en cours ou toute autre tâche).
- Il s'agit de la file d'attente dynamique temporaire créée par l'appel MQOPEN qui a renvoyé *Hobj*. Dans ce cas, tous les messages de la file d'attente sont purgés.

Dans tous les autres cas, y compris le cas où *Hobj* a été renvoyé sur un appel MQSUB, l'appel échoue avec le code anomalie MQRC_OPTION_NOT_VALID_FOR_TYPE et l'objet n'est pas supprimé.

Sous z/OS, si la file d'attente est une file d'attente dynamique qui a été supprimée logiquement et qu'il s'agit du dernier descripteur associé, la file d'attente est physiquement supprimée. Pour plus d'informations, voir «Notes d'utilisation», à la page 639.

MQCO_DELETE_PURGE
La file d'attente est supprimée et tous les messages qu'elle contient sont purgés, si l'une des conditions suivantes est vérifiée:

- Il s'agit d'une file d'attente dynamique permanente, créée par un appel MQOPEN précédent, et aucune demande d'extraction ou d'insertion non validée n'est en attente pour la file d'attente (pour la tâche en cours ou toute autre tâche).
- Il s'agit de la file d'attente dynamique temporaire créée par l'appel MQOPEN qui a renvoyé *Hobj*.

Dans tous les autres cas, y compris le cas où *Hobj* a été renvoyé sur un appel MQSUB, l'appel échoue avec le code anomalie MQRC_OPTION_NOT_VALID_FOR_TYPE et l'objet n'est pas supprimé.

Le tableau indique les options de fermeture valides et indique si l'objet est conservé ou supprimé.

Type d'objet ou de file d'attente	**MQCO_AUCUN**	**MQCO_DELETE**	**MQCO_DELETE_PURGE**
Objet autre qu'une file d'attente	Conservé	Non valide	Non valide
File d'attente prédéfinie	Conservé	Non valide	Non valide
file d'attente dynamique permanente	Conservé	Supprimé s'il est vide et qu'aucune mise à jour n'est en attente	Messages supprimés ; file d'attente supprimée si aucune mise à jour n'est en attente
File d'attente dynamique temporaire (appel émis par le créateur de la file d'attente)	Supprimé	Supprimé	Supprimé
File d'attente dynamique temporaire (appel non émis par le créateur de la file d'attente)	Conservé	Non valide	Non valide
Liste de distribution	Conservé	Non valide	Non valide
Destination d'abonnement géré	Conservé	Non valide	Non valide
Liste de distribution (l'abonnement a été supprimé)	Messages supprimés ; file d'attente supprimée	Non valide	Non valide

Options de fermeture d'abonnement: ces options contrôlent si les abonnements durables sont supprimés lorsque le descripteur est fermé et si les publications en attente de lecture par l'application sont nettoyées. Ces options ne peuvent être utilisées qu'avec un descripteur d'objet renvoyé dans le paramètre *Hsub* d'un appel MQSUB.

MQCO_KEEP_SUB
Le descripteur de l'abonnement est fermé, mais l'abonnement effectué est conservé. Les publications continuent d'être envoyées à la destination spécifiée dans l'abonnement. Cette option est valide uniquement si l'abonnement a été effectué avec l'option MQSO_DURABLE.

MQCO_KEEP_SUB est la valeur par défaut si l'abonnement est durable

SOUS-TITRE_REMOVE_MQQUE
L'abonnement est supprimé et le descripteur de l'abonnement est fermé.

Le paramètre *Hobj* de l'appel MQSUB n'est pas invalidé par la fermeture du paramètre *Hsub* et peut continuer à être utilisé pour que MQGET ou MQCB reçoive les publications restantes. Lorsque le paramètre *Hobj* de l'appel MQSUB est également fermé, s'il s'agissait d'une destination gérée, les publications non extraites sont supprimées.

MQCO_REMOVE_SUB est la valeur par défaut si l'abonnement est non durable.

Ces options de clôture d'abonnement sont récapitulées dans les tableaux suivants.

Pour fermer un descripteur d'abonnement durable mais conserver l'abonnement, utilisez les options de fermeture d'abonnement suivantes:

tâche	**Option de fermeture d'abonnement**
Conserver les publications sur un descripteur MQOPENed	MQCO_KEEP_SUB
Suppression de publications sur un descripteur MQOPENed	Action non autorisée
Conserver les publications sur un descripteur MQSO_MANAGED	MQCO_KEEP_SUB
Suppression de publications sur un descripteur MQSO_MANAGED	Action non autorisée

Pour vous désabonner, en fermant un descripteur d'abonnement durable et en le désabonnant ou en fermant un descripteur d'abonnement non durable, utilisez les options de fermeture d'abonnement suivantes:

tâche	**Option de fermeture d'abonnement**
Conserver les publications sur un descripteur MQOPENed	SOUS-TITRE_REMOVE_MQQUE
Suppression de publications sur un descripteur MQOPENed	Action non autorisée
Conserver les publications sur un descripteur MQSO_MANAGED	SOUS-TITRE_REMOVE_MQQUE

Options de lecture anticipée: Les options suivantes contrôlent ce qui arrive aux messages non persistants qui ont été envoyés au client avant qu'une application ne les demande et qui n'ont pas encore été consommés par l'application. Ces messages sont stockés dans la mémoire tampon de lecture anticipée du client en attente d'être demandés par l'application et peuvent être supprimés ou consommés dans la file d'attente avant la fin de l'opération MQCLOSE.

MQCO_IMMEDIATE
L'objet est fermé immédiatement et tous les messages qui ont été envoyés au client avant qu'une application ne les demande sont supprimés et ne peuvent être consommés par aucune application. Il s'agit de la valeur par défaut.

MQCO_QUIESCE
Une demande de fermeture de l'objet est effectuée, mais si des messages qui ont été envoyés au client avant qu'une application ne les demande, résident toujours dans la mémoire tampon de lecture anticipée du client, l'appel MQCLOSE est renvoyé avec un avertissement MQRC_READ_AHEAD_MSGS et le descripteur d'objet reste valide.

L'application peut ensuite continuer à utiliser le descripteur d'objet pour extraire des messages jusqu'à ce qu'il ne soit plus disponible, puis fermer à nouveau l'objet. Aucun autre message n'est envoyé au client avant qu'une application ne le demande. La lecture anticipée est désormais désactivée.

Il est conseillé aux applications d'utiliser MQCO_QUIESCE plutôt que d'essayer d'atteindre un point où il n'y a plus de messages dans la mémoire tampon de lecture anticipée du client, car un message pourrait arriver entre le dernier appel MQGET et le MQCLOSE suivant qui serait supprimé si MQCO_IMMEDIATE était utilisé.

Si un MQCLOSE avec MQCO_QUIESCE est émis à partir d'une fonction de rappel asynchrone, le même comportement de lecture des messages à l'avance s'applique. Si l'avertissement MQRC_READ_AHEAD_MSGS est renvoyé, la fonction de rappel est appelée au moins une fois de plus. Lorsque le dernier message restant ayant fait l'objet d'une lecture anticipée a été transmis à la fonction de rappel, la zone ConsumerFlags de MQCBC est définie sur MQCBCF_READA_BUFFER_EMPTY.

Option par défaut: Si vous n'avez besoin d'aucune des options décrites ci-dessus, vous pouvez utiliser l'option suivante:

MQCO_AUCUN
Aucun traitement de fermeture facultatif n'est requis.

Cet élément *doit* être spécifié pour:

- Objets autres que des files d'attente
- Files d'attente prédéfinies
- Files d'attente dynamiques temporaires (mais uniquement dans les cas où *Hobj* n'est *pas* le descripteur renvoyé par l'appel MQOPEN qui a créé la file d'attente).
- Listes de diffusion

Dans tous les cas ci-dessus, l'objet est conservé et non supprimé.

Si cette option est spécifiée pour une file d'attente dynamique temporaire:

- La file d'attente est supprimée si elle a été créée par l'appel MQOPEN qui a renvoyé *Hobj*; tous les messages qui se trouvent dans la file d'attente sont purgés.
- Dans tous les autres cas, la file d'attente (et tous les messages qu'elle contient) est conservée.

Si cette option est spécifiée pour une file d'attente dynamique permanente, la file d'attente est conservée et n'est pas supprimée.

Sous z/OS, si la file d'attente est une file d'attente dynamique qui a été supprimée logiquement et qu'il s'agit du dernier descripteur associé, la file d'attente est physiquement supprimée. Pour plus d'informations, voir «Notes d'utilisation», à la page 639.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Les codes anomalie répertoriés sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre *Reason*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Groupe de messages non complet.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Message logique non complet.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CF_STRUC_FAILED
(2373, X'945') Echec de la structure de l'unité de couplage.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC__DB2_NOT_AVAILABLE
(2342, X' 926') Le sous-système Db2 n'est pas disponible.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HOBJ_ERROR
(2019, X'7E3') Descripteur d'objet non valide.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

MQRC_OPTION_NOT_VALID_FOR_TYPE
(2045, X'7FD') Dans un appel MQOPEN ou MQCLOSE: option non valide pour le type d'objet.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_Q_NOT_EMPTY
(2055, X'807') La file d'attente contient un ou plusieurs messages ou des demandes d'insertion ou d'extraction non validées.

MQRC_READ_AHEAD_MSGS
(nnnn, X'xxx') Le client a lu des messages d'avance qui n'ont pas encore été consommés par l'application.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SECURITY_ERROR (ERREUR DE SECURITE MQ)
(2063, X'80F') Une erreur de sécurité s'est produite.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. Lorsqu'une application émet l'appel MQDISC ou se termine normalement ou anormalement, tous les objets qui ont été ouverts par l'application et qui sont toujours ouverts sont fermés automatiquement avec l'option MQCO_NONE.
2. Les points suivants s'appliquent si l'objet en cours de fermeture est une *file d'attente*:
   - Si les opérations sur la file d'attente sont effectuées dans le cadre d'une unité de travail, la file d'attente peut être fermée avant ou après l'exécution du point de synchronisation sans affecter le résultat du point de synchronisation. Si la file d'attente est déclenchée, l'exécution d'une annulation avant la fermeture de la file d'attente peut entraîner l'émission d'un message de déclenchement. Pour plus d'informations sur les messages de déclenchement, voir Propriétés des messages de déclenchement.
   - Si la file d'attente a été ouverte avec l'option MQOO_BROWSE, le curseur de navigation est détruit. Si la file d'attente est rouverte avec l'option MQOO_BROWSE, un nouveau curseur de navigation est créé (voir MQOO_BROWSE).
   - Si un message est actuellement verrouillé pour ce descripteur au moment de l'appel MQCLOSE, le verrou est libéré (voir MQGMO_LOCK).
   - Sous z/OS, s'il existe une demande MQGET avec l'option MQGMO_SET_SIGNAL en attente alors que le descripteur de file d'attente est en cours de fermeture, la demande est annulée (voir MQGMO_SET_SIGNAL). Les demandes de signal pour la même file d'attente mais déposées sur des descripteurs différents (*Hobj*) ne sont pas affectées (sauf si une file d'attente dynamique est supprimée, auquel cas elles sont également annulées).
3. Les points suivants s'appliquent si l'objet en cours de fermeture est une *file d'attente dynamique* (permanente ou temporaire):
   - Pour une file d'attente dynamique, vous pouvez spécifier les options MQCO_DELETE et MQCO_DELETE_PURGE, quelles que soient les options spécifiées dans l'appel MQOPEN correspondant.

- Lorsqu'une file d'attente dynamique est supprimée, tous les appels MQGET avec l'option MQGMO_WAIT qui sont en attente dans la file d'attente sont annulés et le code anomalie MQRC_Q_DELETED est renvoyé. Voir MQGMO_WAIT.

  Bien que les applications ne puissent pas accéder à une file d'attente supprimée, la file d'attente n'est pas supprimée du système et les ressources associées ne sont pas libérées tant que tous les descripteurs qui font référence à la file d'attente n'ont pas été fermés et que toutes les unités de travail qui affectent la file d'attente n'ont pas été validées ou annulées.

  Sous z/OS, une file d'attente qui a été supprimée logiquement mais qui n'a pas encore été supprimée du système empêche la création d'une nouvelle file d'attente portant le même nom que la file d'attente supprimée ; l'appel MQOPEN échoue avec le code anomalie MQRC_NAME_IN_USE dans ce cas. De plus, une telle file d'attente peut toujours être affichée à l'aide des commandes MQSC, même si les applications ne peuvent pas y accéder.
- Lorsqu'une file d'attente dynamique permanente est supprimée, si le descripteur *Hobj* indiqué dans l'appel MQCLOSE n'est *pas* celui renvoyé par l'appel MQOPEN qui a créé la file d'attente, il est vérifié que l'ID utilisateur utilisé pour valider l'appel MQOPEN est autorisé à supprimer la file d'attente. Si l'option MQOO_ALTERNATE_USER_AUTHORITY a été spécifiée sur l'appel MQOPEN, l'ID utilisateur vérifié est *AlternateUserId*.

  Cette vérification n'est pas effectuée si:

  – Le descripteur indiqué est celui renvoyé par l'appel MQOPEN qui a créé la file d'attente.

  – La file d'attente en cours de suppression est une file d'attente dynamique temporaire.
- Lorsqu'une file d'attente dynamique temporaire est fermée, si le descripteur *Hobj* indiqué dans l'appel MQCLOSE est celui qui a été renvoyé par l'appel MQOPEN qui a créé la file d'attente, la file d'attente est supprimée. Cela se produit quelles que soient les options de fermeture spécifiées dans l'appel MQCLOSE. S'il y a des messages dans la file d'attente, ils sont supprimés ; aucun message de rapport n'est généré.

  Si des unités de travail non validées affectent la file d'attente, la file d'attente et ses messages sont toujours supprimés, mais les unités de travail n'échouent pas. Cependant, comme décrit ci-dessus, les ressources associées aux unités de travail ne sont pas libérées tant que chacune des unités de travail n'a pas été validée ou annulée.

4. Les points suivants s'appliquent si l'objet en cours de fermeture est une *liste de distribution*:
   - La seule option de fermeture valide pour une liste de distribution est MQCO_NONE ; l'appel échoue avec le code anomalie MQRC_OPTIONS_ERROR ou MQRC_OPTION_NOT_VALID_FOR_TYPE si d'autres options sont spécifiées.
   - Lorsqu'une liste de distribution est fermée, les codes achèvement et les codes raison individuels ne sont pas renvoyés pour les files d'attente de la liste ; seuls les paramètres *CompCode* et *Reason* de l'appel sont disponibles à des fins de diagnostic.

     Si un incident se produit lors de la fermeture de l'une des files d'attente, le gestionnaire de files d'attente poursuit le traitement et tente de fermer les files d'attente restantes de la liste de distribution. Les paramètres *CompCode* et *Reason* de l'appel sont définis pour renvoyer des informations décrivant l'échec. Il est possible que le code achèvement soit MQCC_FAILED, même si la plupart des files d'attente ont été fermées avec succès. La file d'attente qui a rencontré l'erreur n'est pas identifiée.

     S'il y a un échec dans plusieurs files d'attente, il n'est pas défini quel échec est signalé dans les paramètres *CompCode* et *Reason* .
5. Sous IBM i, si l'application a été connectée implicitement lorsque le premier appel MQOPEN a été émis, un MQDISC implicite se produit lorsque le dernier appel MQCLOSE est émis.

   Seules les applications s'exécutant en mode compatibilité peuvent être connectées implicitement ; les autres applications doivent émettre l'appel MQCONN ou MQCONNX pour se connecter explicitement au gestionnaire de files d'attente.

Appel C

```
MQCLOSE (Hconn, &Hobj, Options, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
     CALL 'MQCLOSE' USING HCONN, HOBJ, OPTIONS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQCLOSE (Hconn, Hobj, Options, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQCLOSE,(HCONN,HOBJ,OPTIONS,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Appel Visual Basic

```
MQCLOSE Hconn, Hobj, Options, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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-Validation des modifications

L'appel MQCMIT indique au gestionnaire de files d'attente que l'application a atteint un point de synchronisation et que tous les messages d'obtention et d'insertion qui se sont produits depuis le dernier point de synchronisation doivent être rendus permanents.

Les messages insérés dans une unité de travail sont mis à la disposition d'autres applications ; les messages extraits dans le cadre d'une unité de travail sont supprimés.

- Sous z/OS, l'appel est utilisé uniquement par les programmes batch (y compris les programmes batch DL/I IMS).
- Sous IBM i, cet appel n'est pas pris en charge pour les applications s'exécutant en mode compatibilité.

Syntaxe

MQCMIT (*Hconn*, *CompCode*, *Raison*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Les codes anomalie répertoriés sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre *Reason*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_BACKED_OUT
(2003, X'7D3') Unité d'oeuvre annulée.

MQRC_OUTCOME_EN attente
(2124, X'84C') Le résultat de l'opération de validation est en attente.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CALL_INTERROMPUE
(2549, X'9F5') MQPUT ou MQCMIT a été interrompu et le traitement de la reconnexion ne peut pas rétablir un résultat défini.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Appel non valide dans l'environnement.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

MQRC_OUTCOME_MIXTE
(2123, X'84B') Le résultat de l'opération de validation ou d'exclusion est mélangé.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

Echec de MQRC_RECONNECT_FAILED
(2548, X'9F4') Après la reconnexion, une erreur s'est produite lors de la réinstallation des descripteurs d'une connexion reconnectable.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. Utilisez cet appel uniquement lorsque le gestionnaire de files d'attente lui-même coordonne l'unité d'oeuvre. Ce nom peut être :

   - Unité d'oeuvre locale dans laquelle les modifications affectent uniquement les ressources WebSphere MQ .

- Unité de travail globale dans laquelle les modifications peuvent affecter les ressources appartenant à d'autres gestionnaires de ressources, ainsi que les ressources WebSphere MQ .

Pour plus de détails sur les unités de travail locales et globales, voir «MQBEGIN-Début de l'unité de travail», à la page 615.

2. Dans les environnements où le gestionnaire de files d'attente ne coordonne pas l'unité d'oeuvre, l'appel de validation approprié doit être utilisé à la place de MQCMIT. L'environnement peut également prendre en charge une validation implicite provoquée par l'arrêt normal de l'application.
   - Sous z/OS, utilisez les appels suivants:
     - Les programmes batch (y compris les programmes batch DL/I IMS) peuvent utiliser l'appel MQCMIT si l'unité de travail affecte uniquement les ressources WebSphere MQ . Toutefois, si l'unité de travail affecte à la fois les ressources WebSphere MQ et les ressources appartenant à d'autres gestionnaires de ressources (par exemple, DB2), utilisez l'appel SRRCMIT fourni par le service RRS (Recoverable Resource Service) z/OS . L'appel SRRCMIT valide les modifications apportées aux ressources appartenant aux gestionnaires de ressources qui ont été activés pour la coordination RRS.
     - Les applications CICS doivent utiliser la commande `EXEC CICS SYNCPOINT` pour valider explicitement l'unité de travail. Sinon, l'arrêt de la transaction entraîne une validation implicite de l'unité d'oeuvre. L'appel MQCMIT ne peut pas être utilisé pour les applications CICS .
     - Les applications IMS (autres que les programmes batch DL/I) doivent utiliser des appels IMS tels que GU et CHKP pour valider l'unité de travail. L'appel MQCMIT ne peut pas être utilisé pour les applications IMS (autres que les programmes batch DL/I).
   - Sous IBM i, utilisez cet appel pour les unités de travail locales coordonnées par le gestionnaire de files d'attente. Cela signifie qu'une définition de validation ne doit pas exister au niveau du travail, c'est-à-dire que la commande STRCMTCTL avec le paramètre `CMTSCOPE(*JOB)` ne doit pas avoir été émise pour le travail.
3. Si une application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications varie selon que l'application se termine normalement ou anormalement. Pour plus d'informations, voir MQDISC usage notes .
4. Lorsqu'une application insère ou extrait des messages dans des groupes ou des segments de messages logiques, le gestionnaire de files d'attente conserve les informations relatives au groupe de messages et au message logique pour les derniers appels MQPUT et MQGET ayant abouti. Ces informations sont associées à l'identificateur de file d'attente et incluent les éléments suivants:
   - Valeurs des zones *GroupId*, *MsgSeqNumber*, *Offset*et *MsgFlags* dans MQMD.
   - Indique si le message fait partie d'une unité de travail.
   - Pour l'appel MQPUT: indique si le message est persistant ou non persistant.

   Lorsqu'une unité d'oeuvre est validée, le gestionnaire de files d'attente conserve les informations de groupe et de segment et l'application peut continuer à insérer ou à extraire des messages dans le groupe de messages ou le message logique en cours.

   La conservation des informations de groupe et de segment lors de la validation d'une unité de travail permet à l'application de répartir un grand groupe de messages ou un grand message logique composé de plusieurs segments sur plusieurs unités de travail. L'utilisation de plusieurs unités d'oeuvre est avantageuse si le gestionnaire de files d'attente local ne dispose que d'une mémoire de file d'attente limitée. Toutefois, l'application doit conserver suffisamment d'informations pour redémarrer l'insertion ou l'obtention de messages au point approprié en cas de défaillance du système. Pour plus d'informations sur le redémarrage au point approprié après une défaillance du système, voir MQPMO_LOGICAL_ORDER et MQGMO_LOGICAL_ORDER.

   Les autres remarques d'utilisation s'appliquent uniquement lorsque le gestionnaire de files d'attente coordonne les unités de travail:
5. Une unité de travail a la même portée qu'un descripteur de connexion ; tous les appels WebSphere MQ qui affectent une unité de travail particulière doivent être exécutés à l'aide du même descripteur de connexion. Les appels émis à l'aide d'un descripteur de connexion différent (par exemple, les appels

émis par une autre application) affectent une unité d'oeuvre différente. Pour plus d'informations sur la portée des descripteurs de connexion, voir le paramètre *Hconn* décrit dans MQCONN.

6. Seuls les messages insérés ou extraits dans le cadre de l'unité d'oeuvre en cours sont affectés par cet appel.
7. Une application à exécution longue qui émet des appels MQGET, MQPUT ou MQPUT1 dans une unité de travail, mais qui n'émet jamais d'appel de validation ou d'exclusion, peut remplir les files d'attente avec des messages qui ne sont pas disponibles pour d'autres applications. Pour éviter cela, l'administrateur doit définir l'attribut de gestionnaire de files d'attente *MaxUncommittedMsgs* sur une valeur suffisamment faible pour empêcher les applications en boucle indéfinie de remplir les files d'attente, mais suffisamment élevée pour permettre aux applications de messagerie attendues de fonctionner correctement.
8. Sur les systèmes UNIX et Windows , si le paramètre *Reason* est MQRC_CONNECTION_BROKEN (avec un *CompCode* de MQCC_FAILED) ou MQRC_UNEXPECTED_ERROR, il est possible que l'unité de travail ait été correctement validée.

Appel C

```
MQCMIT (Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

Appel COBOL

```
CALL 'MQCMIT' USING HCONN, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQCMIT (Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn     fixed bin(31);  /* Connection handle */
dcl CompCode  fixed bin(31);  /* Completion code */
dcl Reason    fixed bin(31);  /* Reason code qualifying CompCode */
```

Appel d'assembleur de haut niveau

```
CALL MQCMIT,(HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Appel Visual Basic

```
MQCMIT Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQCONN-Connexion du gestionnaire de files d'attente

L'appel MQCONN connecte un programme d'application à un gestionnaire de files d'attente.

Il fournit un descripteur de connexion de gestionnaire de files d'attente, que l'application utilise lors des appels de mise en file d'attente de messages ultérieurs.

- Sous z/OS, les applications CICS n'ont pas besoin d'émettre cet appel. Ces applications sont connectées automatiquement au gestionnaire de files d'attente auquel le système CICS est connecté. Toutefois, les appels MQCONN et MQDISC sont toujours acceptés à partir des applications CICS .
- Sous IBM i, les applications s'exécutant en mode compatibilité n'ont pas besoin d'émettre cet appel. Ces applications sont connectées automatiquement au gestionnaire de files d'attente lorsqu'elles émettent le premier appel MQOPEN. Toutefois, les appels MQCONN et MQDISC sont toujours acceptés à partir des applications IBM i.

  Les autres applications (c'est-à-dire les applications qui ne s'exécutent pas en mode compatibilité) doivent utiliser l'appel MQCONN ou MQCONNX pour se connecter au gestionnaire de files d'attente et l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente. Il s'agit du style de programmation recommandé.

Une connexion client ne peut pas être établie sur une installation serveur uniquement et une connexion locale ne peut pas être établie sur une installation client uniquement.

Syntaxe

MQCONN (*QMgrName*, *Hconn*, *CompCode*, *Motif*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Il s'agit du nom du gestionnaire de files d'attente auquel l'application souhaite se connecter. Le nom peut contenir les caractères suivants:

- Caractères alphabétiques en majuscules (A à Z)
- Caractères alphabétiques minuscules (a à z)
- Chiffres (0 à 9)
- Point (.), barre oblique (/), trait de soulignement (_), pourcentage (%)

Le nom ne doit pas contenir d'espaces de début ou d'espaces imbriqués, mais peut contenir des espaces de fin. Un caractère NULL peut être utilisé pour indiquer la fin des données significatives dans le nom ; la valeur NULL et les caractères qui la suivent sont traités comme des blancs. Les restrictions suivantes s'appliquent dans les environnements indiqués:

- Sur les systèmes qui utilisent EBCDIC Katakana, les caractères minuscules ne peuvent pas être utilisés.
- Sous z/OS, les noms commençant ou se terminant par un trait de soulignement ne peuvent pas être traités par les panneaux d'opérations et de contrôle. Pour cette raison, évitez ces noms.
- Sous IBM i, placez les noms contenant des minuscules, des barres obliques ou des pourcentages entre guillemets lorsqu'ils sont spécifiés dans les commandes. N'indiquez pas ces guillemets dans le paramètre *QMgrName* .

Si le nom est entièrement vide, le nom du gestionnaire de files d'attente *par défaut* est utilisé.

Le nom spécifié pour *QMgrName* doit être le nom d'un gestionnaire de files d'attente *connectable* .

Sous z/OS, les gestionnaires de files d'attente auxquels il est possible de se connecter sont déterminés par l'environnement:

- Pour CICS, vous pouvez utiliser uniquement le gestionnaire de files d'attente auquel le système CICS est connecté. Le paramètre *QMgrName* doit toujours être spécifié, mais sa valeur est ignorée ; des blancs sont recommandés.
- Pour IMS, seuls les gestionnaires de files d'attente répertoriés dans la table de définition de sous-système (CSQQDEFV), *et* répertoriés dans la table SSM dans IMS, sont connectables (voir la note d'utilisation 6).
- Pour z/OS batch et TSO, seuls les gestionnaires de files d'attente qui résident sur le même système que l'application sont connectables (voir la remarque d'utilisation 6).

Groupes de partage de files d'attente:Sur les systèmes où plusieurs gestionnaires de files d'attente existent et sont configurés pour former un groupe de partage de files d'attente, le nom du groupe de partage de files d'attente peut être spécifié pour *QMgrName* à la place du nom d'un gestionnaire de files d'attente. Cela permet à l'application de se connecter à *tout gestionnaire de files d'attente* disponible dans le groupe de partage de files d'attente et qui se trouve sur la même image z/OS que l'application. Le système peut également être configuré de sorte que l'utilisation d'un *QMgrName* vide se connecte au groupe de partage de files d'attente au lieu du gestionnaire de files d'attente par défaut.

Si *QMgrName* indique le nom du groupe de partage de files d'attente, mais qu'il existe également un gestionnaire de files d'attente portant ce nom sur le système, la connexion est établie à ce dernier de préférence au premier. La connexion à l'un des gestionnaires de files d'attente du groupe de partage de files d'attente n'est tentée que si cette connexion échoue.

Si la connexion aboutit, vous pouvez utiliser le descripteur renvoyé par l'appel MQCONN ou MQCONNX pour accéder à *toutes* les ressources (partagées et non partagées) appartenant au gestionnaire de files d'attente auquel la connexion a été établie. L'accès à ces ressources est soumis aux contrôles d'autorisation standard.

Si l'application émet deux appels MQCONN ou MQCONNX pour établir des connexions simultanées et qu'un ou les deux appels spécifient le nom du groupe de partage de files d'attente, le deuxième appel renvoie le code achèvement MQCC_WARNING et le code anomalie MQRC_ALREADY_CONNECTED lorsqu'il se connecte au même gestionnaire de files d'attente que le premier appel.

Les groupes de partage de files d'attente sont pris en charge uniquement sous z/OS. La connexion à un groupe de partage de files d'attente est prise en charge uniquement dans les environnements par lots, RRS par lots et TSO.

WebSphere MQ Applications client MQI: pour les applications client WebSphere MQ MQI, une connexion est tentée pour chaque définition de canal de connexion client avec le nom de gestionnaire de files d'attente spécifié, jusqu'à ce qu'elle aboutisse. Toutefois, le gestionnaire de files d'attente doit avoir le même nom que le nom spécifié. Si un nom vide est spécifié, chaque canal de connexion client avec un nom de gestionnaire de files d'attente vide est tenté jusqu'à ce qu'il aboutisse ; dans ce cas, il n'y a pas de contrôle sur le nom réel du gestionnaire de files d'attente.

WebSphere MQ ne sont pas prises en charge dans z/OS, mais z/OS peut agir en tant que serveur WebSphere MQ , auquel les applications client WebSphere MQ peuvent se connecter.

WebSphere MQ Groupes de gestionnaires de files d'attente client MQI: Si le nom spécifié commence par un astérisque (*), le nom du gestionnaire de files d'attente auquel la connexion est établie peut être différent de celui spécifié par l'application. Le nom spécifié (sans astérisque) définit un *groupe* de gestionnaires de files d'attente éligibles pour la connexion. L'implémentation en sélectionne une dans le groupe en essayant chacune à son tour jusqu'à ce qu'une connexion soit trouvée. L'ordre dans lequel les connexions sont tentées est influencé par la pondération du canal client et les valeurs d'affinité de connexion des canaux candidats. Si aucun des gestionnaires de files d'attente du groupe n'est disponible pour la connexion, l'appel échoue. Chaque gestionnaire de files d'attente est essayé une seule fois. Si un astérisque seul est spécifié pour le nom, un groupe de gestionnaires de files d'attente par défaut défini par l'implémentation est utilisé.

Les groupes de gestionnaires de files d'attente sont pris en charge uniquement pour les applications s'exécutant dans un environnement MQ-client ; l'appel échoue si une application non client spécifie un nom de gestionnaire de files d'attente commençant par un astérisque. Un groupe est défini en fournissant plusieurs définitions de canal de connexion client avec le même nom de gestionnaire de files d'attente (nom spécifié sans astérisque), pour communiquer avec chacun des gestionnaires de files d'attente du groupe. Le groupe par défaut est défini en fournissant une ou plusieurs définitions de canal de connexion client, chacune avec un nom de gestionnaire de files d'attente vide (la spécification d'un nom entièrement vide a donc le même effet que la spécification d'un astérisque unique pour le nom d'une application client).

Après la connexion à un gestionnaire de files d'attente d'un groupe, une application peut spécifier des blancs de la manière habituelle dans les zones de nom de gestionnaire de files d'attente des descripteurs de message et d'objet pour désigner le nom du gestionnaire de files d'attente auquel l'application s'est connectée (le *gestionnaire de files d'attente local*). Si l'application doit connaître ce nom, utilisez l'appel MQINQ pour interroger l'attribut de gestionnaire de files d'attente *QMgrName* .

L'ajout d'un astérisque au nom de connexion implique que l'application ne dépend pas de la connexion à un gestionnaire de files d'attente particulier dans le groupe. Les applications appropriées sont les suivantes:

- Applications qui placent des messages mais n'en reçoivent pas.
- Applications qui placent des messages de demande, puis extraient les messages de réponse d'une file d'attente *dynamique temporaire* .

Les applications inadaptées sont celles qui doivent extraire des messages d'une file d'attente particulière sur un gestionnaire de files d'attente particulier ; ces applications ne doivent pas faire précéder le nom d'un astérisque.

Si vous indiquez un astérisque, la longueur maximale du reste du nom est de 47 caractères.

Les groupes de gestionnaires de files d'attente ne sont pas pris en charge sous z/OS.

La longueur de ce paramètre est donnée par MQ_Q_MGR_NAME_LENGTH.

Hconn
Type: MQHCONN-sortie

Ce descripteur représente la connexion au gestionnaire de files d'attente. Spécifiez-le sur tous les appels de mise en file d'attente de messages ultérieurs émis par l'application. Elle cesse d'être valide lorsque l'appel MQDISC est émis ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête.

WebSphere MQ fournit désormais la bibliothèque mqm avec des packages client ainsi que des packages serveur. Cela signifie que lorsqu'un appel MQI trouvé dans la bibliothèque mqm est effectué, le type de connexion est vérifié pour voir s'il s'agit d'une connexion client ou serveur, puis l'appel sous-jacent correct est effectué. Par conséquent, un exit auquel est transmis un *Hconn* peut désormais être lié à la bibliothèque mqm, mais utilisé sur une installation client.

Portée de l'indicateur:La portée de l'identificateur renvoyé dépend de l'appel utilisé pour se connecter au gestionnaire de files d'attente (MQCONN ou MQCONNX). Si l'appel utilisé est MQCONNX, la portée du descripteur dépend également de l'option MQCNO_HANDLE_SHARE_ * spécifiée dans la zone *Options* de la structure MQCNO.

- Si l'appel est MQCONN ou que l'option MQCNO_HANDLE_SHARE_NONE est spécifiée, le descripteur renvoyé est un descripteur *non partagé* .

  La portée d'un descripteur non partagé est la plus petite unité de traitement parallèle prise en charge par la plateforme sur laquelle l'application s'exécute (voir Tableau 564, à la page 649 pour plus de détails) ; le descripteur n'est pas valide en dehors de l'unité de traitement parallèle à partir de laquelle l'appel a été émis.
- Si vous spécifiez l'option MQCNO_HANDLE_SHARE_BLOCK ou MQCNO_HANDLE_SHARE_NO_BLOCK, le descripteur renvoyé est un descripteur *partagé* .

  La portée d'un descripteur partagé est le processus qui possède l'unité d'exécution à partir de laquelle l'appel a été émis ; le descripteur peut être utilisé à partir de n'importe quelle unité d'exécution appartenant à ce processus. Toutes les plateformes ne prennent pas en charge les unités d'exécution.
- Si l'appel MQCONN ou MQCONNX échoue avec un code achèvement égal à MQCC_FAILED, la valeur Hconn n'est pas définie.

Tableau 564. Portée des descripteurs non partagés sur différentes plateformes

Plateforme	**Portée du descripteur non partagé**
z/OS	• CICS: tâche CICS • IMS: tâche, jusqu'au point de synchronisation suivant (à l'exclusion des sous-tâches de la tâche) • z/OS batch et TSO: la tâche (à l'exclusion des sous-tâches de la tâche)
IBM i	Travail
Systèmes UNIX	Unité d'exécution
Applications Windows 16 bits	Processus
Applications Windows 32 bits	Unité d'exécution

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, la valeur renvoyée est:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_ALREADY_CONNECTED
(2002, X'7D2') Application déjà connectée.

MQRC_CLUSTER_EXIT_LOAD_ERROR
(2267, X'8DB') Impossible de charger l'exit de charge de travail du cluster.

MQRC_SSL_ALREADY_INITIALISÉ
(2391, X' 957') SSL déjà initialisé.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_CONN_LOAD_ERROR
(2129, X'851') Impossible de charger le module de connexion de l'adaptateur.

MQRC_ADAPTER_DEFS_ERREUR
(2131, X'853') Module de définition de sous-système d'adaptateur incorrect.

MQRC_ADAPTER_DEFS_LOAD_ERROR
(2132, X'854') Impossible de charger le module de définition de sous-système d'adaptateur.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ADAPTER_STOCKAGE_RUPTURE
(2127, X'84F') Mémoire insuffisante pour l'adaptateur.

MQRC_ANOTHER_Q_MGR_CONNECTED
(2103, X'837') Un autre gestionnaire de files d'attente est déjà connecté.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_INIT_ERROR
(2375, X' 947') L'initialisation de l'exit API a échoué.

MQRC_API_EXIT_TERM_ERREUR
(2376, X' 948') L'arrêt de l'exit API a échoué.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONN_ID_IN_USE
(2160, X'870') Identificateur de connexion déjà utilisé.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_ERREUR
(2273, X'8E1') Erreur lors du traitement de l'appel MQCONN.

MQRC_CONNECTION_NOT_AVAILABLE
(2568, X'A08') se produit sur un appel MQCONN ou MQCONNX lorsque le gestionnaire de files d'attente ne parvient pas à fournir une connexion du type de connexion demandé sur l'installation en cours. Une connexion client ne peut pas être établie sur une installation serveur uniquement. Une connexion locale ne peut pas être établie sur une installation client uniquement.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Connexion en cours de mise au repos.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_CRYPTO_ERREUR_MATÉRIEL
(2382, X'94E') Erreur de configuration matérielle cryptographique.

MQRC_DUPLICATE_RECOV_COORD
(2163, X'873') Le coordinateur de reprise existe.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Appel non valide dans l'environnement.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HOST_NOT_AVAILABLE
(2538, X'9EA') Un appel MQCONN a été émis par un client pour se connecter à un gestionnaire de files d'attente, mais la tentative d'allocation d'une conversation au système distant a échoué.

Non-concordance de MQRC_INSTALLATION_MISMATCH
(2583, X'A17') Non concordance entre l'installation du gestionnaire de files d'attente et la bibliothèque sélectionnée.

MQRC_KEY_REPOSITORY_ERROR
(2381, X'94D') Référentiel de clés non valide.

MQRC_MAX_CONNS_LIMIT_ATTEINTE
(2025, X'7E9') Nombre maximal de connexions atteint.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_OPEN_FAILED
(2137, X'859') L'ouverture de l'objet a échoué.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_QUIESCING
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SECURITY_ERROR (ERREUR DE SECURITE MQ)
(2063, X'80F') Une erreur de sécurité s'est produite.

MQRC_SSL_INITIALIZATION_ERROR
(2393, X' 959') Erreur d'initialisation SSL.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. Le gestionnaire de files d'attente auquel la connexion est établie à l'aide de l'appel MQCONN est appelé *gestionnaire de files d'attente local*.
2. Les files d'attente appartenant au gestionnaire de files d'attente local apparaissent à l'application en tant que files d'attente locales. Il est possible d'insérer et d'extraire des messages de ces files d'attente.

   Les files d'attente partagées appartenant au groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local apparaissent dans l'application en tant que files d'attente locales. Il est possible d'insérer et d'extraire des messages de ces files d'attente.

Les files d'attente appartenant à des gestionnaires de files d'attente éloignées apparaissent comme des files d'attente éloignées. Il est possible d'insérer des messages dans ces files d'attente, mais pas d'en extraire.

3. Si le gestionnaire de files d'attente échoue alors qu'une application est en cours d'exécution, l'application doit émettre à nouveau l'appel MQCONN pour obtenir un nouveau descripteur de connexion à utiliser lors des appels WebSphere MQ ultérieurs. L'application peut émettre l'appel MQCONN périodiquement jusqu'à ce que l'appel aboutisse.

   Si une application ne sait pas si elle est connectée au gestionnaire de files d'attente, elle peut émettre un appel MQCONN en toute sécurité pour obtenir un descripteur de connexion. Si l'application est déjà connectée, le descripteur renvoyé est identique à celui renvoyé par l'appel MQCONN précédent, mais avec le code achèvement MQCC_WARNING et le code anomalie MQRC_ALREADY_CONNECTED.
4. Lorsque l'application a terminé d'utiliser les appels WebSphere MQ , elle doit utiliser l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente.
5. Si l'appel MQCONN échoue avec un code achèvement égal à MQCC_FAILED, la valeur Hconn n'est pas définie.
6. Sous z/OS :

   - Les applications par lots, TSO et IMS doivent émettre l'appel MQCONN pour utiliser les autres appels WebSphere MQ . Ces applications peuvent se connecter simultanément à plusieurs gestionnaires de files d'attente.

     Si le gestionnaire de files d'attente échoue, l'application doit émettre à nouveau l'appel après le redémarrage du gestionnaire de files d'attente pour obtenir un nouveau descripteur de connexion.

     Bien que les applications IMS puissent émettre l'appel MQCONN à plusieurs reprises, même lorsqu'elles sont déjà connectées, cela n'est pas recommandé pour les programmes de traitement de messages en ligne (MPP).
   - Les applications CICS n'ont pas besoin d'émettre l'appel MQCONN pour utiliser les autres appels WebSphere MQ , mais elles peuvent le faire si elles le souhaitent ; l'appel MQCONN et l'appel MQDISC sont acceptés. Toutefois, il n'est pas possible de se connecter simultanément à plusieurs gestionnaires de files d'attente.

     En cas de défaillance du gestionnaire de files d'attente, ces applications sont automatiquement reconnectées au redémarrage du gestionnaire de files d'attente et n'ont donc pas besoin d'émettre l'appel MQCONN.
7. Sous z/OS, pour définir les gestionnaires de files d'attente disponibles:

   - Pour les applications par lots, les programmeurs système peuvent utiliser la macro CSQBDEF pour créer un module (CSQBDEFV) qui définit le nom du gestionnaire de files d'attente par défaut ou le nom du groupe de partage de files d'attente.
   - Pour les applications IMS , les programmeurs système peuvent utiliser la macro CSQQDEFX pour créer un module (CSQQDEFV) qui définit les noms des gestionnaires de files d'attente disponibles et spécifie le gestionnaire de files d'attente par défaut.

     En outre, chaque gestionnaire de files d'attente doit être défini dans la région de contrôle IMS et dans chaque région dépendante accédant à ce gestionnaire de files d'attente. Pour ce faire, vous devez créer un membre de sous-système dans IMS.Bibliothèque PROCLIB et identifiez le membre de sous-système dans les régions IMS applicables. Si une application tente de se connecter à un gestionnaire de files d'attente qui n'est pas défini dans le membre de sous-système pour sa région IMS , l'application se termine de manière anormale.
8. Sous IBM i, les applications écrites pour les éditions précédentes du gestionnaire de files d'attente peuvent s'exécuter sans recompilation. Cet élément est appelé *mode compatibilité*. Ce mode de fonctionnement fournit un environnement d'exécution compatible pour les applications. Il comprend les éléments suivants:

   - Le programme de service AMQZSTUB se trouvant dans la bibliothèque QMQM.

AMQZSTUB fournit la même interface publique que les éditions précédentes et possède la même signature. Utilisez ce programme de service pour accéder à l'interface MQI via des appels de procédure liés.

- Programme QMQM résidant dans la bibliothèque QMQM.

  QMQM permet d'accéder à l'interface MQI via des appels de programme dynamiques.

- Programmes MQCLOSE, MQCONN, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQPUT1et MQSET résidant dans la bibliothèque QMQM.

  Ces programmes permettent également d'accéder à l'interface MQI via des appels de programme dynamique, mais avec une liste de paramètres correspondant aux descriptions standard des appels WebSphere MQ .

Ces trois interfaces n'incluent pas les fonctions introduites dans WebSphere MQ version 5.1. Par exemple, les appels MQBACK, MQCMIT et MQCONNX ne sont pas pris en charge. La prise en charge fournie par ces interfaces concerne uniquement les applications à unité d'exécution unique.

La prise en charge des nouveaux appels WebSphere MQ dans les applications à unité d'exécution unique et de tous les appels WebSphere MQ dans les applications à unités d'exécution multiples est assurée par les programmes de service LIBMQM et LIBMQM_R.

9. Sous IBM i, les programmes qui se terminent de manière anormale ne sont pas automatiquement déconnectés du gestionnaire de files d'attente. Ecrivez des applications pour permettre à l'appel MQCONN ou MQCONNX de renvoyer le code achèvement MQCC_WARNING et le code anomalie MQRC_ALREADY_CONNECTED. Utilisez normalement le descripteur de connexion renvoyé dans cette situation.

Appel C

```
MQCONN (QMgrName, &Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQCHAR48  QMgrName;  /* Name of queue manager */
MQHCONN   Hconn;     /* Connection handle */
MQLONG    CompCode;  /* Completion code */
MQLONG    Reason;    /* Reason code qualifying CompCode */
```

Appel COBOL

```
CALL 'MQCONN' USING QMGRNAME, HCONN, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQCONN (QMgrName, Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQCONN,(QMGRNAME,HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
QMGRNAME  DS  CL48  Name of queue manager
HCONN     DS  F     Connection handle
COMPCODE  DS  F     Completion code
REASON    DS  F     Reason code qualifying COMPCODE
```

Appel Visual Basic

```
MQCONN QMgrName, Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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 - Connexion du gestionnaire de files d'attente (étendue)

L'appel MQCONNX connecte un programme d'application à un gestionnaire de files d'attente. Il fournit un descripteur de connexion de gestionnaire de files d'attente, qui est utilisé par l'application lors des appels WebSphere MQ suivants.

L'appel MQCONNX est similaire à l'appel MQCONN, sauf que MQCONNX permet de spécifier des options pour contrôler la façon dont l'appel fonctionne.

- Cet appel est pris en charge sur tous les systèmes WebSphere MQ et les clients WebSphere MQ connectés à ces systèmes.
- Sous IBM i, cet appel n'est pas pris en charge pour les applications s'exécutant en mode compatibilité.

Une connexion client ne peut pas être établie sur une installation serveur uniquement et une connexion locale ne peut pas être établie sur une installation client uniquement.

Syntaxe

MQCONNX (*QMgrName*, *ConnectOpts*, *Hconn*, *CompCode*, *Motif*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Pour plus de détails, voir le paramètre *QMgrName* décrit dans «MQCONN-Connexion du gestionnaire de files d'attente», à la page 646 .

ConnectOpts
Type: MQCNO-entrée/sortie

Voir «MQCNO-Options de connexion», à la page 298 pour plus de détails.

Hconn
Type: MQHCONN-sortie

Ce descripteur représente la connexion au gestionnaire de files d'attente. Spécifiez-le sur tous les appels de mise en file d'attente de messages ultérieurs émis par l'application. Elle cesse d'être valide lorsque l'appel MQDISC est émis ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête.

WebSphere MQ fournit désormais la bibliothèque mqm avec des packages client ainsi que des packages serveur. Cela signifie que lorsqu'un appel MQI trouvé dans la bibliothèque mqm est effectué, le type de connexion est vérifié pour voir s'il s'agit d'une connexion client ou serveur, puis l'appel sous-jacent correct est effectué. Par conséquent, un exit auquel est transmis un *Hconn* peut désormais être lié à la bibliothèque mqm, mais utilisé sur une installation client.

Portée de l'indicateur:La portée de l'identificateur renvoyé dépend de l'appel utilisé pour se connecter au gestionnaire de files d'attente (MQCONN ou MQCONNX). Si l'appel utilisé est MQCONNX, la portée du descripteur dépend également de l'option MQCNO_HANDLE_SHARE_ * spécifiée dans la zone *Options* de la structure MQCNO.

- Si l'appel est MQCONN ou que l'option MQCNO_HANDLE_SHARE_NONE est spécifiée, le descripteur renvoyé est un descripteur *non partagé* .

  La portée d'un descripteur non partagé est la plus petite unité de traitement parallèle prise en charge par la plateforme sur laquelle l'application s'exécute (voir Tableau 565, à la page 655 pour plus de détails) ; le descripteur n'est pas valide en dehors de l'unité de traitement parallèle à partir de laquelle l'appel a été émis.
- Si vous spécifiez l'option MQCNO_HANDLE_SHARE_BLOCK ou MQCNO_HANDLE_SHARE_NO_BLOCK, le descripteur renvoyé est un descripteur *partagé* .

  La portée d'un descripteur partagé est le processus qui possède l'unité d'exécution à partir de laquelle l'appel a été émis ; le descripteur peut être utilisé à partir de n'importe quelle unité d'exécution appartenant à ce processus. Toutes les plateformes ne prennent pas en charge les unités d'exécution.
- Si l'appel MQCONN ou MQCONNX échoue avec un code achèvement égal à MQCC_FAILED, la valeur Hconn n'est pas définie.

Tableau 565. Portée des descripteurs non partagés sur différentes plateformes

Plateforme	**Portée du descripteur non partagé**
z/OS	• CICS: tâche CICS • IMS: tâche, jusqu'au point de synchronisation suivant (à l'exclusion des sous-tâches de la tâche) • z/OS batch et TSO: la tâche (à l'exclusion des sous-tâches de la tâche)
IBM i	Travail
Systèmes UNIX	Unité d'exécution
Applications Windows 16 bits	Processus
Applications Windows 32 bits	Unité d'exécution

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, la valeur renvoyée est:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

CompCode
Type : MQLONG - sortie

Pour plus de détails, voir le paramètre *CompCode* décrit dans «MQCONN-Connexion du gestionnaire de files d'attente», à la page 646 .

raison
Type : MQLONG - sortie

Les codes suivants peuvent être renvoyés par les appels MQCONN et MQCONNX. Pour obtenir la liste des codes supplémentaires pouvant être renvoyés par l'appel MQCONNX, voir les codes suivants.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_ALREADY_CONNECTED
(2002, X'7D2') Application déjà connectée.

MQRC_CLUSTER_EXIT_LOAD_ERROR
(2267, X'8DB') Impossible de charger l'exit de charge de travail du cluster.

MQRC_SSL_ALREADY_INITIALISÉ
(2391, X' 957') SSL déjà initialisé.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_CONN_LOAD_ERROR
(2129, X'851') Impossible de charger le module de connexion de l'adaptateur.

MQRC_ADAPTER_DEFS_ERREUR
(2131, X'853') Module de définition de sous-système d'adaptateur incorrect.

MQRC_ADAPTER_DEFS_LOAD_ERROR
(2132, X'854') Impossible de charger le module de définition de sous-système d'adaptateur.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ADAPTER_STOCKAGE_RUPTURE
(2127, X'84F') Mémoire insuffisante pour l'adaptateur.

MQRC_ANOTHER_Q_MGR_CONNECTED
(2103, X'837') Un autre gestionnaire de files d'attente est déjà connecté.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_INIT_ERROR
(2375, X' 947') L'initialisation de l'exit API a échoué.

MQRC_API_EXIT_TERM_ERREUR
(2376, X' 948') L'arrêt de l'exit API a échoué.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONN_ID_IN_USE
(2160, X'870') Identificateur de connexion déjà utilisé.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_ERREUR
(2273, X'8E1') Erreur lors du traitement de l'appel MQCONN.

MQRC_CONNECTION_NOT_AVAILABLE
(2568, X'A08') se produit sur un appel MQCONN ou MQCONNX lorsque le gestionnaire de files d'attente ne parvient pas à fournir une connexion du type de connexion demandé sur l'installation en cours. Une connexion client ne peut pas être établie sur une installation serveur uniquement. Une connexion locale ne peut pas être établie sur une installation client uniquement.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Connexion en cours de mise au repos.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_CRYPTO_ERREUR_MATÉRIEL
(2382, X'94E') Erreur de configuration matérielle cryptographique.

MQRC_DUPLICATE_RECOV_COORD
(2163, X'873') Le coordinateur de reprise existe.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Appel non valide dans l'environnement.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HOST_NOT_AVAILABLE
(2538, X'9EA') Un appel MQCONN a été émis par un client pour se connecter à un gestionnaire de files d'attente, mais la tentative d'allocation d'une conversation au système distant a échoué.

Non-concordance de MQRC_INSTALLATION_MISMATCH
(2583, X'A17') Non concordance entre l'installation du gestionnaire de files d'attente et la bibliothèque sélectionnée.

MQRC_KEY_REPOSITORY_ERROR
(2381, X'94D') Référentiel de clés non valide.

MQRC_MAX_CONNS_LIMIT_ATTEINTE
(2025, X'7E9') Nombre maximal de connexions atteint.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_OPEN_FAILED
(2137, X'859') L'ouverture de l'objet a échoué.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_QUIESCING
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SECURITY_ERROR (ERREUR DE SECURITE MQ)
(2063, X'80F') Une erreur de sécurité s'est produite.

MQRC_SSL_INITIALIZATION_ERROR
(2393, X' 959') Erreur d'initialisation SSL.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Les codes anomalie supplémentaires suivants peuvent être renvoyés par l'appel MQCONNX:

Si *CompCode* a pour valeur MQCC_FAILED :

ERREUR MQRC_AIR_ERROR
(2385, X' 951') Enregistrement des informations d'authentification non valide.

MQRC_AUTH_INFO_CONN_NAME_ERROR
(2387, X' 953') Nom de connexion des informations d'authentification incorrect.

MQRC_AUTH_INFO_REC_COUNT_ERROR
(2383, X'94F') Le nombre d'enregistrements d'informations d'authentification n'est pas valide.

MQRC_AUTH_INFO_REC_ERROR
(2384, X' 950') Zones d'enregistrement d'informations d'authentification non valides.

MQRC_AUTH_INFO_TYPE_ERROR
(2386, X' 952') Type d'informations d'authentification non valide.

MQRC_CD_ERREUR
(2277, X'8E5') Définition de canal non valide.

MQRC_CLIENT_CONN_ERROR
(2278, X'8E6') Zones de connexion client non valides.

MQRC_CNO_ERREUR
(2139, X'85B') Structure d'options de connexion non valide.

MQRC_CONN_TAG_IN_USE
(2271, X'8DF') Balise de connexion utilisée.

MQRC_CONN_TAG_NON_UTILISABLE
(2350, X'92E') La balise de connexion n'est pas utilisable.

MQRC_LDAP_PASSWORD_ERROR
(2390, X' 956') Mot de passe LDAP non valide.

MQRC_LDAP_USER_NAME_ERROR
(2388, X' 954') Zones de nom d'utilisateur LDAP non valides.

MQRC_LDAP_NOM_UTILISATEUR_LENGTH_ERR
(2389, X' 955') Longueur du nom d'utilisateur LDAP incorrecte.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_ERREUR DE SCO_DU
(2380, X'94C') La structure des options de configuration SSL n'est pas valide.

MQRC_SSL_CONFIG_ERROR
(2392, X' 958') Erreur de configuration SSL.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

Pour le langage de programmation Visual Basic, le point suivant s'applique:

- Le paramètre *ConnectOpts* est déclaré comme étant de type MQCNO. Si l'application s'exécute en tant que client WebSphere MQ MQI et que vous souhaitez spécifier les paramètres du canal de connexion client, déclarez le paramètre *ConnectOpts* comme étant de type Anyafin que l'application puisse spécifier une structure MQCNOCD sur l'appel à la place d'une structure MQCNO. Toutefois, cela signifie que le paramètre *ConnectOpts* ne peut pas être vérifié pour s'assurer qu'il s'agit du type de données correct.

Appel C

```
MQCONNX (QMgrName, &ConnectOpts, &Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQCONNX' USING QMGRNAME, CONNECTOPTS, HCONN, COMPCODE,
REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQCONNX (QMgrName, ConnectOpts, Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQCONNX,(QMGRNAME,CONNECTOPTS,HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Appel Visual Basic

```
MQCONNX QMgrName, ConnectOpts, Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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-Création d'un descripteur de message

L'appel MQCRTMH renvoie un descripteur de message.

Une application peut utiliser l'appel MQCRTMH sur les appels de mise en file d'attente de messages suivants:

- Utilisez l'appel MQSETMP pour définir une propriété du descripteur de message.
- Utilisez l'appel MQINQMP pour demander la valeur d'une propriété du descripteur de message.
- Utilisez l'appel MQDLTMP pour supprimer une propriété du descripteur de message.

Le descripteur de message peut être utilisé sur les appels MQPUT et MQPUT1 pour associer les propriétés du descripteur de message à celles du message en cours d'insertion. De même, en spécifiant un descripteur de message dans l'appel MQGET, les propriétés du message en cours d'extraction sont accessibles à l'aide du descripteur de message une fois l'appel MQGET terminé.

Utilisez MQDLTMH pour supprimer le descripteur de message.

Syntaxe

MQCRTMH (*Hconn*, *CrtMsgHOpts*, *Hmsg*, *CompCode*, *Motif*)

Paramètres

Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent. Si la connexion au gestionnaire de files d'attente cesse d'être valide et qu'aucun appel WebSphere MQ n'est en cours d'exécution sur l'identificateur de message, MQDLTMH est implicitement appelé pour supprimer le message.

Vous pouvez également spécifier la valeur suivante:

MQHC_UNASSOCIATED_HCONN

Le descripteur de connexion ne représente pas une connexion à un gestionnaire de files d'attente particulier.

Lorsque cette valeur est utilisée, le descripteur de message doit être supprimé avec un appel explicite à MQDLTMH afin de libérer le stockage qui lui est alloué ; WebSphere MQ ne supprime jamais implicitement le descripteur de message.

Au moins une connexion valide à un gestionnaire de files d'attente doit être établie sur l'unité d'exécution qui crée le descripteur de message, sinon l'appel échoue avec MQRC_HCONN_ERROR.

Dans un environnement comportant plusieurs installations sur un seul système, la valeur MQHC_UNASSOCIATED_HCONN est limitée à une utilisation avec la première installation chargée

dans le processus. Le code anomalie MQRC_HMSG_NOT_AVAILABLE est renvoyé si le descripteur de message est fourni à une installation différente.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et vous pouvez spécifier la valeur suivante pour *Hconn*:

CONN MQHC_DEFAUT
Descripteur de connexion par défaut

CrtMsgHOpts
Type: MQCMHO-entrée

Options qui contrôlent l'action de MQCRTMH. Pour plus d'informations, voir MQCMHO .

Msg
Type: MQHMSG-sortie

En sortie, un descripteur de message est renvoyé et peut être utilisé pour définir, interroger et supprimer les propriétés du descripteur de message. Initialement, le descripteur de message ne contient aucune propriété.

Un descripteur de message est également associé à un descripteur de message. Initialement, il contient les valeurs par défaut. Les valeurs des zones de descripteur de message associées peuvent être définies et renseignées à l'aide des appels MQSETMP et MQINQMP. L'appel MQDLTMP réinitialise une zone du descripteur de message à sa valeur par défaut.

Si le paramètre *Hconn* est spécifié comme valeur MQHC_UNASSOCIATED_HCONN, le descripteur de message renvoyé peut être utilisé sur les appels MQGET, MQPUT ou MQPUT1 avec n'importe quelle connexion au sein de l'unité de traitement, mais il ne peut être utilisé que par un seul appel WebSphere MQ à la fois. Si le descripteur est utilisé lorsqu'un second appel WebSphere MQ tente d'utiliser le même descripteur de message, le second appel WebSphere MQ échoue avec le code anomalie MQRC_MSG_HANDLE_IN_USE.

Si le paramètre *Hconn* n'est pas MQHC_UNASSOCIATED_HCONN, le descripteur de message renvoyé ne peut être utilisé que sur la connexion spécifiée.

La même valeur de paramètre *Hconn* doit être utilisée sur les appels MQI suivants où ce descripteur de message est utilisé:

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMHBUF
- MQBUFMH

L'identificateur de message renvoyé cesse d'être valide lorsque l'appel MQDLTMH est émis pour l'identificateur de message ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête. MQDLTMH est appelé implicitement si une connexion spécifique est fournie lors de la création du descripteur de message et que la connexion au gestionnaire de files d'attente cesse d'être valide, par exemple, si MQDBC est appelé.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Carte non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

ERREUR MQRC_CMHO_ERROR
(2461, X'099D') La structure des options de création de descripteur de message n'est pas valide.

MQRC_CONNECTION_BROKEN
(2273, X'7D9') La connexion au gestionnaire de files d'attente a été perdue.

MQRC_HANDLE_NOT_AVAILABLE
(2017, X'07E1') Plus de descripteurs disponibles.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

ERREUR MQRC_HMSG_ERROR
(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Options non valides ou non cohérentes.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

C

```
MQCRTMH (Hconn, &CrtMsgHOpts, &Hmsg, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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.
```

Déclarez les paramètres comme suit :

```
**   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);
```

Déclarez les paramètres comme suit :

```
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)
```

Déclarez les paramètres comme suit :

```
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-Rappels de contrôle

L'appel MQCTL effectue des actions de contrôle sur les rappels et les descripteurs d'objet ouverts pour une connexion.

Syntaxe

MQCTL (*Hconn*, *Opération*, *ControlOpts*, *CompCode*, *Motif*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et vous pouvez spécifier la valeur spéciale suivante pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

operation
Type : MQLONG - entrée

Opération en cours de traitement sur le rappel défini pour le descripteur d'objet spécifié. Vous devez spécifier une seule des options suivantes:

DEMARRAGE MQOP

Démarrer la consommation des messages pour toutes les fonctions de consommateur de message définies pour le descripteur de connexion spécifié.

Les rappels s'exécutent sur une unité d'exécution démarrée par le système, qui est différente de toutes les unités d'exécution d'application.

Cette opération permet de contrôler la poignée de connexion fournie au système. Les seuls appels MQI pouvant être émis par une unité d'exécution autre que l'unité d'exécution destinataire sont les suivants:

- MQCTL avec l'opération MQOP_STOP
- MQCTL avec l'opération MQOP_SUSPEND
- MQDISC-Exécute MQCTL avec l'opération MQOP_STOP avant de déconnecter HConn.

MQRC_HCONN_ASYNC_ACTIVE est renvoyé si un appel API WebSphere MQ est émis alors que le descripteur de connexion est démarré et que l'appel ne provient pas d'une fonction de consommateur de message.

Si un consommateur de message arrête la connexion lors de l'appel MQCBCT_START_CALL, l'appel MQCTL est renvoyé avec le code anomalie d'échec MQRC_CONNECTION_STOPPED.

Cela peut être émis dans une fonction de consommateur. Pour la même connexion que la routine de rappel, son seul objectif est d'annuler une opération MQOP_STOP précédemment émise.

Cette option n'est pas prise en charge dans les environnements suivants: CICS sous z/OS ou si l'application est liée à une bibliothèque WebSphere MQ sans unités d'exécution.

MQOP_START_WAIT

Démarrer la consommation des messages pour toutes les fonctions de consommateur de message définies pour le descripteur de connexion spécifié.

Les consommateurs de message s'exécutent sur la même unité d'exécution et le contrôle n'est pas renvoyé à l'appelant de MQCTL tant que:

- Publié par l'utilisation des opérations MQCTL MQOP_STOP ou MQOP_SUSPEND, ou
- Toutes les routines de consommateur ont été désenregistrées ou suspendues.

Si tous les consommateurs sont désenregistrés ou suspendus, une opération MQOP_STOP implicite est émise.

Cette option ne peut pas être utilisée à partir d'une routine de rappel, que ce soit pour le descripteur de connexion en cours ou tout autre descripteur de connexion. Si l'appel est tenté, il est renvoyé avec MQRC_ENVIRONMENT_ERROR.

Si, à un moment quelconque au cours d'une opération MQOP_START_WAIT, il n'y a pas de consommateurs enregistrés et non suspendus, l'appel échoue avec le code anomalie MQRC_NO_CALLBACKS_ACTIVE.

Si, au cours d'une opération MQOP_START_WAIT, la connexion est suspendue, l'appel MQCTL renvoie un code anomalie d'avertissement MQRC_CONNECTION_SUSPENDED ; la connexion reste'démarrée'.

L'application peut choisir d'émettre MQOP_STOP ou MQOP_RESUME. Dans cette instance, l'opération MQOP_RESUME se bloque.

Cette option n'est pas prise en charge dans un client à unité d'exécution unique.

MQOP_ARRET

Arrêtez la consommation des messages et attendez que tous les destinataires aient terminé leurs opérations avant de terminer cette option. Cette opération libère le descripteur de connexion.

Si elle est émise à partir d'une routine de rappel, cette option n'est pas appliquée tant que la routine n'est pas arrêtée. Aucune autre routine de consommateur de message n'est appelée une fois que les routines de consommateur des messages déjà lus sont terminées et après que des appels d'arrêt (si requis) ont été effectués aux routines de rappel.

S'il est émis en dehors d'une routine de rappel, le contrôle ne revient pas à l'appelant tant que les routines de consommateur pour les messages déjà lus ne sont pas terminées et que des appels d'arrêt (si requis) aux rappels n'ont pas été effectués. Les rappels eux-mêmes, cependant, restent enregistrés.

Cette fonction n'a aucun effet sur les messages de lecture anticipée. Vous devez vous assurer que les consommateurs exécutent MQCLOSE (MQCO_QUIESCE), à partir de la fonction de rappel, pour déterminer s'il existe d'autres messages disponibles à distribuer.

MQOP_SUSPENSION
Mettez en pause la consommation des messages. Cette opération libère le descripteur de connexion.

Cela n'a aucun effet sur la lecture avant les messages de l'application. Si vous avez l'intention d'arrêter de consommer des messages pendant longtemps, envisagez de fermer la file d'attente et de la rouvrir lorsque la consommation se poursuit.

S'il est émis à partir d'une routine de rappel, il ne prend effet qu'à la fin de la routine. Aucune autre routine de consommateur de message ne sera appelée après la sortie de la routine en cours.

S'il est émis en dehors d'un rappel, le contrôle ne revient pas à l'appelant tant que la routine de consommateur en cours n'est pas terminée et qu'aucun autre n'est appelé.

MQOP_RESUME
Reprenez la consommation des messages.

Cette option est normalement émise à partir de l'unité d'exécution de l'application principale, mais elle peut également être utilisée à partir d'une routine de rappel pour annuler une demande de suspension antérieure émise dans la même routine.

Si MQOP_RESUME est utilisé pour reprendre un MQOP_START_WAIT, l'opération se bloque.

ControlOpts
Type: MQCTLO-entrée

Options qui contrôlent l'action de MQCTL

Pour plus de détails sur la structure, voir «MQCTLO-Structure des options de rappel de contrôle», à la page 318 .

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855') Impossible de charger les modules de service de conversion de données.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

MQRC_CALLBACK_LINK_ERROR
(2487, X'9B7') Impossible d'appeler la routine de rappel

MQRC_CALLBACK_NON_ENREGISTREE
(2448, X' 990') Impossible de désenregistrer, de suspendre ou de reprendre car il n'y a pas de rappel enregistré

MQRC_CALLBACK_ROUTINE_ERROR
(2486, X'9B6') Soit CallbackFunction et CallbackName ont été spécifiés sur un appel MQOP_REGISTER.

Ou bien CallbackFunction ou CallbackName ont été spécifiés mais ne correspondent pas à la fonction de rappel actuellement enregistrée.

MQRC_CALLBACK_TYPE_ERREUR
(2483, X'9B3') Zone de type CallBackincorrecte.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CBD_ERREUR
(2444, X'98C') Le bloc d'options est incorrect.

MQRC_CBD_OPTIONS_ERREUR
(2484, X'9B4') Zone d'options MQCBD incorrecte.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Connexion en cours de mise au repos.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_CORREL_ID_ERROR
(2207, X'89F') Erreur d'identificateur de corrélation.

MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

MQRC_GET_INHIBITED
(2016, X'7E0') Se bloque pour la file d'attente.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Conflit entre les unités d'oeuvre globales.

MQRC_GMO_ERROR
(2186, X'88A') Structure des options de message Get non valide.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HOBJ_ERROR
(2019, X'7E3') Descripteur d'objet non valide.

MQRC_INCONSISTENT_BROWSE
(2259, X'8D3') Spécification de navigation incohérente.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

MQRC_INVALID_MSG_UNDER_CURSOR
(2246, X'8C6') Message sous le curseur non valide pour la récupération.

MQRC_LOCAL_UOW_CONFLICT
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Options de correspondance non valides.

MQRC_MAX_MSG_LENGTH_ERROR
(2485, X'9B5') Zone de longueur MaxMsgincorrecte

MQRC_MD_ERROR
(2026, X'7EA') Descripteur de message non valide.

MQRC_MODULE_ENTRY_NOT_FOUND
(2497, X'9C1') Le point d'entrée de fonction spécifié est introuvable dans le module.

MQRC_MODULE_INVALID
(2496, X'9C0') Le module a été trouvé mais son type est incorrect (32 bit/64 bits) ou il ne s'agit pas d'une dll valide.

MQRC_MODULE_NOT_FOUND
(2495, X'9BF') Module introuvable dans le chemin de recherche ou non autorisé à être chargé.

MQRC_MSG_ID_ERROR
(2206, X'89E') Erreur d'identificateur de message.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Numéro de séquence de message non valide.

MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Utilisation du jeton de message non valide.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') File d'attente non ouverte pour exploration.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') File d'attente non ouverte pour saisie.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

ERREUR D'OPERATION_MQRC
(2488, X'9B8') Code opération incorrect sur l'appel API

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_Q_DELETED
(2052, X'804') File d'attente supprimée.

MQRC_Q_INDEX_TYPE_ERROR
(2394, X'95A') Type d'index incorrect pour la file d'attente.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_QUIESCING
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SIGNAL_OUTSTANDING
(2069, X'815') Signal en attente pour ce descripteur.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818') Prise en charge du point de synchronisation non disponible.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Intervalle d'attente non valide dans MQGMO.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Version MQGMO fournie non valide.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Version MQMD fournie non valide.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. Les routines de rappel doivent vérifier les réponses de tous les services qu'elles appellent, et si la routine détecte une condition qui ne peut pas être résolue, elle doit émettre une commande MQCB MQOP_REGISISTER pour empêcher les appels répétés à la routine de rappel.
2. Sous z/OS, lorsque l'opération est MQOP_START:
   - Les programmes qui utilisent des routines de rappel asynchrone doivent être autorisés à utiliser z/OS UNIX System Services (USS).
   - Les programmes Language Environment (LE) qui utilisent des routines de rappel asynchrones doivent utiliser l'option d'exécution LE POSIX(ON).
   - Les programmes non LE qui utilisent des routines de rappel asynchrones ne doivent pas utiliser l'interface USS pthread_create (service de rappel BPX1PTC).

3. MQCTL n'est pas pris en charge dans l'adaptateur IMS .

Remarque : Dans CICS, MQOP_START n'est pas pris en charge. A la place, utilisez l'appel de fonction MQOP_START_WAIT.

Appel C

```
MQCTL (Hconn, Operation, &ControlOpts, &CompCode, &Reason)
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQCTL' USING HCONN, OPERATION, CTLOPTS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQCTL(Hconn, Operation, CtlOpts, CompCode, Reason)
```

Déclarez les paramètres comme suit :

```
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-Déconnexion du gestionnaire de files d'attente

L'appel MQDISC interrompt la connexion entre le gestionnaire de files d'attente et le programme d'application et est l'inverse de l'appel MQCONN ou MQCONNX.

- Sous z/OS, toutes les applications qui utilisent la consommation de messages asynchrones, la gestion d'événements ou le rappel, l'unité d'exécution de contrôle principale doit émettre un appel MQDISC avant de se terminer. Voir Consommation asynchrone des messages WebSphere MQ pour plus de détails.
- Sous z/OS, les applications CICS n'ont pas besoin d'émettre cet appel pour se déconnecter du gestionnaire de files d'attente, mais peuvent avoir besoin de l'émettre pour mettre fin à l'utilisation d'une balise de connexion.

- Sous IBM i, les applications qui s'exécutent en mode compatibilité n'ont pas besoin d'émettre cet appel. Pour plus d'informations, voir «MQCONN-Connexion du gestionnaire de files d'attente», à la page 646.

Syntaxe

MQDISC (*Hconn*, *CompCode*, *Raison*)

Paramètres

Hconn
Type: MQHCONN-entrée / sortie

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, vous pouvez omettre l'appel MQCONN et spécifier la valeur suivante pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

Une fois l'appel terminé, le gestionnaire de files d'attente définit *Hconn* sur une valeur qui n'est pas un descripteur valide pour l'environnement. Cette valeur est:

MQHC_UNUSABLE_HCONN
Descripteur de connexion inutilisable.

Sous z/OS, *Hconn* est défini sur une valeur non définie.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit de l'un des codes suivants:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_BACKED_OUT
(2003, X'7D3') Unité d'oeuvre annulée.

MQRC_CONN_TAG_NOT_RELÂCHÉE
(2344, X' 928') La balise de connexion n'a pas été libérée.

MQRC_OUTCOME_EN attente
(2124, X'84C') Le résultat de l'opération de validation est en attente.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_DISC_LOAD_ERROR
(2138, X'85A') Impossible de charger le module de déconnexion de l'adaptateur.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_INIT_ERROR
(2375, X' 947') L'initialisation de l'exit API a échoué.

MQRC_API_EXIT_TERM_ERREUR
(2376, X' 948') L'arrêt de l'exit API a échoué.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_OUTCOME_MIXTE
(2123, X'84B') Le résultat de l'opération de validation ou d'exclusion est mélangé.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. Si un appel MQDISC est émis alors que la connexion comporte toujours des objets ouverts sous cette connexion, le gestionnaire de files d'attente ferme ces objets avec les options de fermeture définies sur MQCO_NONE.
2. Si l'application se termine avec des modifications non validées dans une unité d'oeuvre, la disposition de ces modifications dépend de la façon dont l'application se termine:

   a. Si l'application émet l'appel MQDISC avant l'arrêt:

   - Pour une unité d'oeuvre coordonnée par le gestionnaire de files d'attente, le gestionnaire de files d'attente émet l'appel MQCMIT pour le compte de l'application. L'unité de travail est validée si possible et annulée si ce n'est pas le cas.

   - Pour une unité de travail coordonnée en externe, le statut de l'unité de travail ne change pas ; toutefois, le gestionnaire de files d'attente indique généralement que l'unité de travail doit être validée lorsqu'elle est demandée par le coordinateur de l'unité de travail.

Sous z/OS, CICS, IMS (autres que les programmes DL/1 par lots) et les applications RRS sont similaires.

b. Si l'application se termine normalement mais sans émettre l'appel MQDISC, l'action effectuée dépend de l'environnement:

- Sous z/OS, à l'exception des applications MQ Java ou MQ JMS, les actions décrites dans la remarque 2a se produisent.
- Dans tous les autres cas, les actions décrites dans la remarque 2c se produisent.

En raison des différences entre les environnements, assurez-vous que les applications que vous souhaitez porter valident ou réutilisent l'unité de travail avant qu'elles ne se terminent.

c. Si l'application se termine *de manière anormale* sans émettre l'appel MQDISC, l'unité de travail est annulée.

3. Sous z/OS, les points suivants s'appliquent:

- Les applications CICS n'ont pas besoin d'émettre l'appel MQDISC pour se déconnecter du gestionnaire de files d'attente, car le système CICS se connecte lui-même au gestionnaire de files d'attente et l'appel MQDISC n'a aucun effet sur cette connexion.
- CICS, IMS (autres que les programmes DL/1 par lots) et les applications RRS utilisent des unités de travail coordonnées par un coordinateur d'unité de travail externe. Par conséquent, l'appel MQDISC n'affecte pas le statut de l'unité de travail (le cas échéant) qui existe lorsque l'appel est émis.

  Toutefois, l'appel MQDISC *indique* la fin d'utilisation de la balise de connexion *ConnTag* qui a été associée à la connexion par un appel MQCONNX antérieur émis par l'application. Si une unité d'oeuvre active fait référence à la balise de connexion lors de l'émission de l'appel MQDISC, l'appel se termine avec le code achèvement MQCC_WARNING et le code anomalie MQRC_CONN_TAG_NOT_RELÂCHÉE. La balise de connexion ne peut pas être réutilisée tant que le coordinateur de l'unité de travail externe n'a pas résolu l'unité de travail.

4. Sous IBM i, les applications qui s'exécutent en mode compatibilité n'ont pas à émettre cet appel ; voir l'appel MQCONN pour plus de détails.

Remarque : Dans CICS, MQOP_START n'est pas pris en charge. A la place, utilisez l'appel de fonction MQOP_START_WAIT.

Appel C

```
MQDISC (&Hconn, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

Appel COBOL

```
CALL 'MQDISC' USING HCONN, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQDISC (Hconn, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
dcl Hconn     fixed bin(31);  /* Connection handle */
dcl CompCode  fixed bin(31);  /* Completion code */
dcl Reason    fixed bin(31);  /* Reason code qualifying CompCode */
```

Appel de l'assembleur System/390

```
CALL MQDISC,(HCONN,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Appel Visual Basic

```
MQDISC Hconn, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQDLTMH-Suppression du descripteur de message

L'appel MQDLTMH supprime un descripteur de message et est l'inverse de l'appel MQCRTMH.

Syntaxe

MQDLTMH (*Hconn*, *Hmsg*, *DltMsgHOpts*, *CompCode*, *Motif*)

Paramètres

Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente.

La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre *Hmsg* .

Si le descripteur de message a été créé à l'aide de MQHC_UNASSOCIATED_HCONN, une connexion valide doit être établie sur l'unité d'exécution qui supprime le descripteur de message, sinon l'appel échoue avec MQRC_CONNECTION_BROKEN.

Msg

Type: MQHMSG-entrée/sortie

Il s'agit du descripteur de message à supprimer. La valeur a été renvoyée par un appel MQCRTMH précédent.

Une fois l'appel terminé, le descripteur est défini sur une valeur non valide pour l'environnement. Cette valeur est:

MQHM_UNUSABLE_HMSG
Descripteur de message inutilisable.

Le descripteur de message ne peut pas être supprimé si un autre appel WebSphere MQ est en cours et a reçu le même descripteur de message.

DltMsgHOpts
Type: MQDMHO-entrée

Voir «MQDMHO-Options de suppression de descripteur de message», à la page 335 pour plus de détails.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Carte non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

MQRC_DMHO_ERREUR
(2462, X'099E') La structure des options de suppression de descripteur de message n'est pas valide.

ERREUR MQRC_HMSG_ERROR
(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Descripteur de message déjà utilisé.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Options non valides ou non cohérentes.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Appel C

```
MQDLTMH (Hconn, &Hmsg, &DltMsgHOpts, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQDLTMH' USING HCONN, HMSG, DLTMSGOPTS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** 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.
```

Appel PL/I

```
call MQDLTMH (Hconn, Hmsg, DltMsgHOpts, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQDLTMH,(HCONN,HMSG,DLTMSGHOPTS,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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-Propriété de message Delete

L'appel MQDLTMP supprime une propriété d'un descripteur de message et est l'inverse de l'appel MQSETMP.

Syntaxe

MQDLTMP (*Hconn*, *Hmsg*, *DltPropOpts*, *Nom*, *CompCode*, *Motif*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre *Hmsg* .

Si le descripteur de message a été créé à l'aide de MQHC_UNASSOCIATED_HCONN, une connexion valide doit être établie sur l'unité d'exécution en supprimant le descripteur de message, sinon l'appel échoue avec MQRC_CONNECTION_BROKEN.

Msg
Type: MQHMSG-entrée

Il s'agit du descripteur de message contenant la propriété à supprimer. La valeur a été renvoyée par un appel MQCRTMH précédent.

DltPropOpts
Type: MQDMPO-entrée

Pour plus de détails, voir le type de données MQDMPO .

Nom
Type: MQCHARV-entrée

Nom de la propriété à supprimer. Pour plus d'informations sur les noms de propriété, voir Noms de propriété .

Les caractères génériques ne sont pas autorisés dans le nom de la propriété.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7') Propriété non disponible.

MQRC_RFH_FORMAT_ERREUR
(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Carte non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'0852') Impossible de charger le module de service de l'adaptateur.

MQRC_ASID_MISMATCH
(2157, X'086D') Les ASID principal et principal diffèrent.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

MQRC_DMPO_ERREUR
(2481, X'09B1') La structure des options de suppression des propriétés de message n'est pas valide.

ERREUR MQRC_HMSG_ERROR
(2460, X'099C') Descripteur de message non valide.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Descripteur de message déjà utilisé.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Options non valides ou non cohérentes.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Nom de propriété non valide.

MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

MQRC_UNEXPECTED_ERROR
(2195, X'0893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir:

- Codes anomalie pour WebSphere MQ for z/OS
- Codes anomalie d'API pour les autres plateformes WebSphere MQ

.

Appel C

```
MQDLTMP (Hconn, Hmsg, &DltPropOpts, &Name, &CompCode, &Reason)
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQDLTMP' USING HCONN, HMSG, DLTPROPOPTS, NAME, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQDLTMP (Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```

CALL MQDLTMP,(HCONN,HMSG,DLTPROPOPTS,NAME,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```

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
```

Message d'extraction MQGET

L'appel MQGET récupère un message à partir d'une file d'attente qui a été ouverte à l'aide de l'appel MQOPEN.

Syntaxe

MQGET (*Hconn*, *Hobj*, *MsgDesc*, *GetMsgOpts*, *BufferLength*, *Buffer*, *DataLength*, *CompCode*, *Cause*)

Paramètres

Hconn

Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

Hobj
Type : MQHOBJ - entrée

Ce descripteur représente la file d'attente à partir de laquelle un message est récupéré. La valeur de *Hobj* a été renvoyée par un appel MQOPEN précédent. La file d'attente doit être ouverte avec une ou plusieurs des options suivantes (voir «MQOPEN-Objet ouvert», à la page 719) :

- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE
- MQOO_INPUT_AS_Q_DEF
- MQOO_BROWSE

MsgDesc
Type : MQMD - entrée/sortie

Cette structure décrit les attributs du message requis ainsi que ceux du message récupéré. Pour plus d'informations, voir «MQMD-Descripteur de message», à la page 394.

Si la valeur de *BufferLength* est inférieure à la longueur du message, la valeur de *MsgDesc* est remplie par le gestionnaire de files d'attente, que l'option MQGMO_ACCEPT_TRUNCATED_MSG soit spécifiée ou non sur le paramètre *GetMsgOpts* (voir MQGMO - Zone d'options).

Si l'application fournit une structure MQMD version-1, le message a renvoyé MQMDE accolé aux données de message d'application, mais *uniquement* si une ou plusieurs des zones de MQMDE possèdent une valeur autre que celle par défaut. Si toutes les zones de MQMDE possèdent des valeurs par défaut, MQMDE est omis. Un nom de format MQFMT_MD_EXTENSION dans la zone *Format* de MQMD indique qu'une structure MQMDE est présente.

L'application n'a pas besoin de fournir une structure MQMD si un descripteur de message valide est fourni dans la zone *MsgHandle*. Si cette zone n'est pas renseignée, le descripteur de message est extrait du descripteur associé aux descripteurs de messages.

Si l'application fournit un descripteur de message au lieu d'une structure MQMD et spécifie l'option MQGMO_PROPERTIES_FORCE_MQRFH2, l'appel échoue avec le code anomalie MQRC_MD_ERROR. L'appel échoue également avec le code anomalie MQRC_MD_ERROR, si l'application ne fournit pas de structure MQMD, qu'elle spécifie l'option MQGMO_PROPERTIES_AS_Q_DEF et que l'attribut de file d'attente *PropertyControl* est MQPROP_FORCE_MQRFH2.

Si des options de concordance sont spécifiées et que le descripteur de message associé est utilisé, les zones de saisie utilisées pour la correspondance proviennent du descripteur de message.

GetMsgOpts
Type : MQGMO - entrée/sortie

Voir «Options MQGMO-Get-message», à la page 345 pour plus de détails.

BufferLength
Type : MQLONG - entrée

Il s'agit de la longueur en octets de la zone *Buffer*. Indiquez 0 pour les messages qui ne comportent aucune donnée ou si le message doit être retiré de la file d'attente et les données supprimées (vous devez spécifier MQGMO_ACCEPT_TRUNCATED_MSG dans ce cas).

Remarque : La longueur du message le plus long pouvant être lu à partir de la file d'attente est spécifiée par l'attribut de file d'attente *MaxMsgLength* ; voir «Attributs des files d'attente», à la page 824.

Tampon
Type : MQBYTExBufferLength - sortie

Il s'agit de la zone devant contenir les données de message. Alignez la mémoire tampon sur une limite appropriée à la nature des données dans le message. L'alignement à 4 octets convient à la plupart des messages (y compris les messages contenant des structures d'en-tête IBM WebSphere MQ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message comportant un entier binaire de 64 bits peut avoir besoin d'un alignement sur 8 octets.

Si la valeur de *BufferLength* est inférieure à la longueur du message, la plus grande partie possible du message est déplacée dans *Buffer*. Cette situation se produit que l'option MQGMO_ACCEPT_TRUNCATED_MSG soit spécifiée ou non sur le paramètre *GetMsgOpts* (pour plus d'informations, voir MQGMO - Zone d'options).

Le jeu de caractères et le codage des données dans *Buffer* sont fournis par les zones *CodedCharSetId* et *Encoding* renvoyées dans le paramètre *MsgDesc*. Si ces valeurs sont différentes des valeurs requises par le récepteur, ce dernier doit convertir les données de message d'application dans le jeu de caractères et le codage requis. L'option MQGMO_CONVERT peut être utilisée (avec un exit écrit par l'utilisateur, si nécessaire) pour convertir les données de message ; voir «Options MQGMO-Get-message», à la page 345 pour en savoir plus sur cette option.

Remarque : Tous les autres paramètres de l'appel MQGET se trouvent dans le jeu de caractères et le codage du gestionnaire de files d'attente local (fourni par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et MQENC_NATIVE).

Si l'appel échoue, le contenu de la mémoire tampon peut avoir changé.

En langage de programmation C, le paramètre est déclaré en tant que pointeur nul : vous pouvez spécifier l'adresse de n'importe quel type de données en tant que paramètre.

Si le paramètre *BufferLength* a pour valeur zéro, *Buffer* n'est pas référencé ; dans ce cas, il se peut que l'adresse de paramètre transmise par les programmes écrits en langage C ou assembleur System/390 ait pour valeur NULL.

DataLength
Type : MQLONG - sortie

Il s'agit de la longueur en octets des données d'application *dans le message*. Si cette valeur est supérieure à *BufferLength*, seuls les octets correspondant à la capacité de la mémoire tampon *BufferLength* sont renvoyés dans le paramètre *Buffer* (autrement dit, le message est tronqué). Si la valeur est zéro, le message ne contient aucune donnée d'application.

Si la valeur de *BufferLength* est inférieure à la longueur du message, la valeur de *DataLength* est remplie par le gestionnaire de files d'attente, que l'option MQGMO_ACCEPT_TRUNCATED_MSG soit spécifiée ou non sur le paramètre *GetMsgOpts* (pour plus d'informations, voir MQGMO - Zone d'options). L'application peut ainsi choisir la taille de la mémoire tampon requise pour contenir les données de message, puis exécuter de nouveau l'appel avec une taille de mémoire tampon appropriée.

Toutefois, si l'option MQGMO_CONVERT est spécifiée et les données de message converties sont trop longues pour être stockées dans la mémoire *Buffer*, la valeur renvoyée pour *DataLength* correspond à :

- la longueur des données *non converties* des formats définis par le gestionnaire de files d'attente.

  Dans ce cas, si la nature des données provoque leur accroissement pendant la conversion, l'application doit allouer une mémoire tampon supérieure à la valeur renvoyée par le gestionnaire de files d'attente pour *DataLength*.
- la valeur renvoyée par l'exit de conversion de données pour les formats d'application.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison

Type : MQLONG - sortie

Les codes anomalie répertoriés sont ceux que le gestionnaire de files d'attente peut renvoyer pour le paramètre *Reason*. Si l'application spécifie l'option MQGMO_CONVERT et qu'un exit écrit par l'utilisateur est appelé pour convertir certaines ou toutes les données de message, l'exit choisit la valeur renvoyée pour le paramètre *Reason*. Par conséquent, les valeurs autres que celles décrites sont possibles.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_CONVERTED_MSG_TOO_BIG
(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

MQRC_CONVERTED_STRING_TOO_BIG
(2190, X'88E') Chaîne convertie trop volumineuse pour la zone.

MQRC_DBCS_ERROR
(2150, X'866') Chaîne DBCS non valide.

MQRC_FORMAT_ERROR
(2110, X'83E') Format de message non valide.

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Groupe de messages non complet.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Message logique non complet.

MQRC_INCONSISTENT_CCSIDS
(2243, X'8C3') Les segments de message possèdent des CCSID différents.

MQRC_INCONSISTENT_ENCODINGS
(2244, X'8C4') Les segments de message possèdent des codages différents.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Utilisation de jeton de message non valide.

MQRC_NO_MSG_LOCKED
(2209, X'8A1') Aucun message verrouillé.

MQRC_NOT_CONVERTED
(2119, X'847') Données de message non converties.

MQRC_OPTIONS_CHANGED
(nnnn, X'xxx') Les options devant être cohérentes ont été modifiées.

MQRC_PARTIALLY_CONVERTED
(2272, X'8E0') Données de message partiellement converties.

MQRC_SIGNAL_REQUEST_ACCEPTED
(2070, X'816') Aucun message renvoyé (mais la demande de signal a été acceptée).

MQRC_SOURCE_BUFFER_ERROR
(2145, X'861') Paramètre de tampon source non valide.

MQRC_SOURCE_CCSID_ERROR
(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

MQRC_SOURCE_DECIMAL_ENC_ERROR
(2113, X'841') Codage décimal condensé dans le message non reconnu.

MQRC_SOURCE_FLOAT_ENC_ERROR
(2114, X'842') Codage à virgule flottante dans le message non reconnu.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840') Codage d'entier source non reconnu.

MQRC_SOURCE_LENGTH_ERROR
(2143, X'85F') Paramètre de longueur source non valide.

MQRC_TARGET_BUFFER_ERROR
(2146, X'862') Paramètre de tampon cible non valide.

MQRC_TARGET_CCSID_ERROR
(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

MQRC_TARGET_DECIMAL_ENC_ERROR
(2117, X'845') Codage décimal condensé spécifié par le récepteur non reconnu.

MQRC_TARGET_FLOAT_ENC_ERROR
(2118, X'846') Codage à virgule flottante spécifié par le récepteur non reconnu.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844') Codage d'entier cible non reconnu.

MQRC_TRUNCATED_MSG_ACCEPTED
(2079, X'81F') Message tronqué renvoyé (traitement terminé).

MQRC_TRUNCATED_MSG_FAILED
(2080, X'820') Message tronqué renvoyé (traitement non terminé).

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855') Impossible de charger les modules de service de conversion de données.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BACKED_OUT
(2003, X'7D3') Unité d'oeuvre annulée.

MQRC_BUFFER_ERROR
(2004, X'7D4') Paramètre de tampon non valide.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CF_STRUC_FAILED
(2373, X'945') Echec de la structure de l'unité de couplage.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347, X'92B') En-tête de liste de la structure d'unité de couplage en cours d'utilisation.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Connexion en cours de mise au repos.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_CORREL_ID_ERROR
(2207, X'89F') Erreur d'identificateur de corrélation.

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Paramètre de longueur des données non valide.

MQRC_DB2_NOT_AVAILABLE
(2342, X'926') Sous-système DB2 non disponible.

MQRC_GET_INHIBITED
(2016, X'7E0') Se bloque pour la file d'attente.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Conflit entre les unités d'oeuvre globales.

MQRC_GMO_ERROR
(2186, X'88A') Structure des options de message Get non valide.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HOBJ_ERROR
(2019, X'7E3') Descripteur d'objet non valide.

MQRC_INCONSISTENT_BROWSE
(2259, X'8D3') Spécification de navigation incohérente.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

MQRC_INVALID_MSG_UNDER_CURSOR
(2246, X'8C6') Message sous le curseur non valide pour la récupération.

MQRC_LOCAL_UOW_CONFLICT
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Options de correspondance non valides.

MQRC_MD_ERROR
(2026, X'7EA') Descripteur de message non valide.

MQRC_MSG_ID_ERROR
(2206, X'89E') Erreur d'identificateur de message.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Numéro de séquence de message non valide.

MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Utilisation du jeton de message non valide.

MQRC_NO_MSG_AVAILABLE
(2033, X'7F1') Aucun message disponible.

MQRC_NO_MSG_UNDER_CURSOR
(2034, X'7F2') Curseur d'exploration non positionné sur le message.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') File d'attente non ouverte pour exploration.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') File d'attente non ouverte pour saisie.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_Q_DELETED
(2052, X'804') File d'attente supprimée.

MQRC_Q_INDEX_TYPE_ERROR
(2394, X'95A') Type d'index incorrect pour la file d'attente.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_QUIESCING
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SECOND_MARK_NOT_ALLOWED
(2062, X'80E') Un message est déjà marqué.

MQRC_SIGNAL_OUTSTANDING
(2069, X'815') Signal en attente pour ce descripteur.

MQRC_SIGNAL1_ERROR
(2099, X'833') Zone de signal non valide.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_SYNCPOINT_LIMIT_REACHED
(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818') Prise en charge du point de synchronisation non disponible.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Intervalle d'attente non valide dans MQGMO.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Version MQGMO fournie non valide.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Version MQMD fournie non valide.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. Le message extrait est normalement supprimé de la file d'attente. Cette suppression peut avoir lieu dans le cadre de l'appel MQGET proprement dit ou d'un point de synchronisation.

   Les options d'exploration sont les suivantes : MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXT et MQGMO_BROWSE_MSG_UNDER_CURSOR.

2. Si l'option MQGMO_LOCK est spécifiée à l'aide d'une des options d'exploration, le message exploré est verrouillé de sorte qu'il ne soit visible que par ce descripteur.

   Si l'option MQGMO_UNLOCK est spécifiée, un message précédemment verrouillé est déverrouillé. Dans ce cas, aucun message n'est extrait et les paramètres *MsgDesc*, *BufferLength*, *Buffer* et *DataLength* ne sont pas vérifiés ou modifiés.

3. Pour les applications émettant un appel MQGET, le message extrait peut être perdu si l'application se termine de façon anormale ou si la connexion est interrompue pendant le traitement de l'appel. Ce problème se produit du fait que le substitut en cours d'exécution sur la même plateforme que le gestionnaire de files d'attente qui émet l'appel MQGET pour le compte de l'application ne peut pas détecter la perte de l'application jusqu'à ce que le substitut soit sur le point de renvoyer le message à l'application, *après* la suppression du message de la file d'attente. Cela peut se produire tant pour les messages persistants que pour les messages non persistants.

   Pour éliminer le risque de perte de messages de cette façon, extrayez systématiquement les messages dans des unités d'oeuvre. C'est-à-dire, en spécifiant l'option MQGMO_SYNCPOINT sur l'appel MQGET et en utilisant l'appel MQCMIT ou MQBACK pour valider ou annuler l'unité d'oeuvre une fois le traitement des messages terminé. Si l'option MQGMO_SYNCPOINT est spécifiée et que le client prend fin de façon anormale ou que la connexion est interrompue, le substitut annule l'unité d'oeuvre sur le gestionnaire de files d'attente et le message est rétabli dans la file d'attente. Pour plus d'informations sur les points de synchronisation, voir les remarques relatives aux points de synchronisation dans les applications WebSphere MQ.

   Cette situation peut se produire avec les clients IBM WebSphere MQ ainsi qu'avec les applications qui s'exécutent sur la même plateforme que le gestionnaire de files d'attente.

4. Si une application place une séquence de messages dans unefile d'attente particulière au sein d'une unité d'oeuvre unique et qu'elle valide ensuite cette unité d'oeuvre correctement, les messages deviennent disponibles pour extraction, comme suit :

   - Si la file d'attente est une file d'attente *non partagée* (c-à-d, une file d'attente locale), tous les messages au sein de l'unité d'oeuvre deviennent disponibles en même temps.
   - Si la file d'attente est une file d'attente *partagée*, les messages au sein de l'unité d'oeuvre deviennent disponibles dans l'ordre dans lequel ils ont été insérés, mais pas tous en même temps. Lorsque le système est extrêmement sollicité, il est possible que le premier message de l'unité d'oeuvre soit extrait correctement mais que l'appel MQGET correspondant au deuxième message ou à un message ultérieur dans l'unité d'oeuvre échoue en raison du code MQRC_NO_MSG_AVAILABLE. Si cela se produit, l'application doit attendre un petit moment avant de relancer l'opération.

5. Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages,l'ordre de ces messages est préservé sous réserve que certaines conditions soient satisfaites. Pour plus de détails, voir Remarques sur l'utilisation de MQPUT. Si les conditions sont satisfaites, les messages sont présentés à l'application réceptrice dans l'ordre dans lequel ils ont été envoyés, sous les réserves suivantes :

   - Un seul récepteur extraie des messages de la file d'attente.

     Si deux applications ou plus extraient des messages de la file d'attente, elles doivent convenir avec l'émetteur du mécanisme à utiliser pour identifier les messages faisant partie d'une séquence. Par

exemple, l'émetteur peut paramétrer toutes les zones *CorrelId* des messages d'une séquence sur une valeur qui était unique à cette séquence de messages.

- Le récepteur ne modifie pas intentionnellement l'ordre de l'extraction, par exemple en spécifiant un paramètre *MsgId* ou *CorrelId* particulier.

Si l'application émettrice place les messages sous la forme d'un groupe de messages, les messages sont présentés à l'application réceptrice dans l'ordre correct sous réserve que l'application réceptrice spécifie l'option MQGMO_LOGICAL_ORDER dans l'appel MQGET. Pour plus d'informations sur les groupes de messages, voir :

- MQMD - Zone MsgFlags
- MQPMO_LOGICAL_ORDER
- MQGMO_LOGICAL_ORDER

Si l'utilisateur extrait des messages d'un groupe dans le cadre d'un point de synchronisation, il doit s'assurer que le groupe entier est traité avant de tenter de terminer la transaction.

6. Les applications doivent rechercher le code retour MQFB_QUIT de la zone *Feedback* du paramètre *MsgDesc* et prendre fin si elles trouvent cette valeur. Pour plus d'informations, voir MQMD - Zone Feedback.
7. Si la file d'attente identifiée par *Hobj* a été ouverte à l'aide de l'option MQOO_SAVE_ALL_CONTEXT et que le code achèvement généré par l'appel MQGET est MQCC_OK ou MQCC_WARNING, le contexte associé au descripteur de file d'attente *Hobj* est paramétré sur le contexte du message extrait (sauf si l'option MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXT, ou MQGMO_BROWSE_MSG_UNDER_CURSOR est définie, auquel cas le contexte est signalé comme étant non disponible).

   Vous pouvez utiliser le contexte enregistré d'un appel MQPUT ou MQPUT1 ultérieur en spécifiant les options MQPMO_PASS_IDENTITY_CONTEXT ou MQPMO_PASS_ALL_CONTEXT. Cela permet de transférer en intégralité ou en partie le contexte du message reçu vers un autre message (par exemple, lorsque le message est réacheminé vers une autre file d'attente). Pour plus d'informations sur le contexte de message, voir Contexte de message.
8. Si vous incluez l'option MQGMO_CONVERT au paramètre *GetMsgOpts*, les données de message d'application sont converties en représentation demandée par l'application réceptrice, avant que les données soient placées dans le paramètre *Buffer* :

   - La zone *Format* des informations de contrôle dans le message identifie la structure des données d'application et les zones *CodedCharSetId* et *Encoding* des informations de contrôle dans le message définissent son identificateur de jeu de caractères et son codage.
   - L'application émettant l'appel MQGET spécifie dans les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* l'identificateur de jeu de caractères et le codage auxquels convertir les données du message d'application.

   Lorsque la conversion des données de message est nécessaire, la conversion est exécutée par le gestionnaire de files d'attente proprement dit ou par un exit écrit par l'utilisateur, en fonction de la valeur de la zone *Format* des informations de contrôle dans le message :

   - Les noms de format suivants correspondent aux formats convertis par le gestionnaire de files d'attente ; ces formats sont appelés formats "intégrés" :
     - MQFMT_ADMIN
     - MQFMT_CICS (z/OS uniquement)
     - MQFMT_COMMAND_1
     - MQFMT_COMMAND_2
     - MQFMT_DEAD_LETTER_HEADER
     - MQFMT_DIST_HEADER
     - MQFMT_EVENT version 1
     - MQFMT_EVENT version 2 (z/OS uniquement)

  - MQFMT_IMS
  - MQFMT_IMS_VAR_STRING
  - MQFMT_MD_EXTENSION
  - MQFMT_PCF
  - MQFMT_REF_MSG_HEADER
  - MQFMT_RF_HEADER
  - MQFMT_RF_HEADER_2
  - MQFMT_STRING
  - MQFMT_TRIGGER
  - MQFMT_WORK_INFO_HEADER (z/OS uniquement)
  - MQFMT_XMIT_Q_HEADER
- Le nom de format MQFMT_NONE est une valeur spéciale indiquant que la nature des données du message n'est pas définie. En conséquence, le gestionnaire de files d'attente ne tente pas la conversion lorsque le message est extrait de la file d'attente.

  Remarque : Si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET pour un message dont le nom de format est MQFMT_NONE et que le jeu de caractères ou le codage du message est différent de celui spécifié dans le paramètre *MsgDesc*, le message est renvoyé dans le paramètre *Buffer* (en supposant aucune autre erreur), mais l'appel se termine avec le code achèvement MQCC_WARNING et le code anomalie MQRC_FORMAT_ERROR.

  Vous pouvez utiliser MQFMT_NONE lorsque la nature des données de message signifie qu'une conversion n'est pas nécessaire ou lorsque les applications émettrice et réceptrice sont convenus du format d'envoi des données de message.
- Tous les autres noms de format transmettent le message à un exit écrit par l'utilisateur, à des fins de conversion. L'exit porte le même nom que le format, hormis des ajouts spécifiques à l'environnement. Les noms de format définis par l'utilisateur ne doivent pas commencer par les lettres WebSphere MQ.

  Pour plus de détails sur l'exit de conversion de données, voir «Exit de conversion de données», à la page 898.

Les données utilisateur du message peuvent être converties d'un jeu de caractères pris en charge à l'autre et d'un codage à l'autre. Cependant, si le message contient une ou plusieurs structures d'en-tête WebSphere MQ, le message ne peut pas être converti à partir ou vers un jeu de caractères codés sur deux octets ou sur plusieurs octets pour n'importe quel caractère valide des noms de file d'attente. Cette tentative donne lieu au code anomalie MQRC_SOURCE_CCSID_ERROR ou MQRC_TARGET_CCSID_ERROR et le message est renvoyé non converti. Le jeu de caractères Unicode UCS-2 est un exemple de jeu de caractères de ce type.

Renvoyé par MQGET, le code anomalie suivant indique que la conversion du message a abouti :

- MQRC_NONE

Le code anomalie suivant indique que la conversion du message a *peut-être* abouti ; pour déterminer cela, l'application doit vérifier les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* :

- MQRC_TRUNCATED_MSG_ACCEPTED

Tous les autres codes anomalie indiquent que le message n'a pas été converti.

Remarque : L'interprétation de ce code anomalie est importante pour les conversions exécutées par un exit écrit par l'utilisateur *seulement* si l'exit est conforme aux instructions de traitement décrites dans la rubrique «Exit de conversion de données», à la page 898.

9. Lors de l'utilisation de l'interface orientée objet pour l'extraction des messages, vous pouvez choisir de ne pas indiquer de mémoire tampon pour le stockage des données de message pour un appel MQGET. Toutefois, dans les précédentes versions de WebSphere MQ, il était possible que MQGET échoue en raison du code anomalie MQRC_CONVERTED_MSG_TO_BIG, même si aucune mémoire

tampon n'avait été indiquée. Dans WebSphere MQ version 7, lors de l'extraction d'un message à l'aide d'une application orientée objet sans restriction de la taille de la mémoire tampon de réception de message, l'application n'échoue pas en raison du code MQRC_CONVERTED_MSG_TOO_BIG et reçoit le message converti. Cela concerne les environnements suivants :

- .NET, y compris les applications gérées en intégralité
- C++
- Java (WebSphere MQ classes for Java)

Remarque : Pour tous les clients, si *sharingConversations* a pour valeur zéro, le canal fonctionne comme dans les versions antérieures à WebSphere MQ version 7.0 et le traitement des messages reprend le comportement de la version 6. Dans cette situation, si la mémoire tampon est trop petite pour recevoir le message converti, le message non converti est renvoyé avec le code anomalie MQRC_CONVERTED_MSG_TOO_BIG. Pour plus d'informations sur *sharingConversations*, voir Utilisation du partage de conversations dans une application client.

10. Pour les formats intégrés, le gestionnaire de files d'attente peut exécuter la *conversion par défaut* des chaînes de caractères du message lorsque l'option MQGMO_CONVERT est spécifiée. La conversion par défaut permet au gestionnaire de files d'attente d'utiliser, lors de la conversion des données de type chaîne, un jeu de caractères par défaut spécifié par l'installation qui se rapproche du jeu de caractères réel. En conséquence, l'appel MQGET peut aboutir avec le code achèvement MQCC_OK, au lieu de se terminer avec MQCC_WARNING et le code anomalie MQRC_SOURCE_CCSID_ERROR ou MQRC_TARGET_CCSID_ERROR.

    Remarque : L'utilisation d'un jeu de caractères approximatif pour la conversion des données de type chaîne donne lieu à la conversion éventuellement incorrecte de certains caractères. Pour éviter cela, utilisez des caractères dans la chaîne qui sont communs au jeu de caractères réel et au jeu de caractères par défaut.

    La conversion par défaut s'applique aux données de message d'application et aux zones alphanumériques des structures MQMD et MQMDE :

    - La conversion par défaut des données de message d'application n'a lieu que lorsque *toutes* les conditions suivantes sont réunies :
        - L'application spécifie MQGMO_CONVERT.
        - Le message contient des données qui doivent être converties à partir ou vers un jeu de caractères non pris en charge.
        - La conversion par défaut a été activée lors de l'installation ou du redémarrage du gestionnaire de files d'attente.
    - La conversion par défaut des zones alphanumériques dans les structures MQMD et MQMDE a lieu selon les besoins, si la conversion par défaut est activée pour le gestionnaire de files d'attente. La conversion est exécutée même si l'option MQGMO_CONVERT n'est pas spécifiée par l'application lors de l'appel MQGET.
11. Pour le langage de programmation Visual Basic, les points suivants s'appliquent :

    - Si la taille du paramètre *Buffer* est inférieure à la longueur indiquée par le paramètre *BufferLength*, l'appel échoue en raison du code anomalie MQRC_STORAGE_NOT_AVAILABLE.
    - Le paramètre *Buffer* est déclaré comme étant de type `String`. Si les données à extraire de la file d'attente ne sont pas de type `String`, utilisezl'appel MQGETAny à la place de MQGET.

      L'appel MQGETAny comporte les mêmes paramètres que l'appel MQGET, à ceci près que le paramètre *Buffer* est déclaré comme étant de type Any, ce qui permet l'extraction de n'importe quel type de données. Cependant, cela signifie que le système ne peut pas vérifier que le paramètre *Buffer* comporte au minimum *BufferLength* octets.
12. Les options MQGET ne sont pas toutes prises en charge lorsque la lecture anticipée est activée. Le tableau ci-dessous présente les options admises et indique si elles peuvent être modifiées d'un appel MQGET à l'autre.

Tableau 566. Options MQGET admises lorsque la lecture anticipée est activée			
	Admise lorsque la lecture anticipée est activée, et modifiable d'un appel MQGET à l'autre	**Admise lorsque la lecture anticipée est activée, mais non modifiable d'un appel MQGET à l'autre[a]**	**Options MQGET non admises lorsque la lecture anticipée est activée[b]**
Valeurs MQGET MD	MsgId[c] CorrelId[c]	Codage CodedCharSetId	
Options MQGET MQGMO	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 MQGMO_CONVERT MQGMO_LOGICAL_ORDER MQGMO_COMPLETE_MSG MQGMO_ALL_MSGS_AVAILABLE MQGMO_ALL_SEGMENTS_AVAILABLE MQGMO_MARK_BROWSE_HANDLE MQGMO_MARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_HANDLE MQGMO_UNMARKED_BROWSE_MSG MQGMO_PROPERTIES_FORCE_MQRFH2 MQGMO_NO_PROPERTIES MQGMO_PROPERTIES_IN_HANDLE MQGMO_PROPERTIES_COMPATIBILITY	MQGMO_SET_SIGNAL MQGMO_SYNCPOINT MQGMO_MARK_SKIP _BACKOUT MQGMO_MSG_UNDER _CURSOR[d] MQGMO_LOCK MQGMO_UNLOCK
Valeurs MQGMO		MsgHandle	

a. Si ces options sont modifiées d'un appel MQGET à l'autre, un code anomalie MQRC_OPTIONS_CHANGED est renvoyé.

b. Si ces options sont spécifiées lors du premier appel MQGET, la lecture anticipée est désactivée. Si ces options sont spécifiées lors d'un appel MQGET ultérieur, un code anomalie MQRC_OPTIONS_ERROR est renvoyé.

c. Les applications client doivent être informées que, si les valeurs MsgId et CorrelId sont modifiées d'un appel MQGET à l'autre, il se peut que les messages comportant les valeurs précédentes aient déjà été envoyés au client et qu'ils restent dans la mémoire tampon de lecture anticipée du client tant qu'ils ne sont pas consommés (ou automatiquement purgés).

d. Le premier appel MQGET détermine si les messages doivent être explorés ou extraits d'une file d'attente lorsque la lecture anticipée est activée. Si l'application tente d'utiliser une combinaison d'exploration et d'extraction, un code anomalie MQRC_OPTIONS_CHANGED est renvoyé.

e. MQGMO_MSG_UNDER_CURSOR n'est pas possible avec la lecture anticipée. Les messages peuvent être explorés ou extraits lorsque la lecture anticipée est activée mais pas une combinaison des deux.

13. Les applications peuvent extraire de façon destructive des messages non validés seulement si ces messages sont placés dans la même unité d'oeuvre locale que l'unité d'oeuvre d'extraction. Les applications ne peuvent pas extraire de messages de façon non destructive.
14. Les messages indiqués par un curseur d'exploration peuvent être extraits dans une unité d'oeuvre. Cette méthode ne permet pas d'extraire un message non validé.

Appel C

```
MQGET (Hconn, Hobj, &MsgDesc, &GetMsgOpts, BufferLength, Buffer,
      &DataLength, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQGET' USING HCONN, HOBJ, MSGDESC, GETMSGOPTS, BUFFERLENGTH,
BUFFER, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQGET (Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,
           DataLength, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
         CALL MQGET,(HCONN,HOBJ,MSGDESC,GETMSGOPTS,BUFFERLENGTH,
               BUFFER,DATALENGTH,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Appel Visual Basic

```
MQGET Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,
DataLength, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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-Attributs d'objet Inquire

L'appel MQINQ renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'un objet.

Les types d'objet suivants sont valides:

- Gestionnaire de files d'attente
- File d'attente
- Liste de noms
- Définition de processus

Syntaxe

MQINQ (*Hconn*, *Hobj*, *SelectorCount*, *Sélecteurs*, *IntAttrCount*, *IntAttrs*, *CharAttrLength*, *CharAttrs*, *CompCode*, *Motif*)

Paramètres

Hconn
Type: MQHCONN -entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

Hobj
Type: MQHOBJ -entrée

Ce descripteur représente l'objet (de tout type) avec les attributs requis. Le descripteur doit être renvoyé par un appel MQOPEN précédent qui a spécifié l'option MQOO_INQUIRE .

SelectorCount
Type: MQLONG -entrée

Nombre de sélecteurs fournis dans le tableau *Selectors* . Il s'agit du nombre d'attributs à renvoyer. Zéro est une valeur valide. Le nombre maximal autorisé est 256.

Sélecteurs
Type: MQLONG × *SelectorCount* -entrée

Il s'agit d'un tableau de sélecteurs d'attribut *SelectorCount* ; chaque sélecteur identifie un attribut (entier ou caractère) avec une valeur obligatoire.

Chaque sélecteur doit être valide pour le type d'objet que *Hobj* représente, sinon l'appel échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_SELECTOR_ERROR.

Dans le cas particulier des files d'attente:

- Si le sélecteur n'est pas valide pour les files d'attente de n'importe quel type, l'appel échoue avec le code achèvement MQCC_FAILED et le code anomalie MQRC_SELECTOR_ERROR.
- Si le sélecteur s'applique uniquement aux files d'attente de types autres que le type de l'objet, l'appel aboutit avec le code achèvement MQCC_WARNING et le code anomalie MQRC_SELECTOR_NOT_FOR_TYPE.
- Si la file d'attente en cours d'interrogation est une file d'attente de cluster, les sélecteurs valides dépendent de la manière dont la file d'attente a été résolue ; voir «Notes d'utilisation», à la page 707 pour plus de détails.

Vous pouvez spécifier des sélecteurs dans n'importe quel ordre. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut de type entier (sélecteursMQIA_*) sont renvoyées dans *IntAttrs* dans l'ordre dans lequel ces sélecteurs apparaissent dans *Selectors*. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut de caractères (sélecteursMQCA_*) sont renvoyées dans *CharAttrs* dans l'ordre dans lequel ces sélecteurs se produisent. Les sélecteurs MQIA_* peuvent être imbriqués avec les sélecteurs MQCA_* ; seul l'ordre relatif au sein de chaque type est important.

Remarque :

1. Les sélecteurs d'attribut de type entier et caractère sont alloués dans deux plages différentes ; les sélecteurs MQIA_* résident dans la plage MQIA_FIRST à MQIA_LASTet les sélecteurs MQCA_* dans la plage MQCA_FIRST à MQCA_LAST.

   Pour chaque plage, les constantes MQIA_LAST_USED et MQCA_LAST_USEDdéfinissent la valeur la plus élevée acceptée par le gestionnaire de files d'attente.
2. Si tous les sélecteurs MQIA_* apparaissent en premier, les mêmes numéros d'élément peuvent être utilisés pour adresser les éléments correspondants dans les tableaux *Selectors* et *IntAttrs* .
3. Si le paramètre *SelectorCount* a pour valeur zéro, il n'est pas fait référence à *Selectors* . Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en langage C ou en assembleur S/390 peut être null.

Les attributs que vous pouvez consulter sont répertoriés dans les tableaux suivants. Pour les sélecteurs MQCA_* , la constante qui définit la longueur en octets de la chaîne résultante dans *CharAttrs* est fournie entre parenthèses.

Les tableaux qui suivent répertorient les sélecteurs, par objet, dans l'ordre alphabétique, comme suit:

- Sélecteurs d'attribut Tableau 567, à la page 693 MQINQ pour les files d'attente
- Sélecteurs d'attribut Tableau 568, à la page 695 MQINQ pour les listes de noms
- Sélecteurs d'attribut Tableau 569, à la page 696 MQINQ pour les définitions de processus
- Sélecteurs d'attribut Tableau 570, à la page 696 MQINQ pour le gestionnaire de files d'attente

Tous les sélecteurs sont pris en charge sur toutes les plateformes IBM WebSphere MQ , sauf indication contraire dans la colonne **Remarque** comme suit:

NONz/OS
Pris en charge sur toutes les plateformes **sauf** z/OS

z/OS
Pris en charge **uniquement** sur z/OS

Tableau 567. Sélecteurs d'attribut MQINQ pour les files d'attente

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Date de la modification la plus récente	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Heure de l'altération la plus récente	
MQCA_BACKOUT_REQ_Q_NAME	MQ_Q_NAME_LENGTH	Nombre excessif de noms de remise en file d'attente d'annulation	
MQCA_BASE_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente dans laquelle l'alias est résolu	
MQCA_CF_STRUC_NAME	MQ_CF_STRUC_NAME_LENGTH	Nom de la structure d'unité de couplage	z/OS
MQCA_CLUS_CHL_NAME	MQ_CHANNEL_NAME_LENGTH	Nom du canal émetteur de cluster qui utilise cette file d'attente comme file d'attente de transmission.	NONz/OS
MQCA_CLUSTER_NAME	MQ_CLUSTER_NAME_LENGTH	Nom du cluster	
MQCA_CLUSTER_NAMELIST	MQ_NAMELIST_NAME_LENGTH	Liste de noms de cluster	
MQCA_CREATION_DATE	MQ_CREATION_DATE_LENGTH	Date de création de file d'attente	
MQCA_CREATION_TIME	MQ_CREATION_TIME_LENGTH	Heure de création de file d'attente	
MQCA_INITIATION_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente d'initialisation	
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	Nom de la définition de processus	
MQCA_Q_DESC	MQ_Q_DESC_LENGTH	Description de file d'attente	
MQCA_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente	
MQCA_REMOTE_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	Nom du gestionnaire de files d'attente éloignées	
MQCA_REMOTE_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente éloignée connue sur le gestionnaire de files d'attente éloignées	
MQCA_STORAGE_CLASS	MQ_STORAGE_CLASS_LENGTH	Nom de la classe de stockage	z/OS
MQCA_TRIGGER_DATA	MQ_TRIGGER_DATA_LENGTH	Données de déclenchement	
MQCA_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente de transmission	
MQIA_ACCOUNTING_Q	MQLONG	Contrôle la collecte des données comptables pour la file d'attente	NONz/OS
MQIA_BACKOUT_THRESHOLD	MQLONG	Seuil d'annulation	
MQIA_CLWL_Q_PRIORITY	MQLONG	Priorité de la file d'attente	
MQIA_CLWL_Q_RANK	MQLONG	Rang de la file d'attente	

Tableau 567. Sélecteurs d'attribut MQINQ pour les files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_CLWL_USEQ	MQLONG	Utilisation des files d'attente éloignées	
MQIA_CURRENT_Q_DEPTH	MQLONG	Nombre de messages dans la file d'attente	
MQIA_DEF_BIND	MQLONG	Liaison par défaut	
MQIA_DEF_INPUT_OPEN_OPTION	MQLONG	Option open-for-input par défaut	
MQIA_DEF_PERSISTENCE	MQLONG	Persistance par défaut	
MQIA_DEF_PRIORITY	MQLONG	Priorité par défaut	
MQIA_DEFINITION_TYPE	MQLONG	Type de définition de file d'attente	
MQIA_DIST_LISTS	MQLONG	Prise en charge de la liste de diffusion	NONz/OS
MQIA_HARDEN_GET_BACKOUT	MQLONG	Indique si le nombre d'annulations doit être durci	
MQIA_INDEX_TYPE	MQLONG	Type d'index géré pour la file d'attente	z/OS
MQIA_INHIBIT_GET	MQLONG	Indique si les opérations d'extraction sont autorisées	
MQIA_INHIBIT_PUT	MQLONG	Indique si les opérations d'insertion sont autorisées	
MQIA_MAX_MSG_LENGTH	MQLONG	Longueur maximale des messages	
MQIA_MAX_Q_DEPTH	MQLONG	Nombre maximal de messages autorisés dans la file d'attente	
MQIA_MSG_DELIVERY_SEQUENCE	MQLONG	Indique si la priorité du message est pertinente	
MQIA_NPM_CLASS	MQLONG	Niveau de fiabilité des messages non persistants	
MQIA_OPEN_INPUT_COUNT	MQLONG	Nombre d'appels MQOPEN ayant la file d'attente ouverte pour l'entrée	
MQIA_OPEN_OUTPUT_COUNT	MQLONG	Nombre d'appels MQOPEN dont la file d'attente est ouverte pour la sortie	
MQIA_PROPERTY_CONTROL	MQLONG	Attribut de contrôle de propriété	
MQIA_Q_DEPTH_HIGH_EVENT	MQLONG	Attribut de contrôle pour les événements de longueur élevée de file d'attente	NONz/OS
MQIA_Q_DEPTH_HIGH_LIMIT	MQLONG	Limite supérieure de la longueur de la file d'attente	NONz/OS
MQIA_Q_DEPTH_LOW_EVENT	MQLONG	Attribut de contrôle pour les événements de longueur faible de la file d'attente	NONz/OS
MQIA_Q_DEPTH_LOW_LIMIT	MQLONG	Limite inférieure de la longueur de la file d'attente	NONz/OS

Tableau 567. Sélecteurs d'attribut MQINQ pour les files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_Q_DEPTH_MAX_EVENT	MQLONG	Attribut de contrôle du nombre maximal d'événements de longueur de file d'attente	NONz/OS
MQIA_Q_SERVICE_INTERVAL	MQLONG	Limite pour l'intervalle de service de file d'attente	NONz/OS
MQIA_Q_SERVICE_INTERVAL_EVENT	MQLONG	Attribut de contrôle pour les événements d'intervalle de service de file d'attente	NONz/OS
MQIA_Q_TYPE	MQLONG	Type de file d'attente	
MQIA_QSG_DISP	MQLONG	Disposition de groupe de partage de files d'attente	z/OS
MQIA_RETENTION_INTERVAL	MQLONG	Intervalle de conservation des files d'attente	
MQIA_SCOPE	MQLONG	Portée de définition de file d'attente	NONz/OS
MQIA_SHAREABILITY	MQLONG	Indique si la file d'attente peut être partagée pour l'entrée	
MQIA_STATISTICS_Q	MQLONG	Contrôle la collecte des données statistiques pour la file d'attente	NONz/OS
MQIA_TRIGGER_CONTROL	MQLONG	Contrôle du déclenchement	
MQIA_TRIGGER_DEPTH	MQLONG	Longueur de déclenchement	
MQIA_TRIGGER_MSG_PRIORITY	MQLONG	Priorité des messages de seuil des déclencheurs	
MQIA_TRIGGER_TYPE	MQLONG	Type de déclenchement	
MQIA_USAGE	MQLONG	Utilisation	

Tableau 568. Sélecteurs d'attribut MQINQ pour les listes de noms

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Date de la modification la plus récente	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Heure de l'altération la plus récente	
MQCA_NAMELIST_DESC	MQ_NAMELIST_DESC_LENGTH	Description de la liste de noms	
MQCA_NAMELIST_NAME	MQ_NAMELIST_NAME_LENGTH	Nom de l'objet liste de noms	
MQIA_NAMELIST_TYPE	MQLONG	Type de liste de noms	z/OS
MQCA_NAMES	MQ_Q_NAME_LENGTH × Nombre de noms dans la liste	Noms dans la liste de noms	

Tableau 568. Sélecteurs d'attribut MQINQ pour les listes de noms (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_NAME_COUNT	MQLONG	Nombre de noms dans la liste de noms	
MQIA_QSG_DISP	MQLONG	Disposition de groupe de partage de files d'attente	z/OS

Tableau 569. Sélecteurs d'attribut MQINQ pour les définitions de processus

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Date de la modification la plus récente	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Heure de l'altération la plus récente	
MQCA_APPL_ID	MQ_PROCESS_APPL_ID_LENGTH	Identificateur d'application	
MQCA_ENV_DATA	MQ_PROCESS_ENV_DATA_LENGTH	Données d'environnement	
MQCA_PROCESS_DESC	MQ_PROCESS_DESC_LENGTH	Description de la définition de processus	
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	Nom de la définition de processus	
MQCA_USER_DATA	MQ_PROCESS_USER_DATA_LENGTH	Données utilisateur	
MQIA_APPL_TYPE	MQLONG	Type d'application	
MQIA_QSG_DISP	MQLONG	Disposition de groupe de partage de files d'attente	z/OS

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Date de la modification la plus récente	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Heure de l'altération la plus récente	
MQCA_CHANNEL_AUTO_DEF_EXIT	MQ_EXIT_NAME_LENGTH	Nom d'exit de définition de canal automatique	
MQCA_CHINIT_SERVICE_PARM		Réservé à l'utilisation par IBM	
MQCA_CLUSTER_WORKLOAD_DATA	MQ_EXIT_DATA_LENGTH	Données transmises à l'exit de charge de travail du cluster	
MQCA_CLUSTER_WORKLOAD_EXIT	MQ_EXIT_NAME_LENGTH	Nom de l'exit de charge de travail de cluster	
MQCA_COMMAND_INPUT_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente d'entrée des commandes système	

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_DEAD_LETTER_Q_NAME	MQ_Q_NAME_LENGTH	Nom de la file d'attente de rebut	
MQCA_DEF_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	Nom de file d'attente de transmission par défaut	
MQCA_DNS_GROUP	MQ_DNS_GROUP_NAME_LENGTH	Nom du groupe du programme d'écoute TCP qui gère les transmissions entrantes du groupe de partage de files d'attente à joindre. Le nom s'applique lors de l'utilisation des services de nom de domaine dynamique de Workload Manager.	z/OS
MQCA_IGQ_USER_ID	MQ_USER_ID_LENGTH	Identificateur utilisateur de la mise en file d'attente intra-groupe	z/OS
MQCA_INSTALLATION_DESC	MQ_INSTALLATION_DESC_LENGTH	Description de l'installation associée	Pas z/OS. NON IBM i
MQCA_INSTALLATION_NAME	MQ_INSTALLATION_NAME_LENGTH	Nom de l'installation associée au gestionnaire de files d'attente	Pas z/OS. NON IBM i
MQCA_INSTALLATION_PATH	MQ_INSTALLATION_PATH_LENGTH	Chemin dans lequel leIBM WebSphere MQ associé est installé	Pas z/OS. NON IBM i
MQCA_LU_GROUP_NAME	MQ_LU_NAME_LENGTH	Nom d'unité logique générique du programme d'écoute d'unité logique 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente à utiliser	z/OS
MQCA_LU_NAME	MQ_LU_NAME_LENGTH	Nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes. Définissez ce nom sur la même unité logique que celle utilisée par le programme d'écoute pour les transmissions entrantes	z/OS
MQCA_LU62_ARM_SUFFIX	MQ_ARM_SUFFIX_LENGTH	Suffixe du SYS1.PARMLIB membre APPCPM*xx*, qui désigne la LUADD pour cet initiateur de canal	z/OS
MQCA_PARENT	MQ_Q_MGR_NAME_LENGTH	Nom d'un gestionnaire de files d'attente connecté de manière hiérarchique qui est désigné comme parent de ce gestionnaire de files d'attente	

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQCA_Q_MGR_DESC	MQ_Q_MGR_DESC_LENGTH	Description du gestionnaire de files d'attente	
MQCA_Q_MGR_IDENTIFIER	MQ_Q_MGR_IDENTIFIER_LENGT	Identificateur du gestionnaire de files d'attente (H)	
MQCA_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	Nom du gestionnaire de files d'attente local	
MQCA_QSG_NAME	MQ_QSG_NAME_LENGTH	Nom du groupe de partage de files d'attente	z/OS
MQCA_REPOSITORY_NAME	MQ_CLUSTER_NAME_LENGTH	Nom du cluster pour lequel le gestionnaire de files d'attente fournit des services de référentiel	
MQCA_REPOSITORY_NAMELIST	MQ_NAMELIST_NAME_LENGTH	Nom de l'objet liste de noms contenant les noms des clusters pour lesquels le gestionnaire de files d'attente fournit des services de référentiel	
MQCA_TCP_NAME	MQ_TCP_NAME_LENGTH	Nom du système TCP/IP que vous utilisez	z/OS
MQIA_ACCOUNTING_CONN_OVERRIDE	MQLONG	Remplacer les paramètres de comptabilité	NON z/OS
MQIA_ACCOUNTING_INTERVAL	MQLONG	Fréquence d'écriture des enregistrements comptables intermédiaires	NON z/OS
MQIA_ACCOUNTING_MQI	MQLONG	Contrôle la collecte des informations de comptabilité pour les données MQI	NON z/OS
MQIA_ACCOUNTING_Q	MQLONG	Contrôle la collecte des informations de comptabilité pour les files d'attente	NON z/OS
MQIA_ACTIVE_CHANNELS	MQLONG	Nombre maximal de canaux pouvant être actifs à tout moment	z/OS
MQIA_ADOPTNEWMCA_CHECK	MQLONG	Eléments vérifiés pour déterminer s'il convient d'adopter un agent MCA. La vérification est effectuée lorsqu'un nouveau canal entrant portant le même nom qu'un agent MCA déjà actif est détecté.	z/OS
MQIA_ADOPTNEWMCA_INTERVAL	MQLONG	Durée, en secondes, pendant laquelle le nouveau canal attend l'arrêt du canal orphelin	NON z/OS

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_ADOPTNEWMCA_TYPE	MQLONG	Indique s'il faut redémarrer automatiquement une instance orpheline d'un agent MCA d'un type de canal particulier lorsqu'une nouvelle demande de canal entrant correspondant aux paramètres AdoptNewMCACheck est détectée	z/OS
MQIA_AUTHORITY_EVENT	MQLONG	Attribut de contrôle des événements de droits d'accès	NON z/OS
MQIA_BRIDGE_EVENT	MQLONG	Attribut de contrôle pour les événements de pont IMS	z/OS
MQIA_CHANNEL_AUTO_DEF	MQLONG	Attribut de contrôle pour la définition de canal automatique	NON z/OS
MQIA_CHANNEL_AUTO_DEF_EVENT	MQLONG	Attribut de contrôle pour les événements de définition de canal automatique	NON z/OS
MQIA_CHANNEL_EVENT	MQLONG	Attribut de contrôle pour les événements de canal	
MQIA_CHINIT_ADAPTERS	MQLONG	Nombre de sous-tâches d'adaptateur à utiliser pour le traitement des appels IBM WebSphere MQ	z/OS
MQIA_CHINIT_DISPATCHERS	MQLONG	Nombre de répartiteurs à utiliser pour l'initiateur de canal	z/OS
MQIA_CHINIT_TRACE_AUTO_START	MQLONG	Indique si la trace de l'initialisateur de canal doit être démarrée automatiquement	z/OS
MQIA_CHINIT_TRACE_TABLE_SIZE	MQLONG	Taille de l'espace des données de trace (en Mo) de l'initiateur de canal	z/OS
MQIA_CLUSTER_WORKLOAD_LENGTH	MQLONG	Longueur de la charge de travail du cluster.	
MQIA_CLWL_MRU_CHANNELS	MQLONG	Nombre de canaux les plus récemment utilisés pour l'équilibrage de charge de cluster	
MQIA_CLWL_USEQ	MQLONG	Utilisation des files d'attente éloignées	
MQIA_CODED_CHAR_SET_ID	MQLONG	Identificateur de jeu de caractères codés	
MQIA_COMMAND_EVENT	MQLONG	Attribut de contrôle pour les événements de commande	
MQIA_COMMAND_LEVEL	MQLONG	Niveau de commande pris en charge par le gestionnaire de files d'attente	

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_CONFIGURATION_EVENT	MQLONG	Attribut de contrôle pour les événements de configuration	NON z/OS
MQIA_DEF_CLUSTER_XMIT_Q_TYPE	MQLONG	Type de file d'attente de transmission par défaut à utiliser pour les canaux émetteurs de cluster.	NON z/OS
MQIA_DIST_LISTS	MQLONG	Prise en charge de la liste de diffusion	NON z/OS
MQIA_DNS_WLM	MQLONG	Indique si le programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente s'enregistre avec Workload Manager for Dynamic Domain Name Services	z/OS
MQIA_EXPIRY_INTERVAL	MQLONG	Intervalle entre les analyses des messages arrivés à expiration	z/OS
MQIA_GROUP_UR	MQLONG	Attribut de contrôle indiquant si les unités de récupération GROUP sont activées pour ce gestionnaire de files d'attente. La disposition de l'unité de récupération GROUP n'est disponible que si le gestionnaire de files d'attente est membre d'un groupe de partage de files d'attente	z/OS
MQIA_IGQ_PUT_AUTHORITY	MQLONG	Droit d'insertion de la mise en file d'attente intragroupe	z/OS
MQIA_INHIBIT_EVENT	MQLONG	Attribut de contrôle pour les événements d'interdiction	NON z/OS
MQIA_INTRA_GROUP_QUEUING	MQLONG	Prise en charge de la mise en file d'attente intra-groupe	z/OS
MQIA_LISTENER_TIMER	MQLONG	Intervalle (en secondes) entre les tentatives de redémarrage du programme d'écoute par IBM WebSphere MQ en cas d'échec d'APPC ou de TCP/IP.	z/OS
MQIA_LOCAL_EVENT	MQLONG	Attribut de contrôle pour les événements locaux	NON z/OS
MQIA_LOGGER_EVENT	MQLONG	Attribut de contrôle pour les événements d'interdiction	NON z/OS
MQIA_LU62_CHANNELS	MQLONG	Nombre maximal de canaux pouvant être en cours ou de clients pouvant être connectés à l'aide du protocole de transmission LU 6.2	z/OS

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_MSG_MARK_BROWSE_INTERVAL	MQLONG	Intervalle de temps (en millisecondes) après lequel le gestionnaire de files d'attente peut supprimer automatiquement une marque des messages de navigation. **Avertissement :** Vous ne devez pas définir cette valeur en dessous de la valeur par défaut de 5000.	
MQIA_MAX_CHANNELS	MQLONG	Nombre maximal de canaux pouvant être en cours (y compris les canaux de connexion serveur avec des clients connectés)	z/OS
MQIA_MAX_HANDLES	MQLONG	Nombre maximal de descripteurs	
MQIA_MAX_MSG_LENGTH	MQLONG	Longueur maximale des messages	
MQIA_MAX_PRIORITY	MQLONG	Priorité maximum	
MQIA_MAX_UNCOMMITTED_MSGS	MQLONG	Nombre maximal de messages non validés dans une unité de travail	
MQIA_OUTBOUND_PORT_MAX	MQLONG	Avec MQIA_OUTBOUND_PORT_MIN, définit la plage de numéros de port à utiliser lors de la liaison des canaux sortants	z/OS
MQIA_OUTBOUND_PORT_MIN	MQLONG	Avec MQIA_OUTBOUND_PORT_MAX, définit la plage de numéros de port à utiliser lors de la liaison des canaux sortants	z/OS
MQIA_PERFORMANCE_EVENT	MQLONG	Attribut de contrôle pour les événements de performances	NON z/OS
MQIA_PLATFORM	MQLONG	Plateforme sur laquelle réside le gestionnaire de files d'attente	
MQIA_PROT_POLICY_CAPABILITY	MQLONG	Indique si les fonctions de sécurité de WebSphere MQ Advanced Message Security sont disponibles pour un gestionnaire de files d'attente.	
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	MQLONG	Nombre de tentatives de traitement d'un message de commande ayant échoué sous le point de synchronisation	

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_PUBSUB_MODE	MQLONG	Indique si le moteur de publication / abonnement et l'interface de publication / abonnement en file d'attente sont en cours d'exécution. Les applications de publication ou d'abonnement à l'aide de l'interface de programmation d'application requièrent le moteur de publication / abonnement. Les files d'attente surveillées par l'interface de publication / abonnement en file d'attente requièrent l'exécution de l'interface de publication / abonnement en file d'attente.	
MQIA_PUBSUB_NP_MSG	MQLONG	Indique s'il faut supprimer (ou conserver) un message d'entrée non distribué	
MQIA_PUBSUB_NP_RESP	MQLONG	Contrôle le comportement des messages de réponse non livrés.	
MQIA_PUBSUB_SYNC_PT	MQLONG	Indique si seuls les messages persistants (ou tous les messages) sont traités sous le point de synchronisation	
MQIA_QMGR_CFCONLOS	MQLONG	Indique l'action à effectuer lorsque le gestionnaire de files d'attente perd la connectivité à la structure d'administration ou à toute structure d'unité de couplage avec CFCONLOS défini sur ASQMGR	z/OS
MQIA_RECEIVE_TIMEOUT	MQLONG	Durée approximative pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. La valeur est numérique, qualifiée par MQIA_RECEIVE_TIMEOUT_TYPE.	z/OS
MQIA_RECEIVE_TIMEOUT_MIN	MQLONG	Durée minimale pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif	z/OS

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)

Sélecteur	Longueur de la zone	Description	Remarque
MQIA_RECEIVE_TIMEOUT_TYPE	MQLONG	Durée approximative pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. MQIA_RECEIVE_TIMEOUT_TYPE est le qualificateur appliqué à MQIA_RECEIVE_TIMEOUT.	z/OS
MQIA_REMOTE_EVENT	MQLONG	Attribut de contrôle pour les événements distants	NON z/OS
MQIA_SECURITY_CASE	MQLONG	Cas des profils de sécurité	z/OS
MQIA_SSL_EVENT	MQLONG	Attribut de contrôle pour les événements de canal	
MQIA_SSL_FIPS_REQUIRED	MQLONG	Utiliser uniquement des algorithmes certifiés FIPS pour la cryptographie	
MQIA_SSL_RESET_COUNT	MQLONG	Nbre avant réinit clé SSL	
MQIA_START_STOP_EVENT	MQLONG	Attribut de contrôle pour les événements d'arrêt de démarrage	NON z/OS
MQIA_STATISTICS_AUTO_CLUSSDR	MQLONG	Contrôle la collecte des informations de surveillance des statistiques pour les canaux émetteurs de cluster	NON z/OS
MQIA_STATISTICS_CHANNEL	MQLONG	Contrôle la collecte des données statistiques pour les canaux	NON z/OS
MQIA_STATISTICS_INTERVAL	MQLONG	Fréquence d'écriture des données de surveillance des statistiques	NON z/OS
MQIA_STATISTICS_MQI	MQLONG	Contrôle la collecte des informations de surveillance des statistiques pour le gestionnaire de files d'attente	NON z/OS
MQIA_STATISTICS_Q	MQLONG	Contrôle la collecte des données statistiques pour les files d'attente	NON z/OS
MQIA_SYNCPOINT	MQLONG	disponibilité des points de synchronisation	
MQIA_TCP_CHANNELS	MQLONG	Nombre maximal de canaux pouvant être en cours ou de clients pouvant être connectés à l'aide du protocole de transmission TCP/IP	z/OS
MQIA_TCP_KEEP_ALIVE	MQLONG	Indique s'il faut utiliser la fonction TCP KEEPALIVE pour vérifier que l'autre extrémité de la connexion est toujours disponible	z/OS

Tableau 570. Sélecteurs d'attribut MQINQ pour le gestionnaire de files d'attente (suite)			
Sélecteur	**Longueur de la zone**	**Description**	**Remarque**
MQIA_TCP_STACK_TYPE	MQLONG	Indique si l'initiateur de canal peut utiliser uniquement l'espace adresse TCP/IP spécifié dans TCPNAME ou s'il peut éventuellement se connecter à une adresse TCP/IP sélectionnée	z/OS
MQIA_TRACE_ROUTE_RECORDING	MQLONG	Contrôle l'enregistrement des informations de trace-route	z/OS
MQIA_TREE_LIFE_TIME	MQLONG	Durée de vie des rubriques non administratives inutilisées	
MQIA_TRIGGER_INTERVAL	MQLONG	Intervalle de déclenchement	

IntAttrCount
Type: MQLONG -entrée

Il s'agit du nombre d'éléments du tableau *IntAttrs* . Zéro est une valeur valide.

Si IntAttrCount est au moins le nombre de sélecteurs MQIA_* dans le paramètre *Selectors* , tous les attributs entiers demandés sont renvoyés.

IntAttrs
Type: MQLONG ×*IntAttrCount* -sortie

Il s'agit d'un tableau de valeurs d'attribut d'entier *IntAttrCount* .

Les valeurs d'attribut entières sont renvoyées dans le même ordre que les sélecteurs MQIA_* dans le paramètre *Selectors* . Si le tableau contient plus d'éléments que le nombre de sélecteurs MQIA_* , les éléments excédentaires restent inchangés.

Si *Hobj* représente une file d'attente, mais qu'un sélecteur d'attribut ne s'applique pas à ce type de file d'attente, la valeur spécifique MQIAV_NOT_APPLICABLE est renvoyée. Il est renvoyé pour l'élément correspondant dans le tableau *IntAttrs* .

Si le paramètre *IntAttrCount* ou *SelectorCount* a pour valeur zéro, il n'est pas fait référence à *IntAttrs* . Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en langage C ou en assembleur S/390 peut être null.

CharAttrLongueur
Type: MQLONG -entrée

Il s'agit de la longueur en octets du paramètre *CharAttrs* .

CharAttrLongueur doit être au moins la somme des longueurs des attributs de caractère demandés (voir *Selectors*). Zéro est une valeur valide.

CharAttrs
Type: MQCHAR × *CharAttrLongueur* -sortie

Il s'agit de la mémoire tampon dans laquelle les attributs de caractère sont renvoyés, concaténés. La longueur de la mémoire tampon est indiquée par le paramètre *CharAttrLength* .

Les attributs de caractère sont renvoyés dans le même ordre que les sélecteurs MQCA_* dans le paramètre *Selectors* . La longueur de chaque chaîne d'attribut est fixe pour chaque attribut (voir *Selectors*) et la valeur qu'il contient est complétée à droite par des blancs si nécessaire. Vous pouvez fournir une mémoire tampon plus grande que nécessaire pour contenir tous les attributs de caractères demandés et le remplissage. Les octets au-delà de la dernière valeur d'attribut renvoyée restent inchangés.

Si *Hobj* représente une file d'attente, mais qu'un sélecteur d'attribut ne s'applique pas à ce type de file d'attente, une chaîne de caractères composée entièrement d'astérisques (*) est renvoyée. L'astérisque est renvoyé en tant que valeur de cet attribut dans *CharAttrs*.

Si le paramètre *CharAttrLength* ou *SelectorCount* a pour valeur zéro, il n'est pas fait référence à *CharAttrs* . Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en langage C ou en assembleur S/390 peut être null.

CompCode
Type: MQLONG -sortie

Code achèvement:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucune raison de signaler.

Si *CompCode* est MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
(2008, X'7D8') Espace insuffisant pour les attributs de caractères.

MQRC_INT_ATTR_COUNT_TOO_SMALL
(2022, X'7E6') Espace insuffisant pour les attributs de type entier.

MQRC_SELECTOR_NOT_FOR_TYPE
(2068, X'814') Sélecteur non applicable au type de file d'attente.

Si *CompCode* est MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service de l'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') L'exit d'API a échoué.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit d'API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principal et principal sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CF_STRUC_FAILED
(2373, X'945') Echec de la structure de l'unité de couplage.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure d'unité de couplage en cours d'utilisation.

MQRC_CHAR_ATTR_LENGTH_ERROR
(2006, X'7D6') La longueur des attributs de caractère n'est pas valide.

MQRC_CHAR_ATTRS_ERROR
(2007, X'7D7') Chaîne d'attributs de caractères non valide.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') La connexion au gestionnaire de files d'attente a été perdue.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Arrêt de la connexion.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion incorrect.

MQRC_HOBJ_ERROR
(2019, X'7E3') Descripteur d'objet non valide.

MQRC_INT_ATTR_COUNT_ERROR
(2021, X'7E5') Nombre d'attributs de type entier non valide.

MQRC_INT_ATTRS_ARRAY_ERROR
(2023, X'7E7') Tableau d'attributs d'entier non valide.

MQRC_NOT_OPEN_FOR_INQUIRE
(2038, X'7F6') File d'attente non ouverte pour interrogation.

MQRC_OBJECT_CHANGED
(2041, X'7F9') La définition d'objet a été modifiée depuis son ouverture.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur lors de l'accès à l'ensemble de données de l'ensemble de pages.

MQRC_Q_DELETED
(2052, X'804') File d'attente supprimée.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Le nom du gestionnaire de files d'attente est incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Le gestionnaire de files d'attente n'est pas disponible pour la connexion.

MQRC_Q_MGR_STOPPING
(2162, X'872') Arrêt du gestionnaire de files d'attente.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SELECTOR_COUNT_ERROR
(2065, X'811') Nombre de sélecteurs non valide.

MQRC_SELECTOR_ERROR
(2067, X'813') Sélecteur d'attribut non valide.

MQRC_SELECTOR_LIMIT_EXCEEDED
(2066, X'812') Nombre de sélecteurs trop élevé.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus d'informations sur ces codes, voir Codes anomalie .

Notes d'utilisation

1. Les valeurs renvoyées sont un instantané des attributs sélectionnés. Il n'est pas garanti que les attributs restent les mêmes avant que l'application puisse agir sur les valeurs renvoyées.
2. Lorsque vous ouvrez une file d'attente modèle, une file d'attente locale dynamique est créée. Une file d'attente locale dynamique est créée même si vous ouvrez la file d'attente modèle pour en savoir plus sur ses attributs.

   Les attributs de la file d'attente dynamique sont en grande partie les mêmes que ceux de la file d'attente modèle lors de la création de la file d'attente dynamique. Si vous utilisez ensuite l'appel MQINQ sur cette file d'attente, le gestionnaire de files d'attente renvoie les attributs de la file d'attente dynamique et non ceux de la file d'attente modèle. Pour plus de détails sur les attributs de la file d'attente modèle qui sont hérités par la file d'attente dynamique, voir Tableau 573, à la page 826 .
3. Si l'objet demandé est une file d'attente alias, les valeurs d'attribut renvoyées par l'appel MQINQ sont les attributs de la file d'attente alias. Il ne s'agit pas des attributs de la file d'attente de base ou de la rubrique dans laquelle l'alias est résolu.
4. Si l'objet demandé est une file d'attente de cluster, les attributs pouvant être demandé dépendent de la manière dont la file d'attente est ouverte:

   - Vous pouvez ouvrir une file d'attente de cluster pour interroger plus une ou plusieurs opérations d'entrée, de navigation ou de définition. Pour ce faire, il doit exister une instance locale de la file d'attente de cluster pour que l'ouverture aboutisse. Dans ce cas, les attributs qui peuvent être renseignés sont les attributs qui sont valides pour les files d'attente locales.

     Si la file d'attente de cluster est ouverte pour interrogation sans entrée, exploration ou définition spécifiée, l'appel renvoie le code achèvement MQCC_WARNING et le code anomalie MQRC_SELECTOR_NOT_FOR_TYPE (2068) si vous tentez d'interroger des attributs qui sont valides uniquement pour les files d'attente locales et non pour les files d'attente de cluster.
   - Vous pouvez ouvrir une file d'attente de cluster pour l'interrogation lors de la transmission du nom de gestionnaire de files d'attente de base du gestionnaire de files d'attente connecté.

     Pour ce faire, il doit exister une instance locale de la file d'attente de cluster pour que l'ouverture aboutisse. Si le gestionnaire de files d'attente de base n'est pas transmis, l'appel renvoie le code achèvement MQCC_WARNING et le code anomalie MQRC_SELECTOR_NOT_FOR_TYPE (2068) si vous tentez d'interroger des attributs qui sont valides uniquement pour les files d'attente locales et non pour les files d'attente de cluster.
   - Si la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie, seuls les attributs répertoriés peuvent être utilisés pour l'interrogation. L'attribut **QType** a la valeur MQQT_CLUSTER dans ce cas:
     - MQCA_Q_DESC
     - MQCA_Q_NAME
     - MQIA_DEF_BIND
     - MQIA_DEF_PERSISTENCE
     - MQIA_DEF_PRIORITY
     - MQIA_INHIBIT_PUT
     - MQIA_Q_TYPE

     Vous pouvez ouvrir la file d'attente de cluster sans liaison fixe. Vous pouvez l'ouvrir avec MQOO_BIND_NOT_FIXED spécifié dans l'appel MQOPEN . Vous pouvez également spécifier MQOO_BIND_AS_Q_DEFet définir l'attribut **DefBind** de la file d'attente sur MQBND_BIND_NOT_FIXED. Si vous ouvrez une file d'attente de cluster sans liaison fixe, les appels MQINQ successifs de la file d'attente peuvent interroger des instances différentes de la file d'attente de cluster. Toutefois, il est typique que toutes les instances aient les mêmes valeurs d'attribut.
   - Un objet file d'attente alias peut être défini pour un cluster. Etant donné que TARGTYPE et TARGET ne sont pas des attributs de cluster, le processus exécutant un processus MQOPEN sur la file d'attente alias ne connaît pas l'objet dans lequel l'alias est résolu.

Lors de la MQOPENinitiale, la file d'attente alias se résout en un gestionnaire de files d'attente et en une file d'attente dans le cluster. La résolution de nom se produit à nouveau au niveau du gestionnaire de files d'attente éloignées et c'est ici que le TARGTPYE de la file d'attente alias est résolu.

Si la file d'attente alias se résout en alias de rubrique, la publication des messages insérés dans la file d'attente alias a lieu sur ce gestionnaire de files d'attente éloigné.

Voir Files d'attente de cluster

5. Vous pouvez demander un certain nombre d'attributs, puis en définir certains à l'aide de l'appel MQSET . Pour programmer une interrogation et une définition efficaces, positionnez les attributs à définir au début des tableaux de sélecteurs. Dans ce cas, les mêmes tableaux avec des effectifs réduits peuvent être utilisés pour MQSET.
6. Si plusieurs situations d'avertissement se produisent (voir le paramètre *CompCode*), le code anomalie renvoyé est le premier de la liste suivante qui s'applique:
   a. MQRC_SELECTOR_NOT_FOR_TYPE
   b. MQRC_INT_ATTR_COUNT_TOO_SMALL
   c. MQRC_CHAR_ATTRS_TOO_SHORT
7. La rubrique suivante contient des informations sur les attributs d'objet:
   - «Attributs des files d'attente», à la page 824
   - «Attributs des listes de noms», à la page 857
   - «Attributs des définitions de processus», à la page 859
   - «Attributs du gestionnaire de files d'attente», à la page 787

Appel C

```
MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
     CALL 'MQINQ' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,
                        INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,
                        CHARATTRS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
           IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
         CALL MQINQ,(HCONN,HOBJ,SELECTORCOUNT,SELECTORS,INTATTRCOUNT,  X
               INTATTRS,CHARATTRLENGTH,CHARATTRS,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Appel Visual Basic

```
MQINQ Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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-Propriété de message d'interrogation

L'appel MQINQMP renvoie la valeur d'une propriété d'un message.

Syntaxe

MQINQMP (*Hconn*, *Hmsg*, *InqPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *DataLength*, *CompCode*, *Reason*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre *Hmsg* .

Si le descripteur de message a été créé à l'aide de MQHC_UNASSOCIATED_HCONN, une connexion valide doit être établie sur l'unité d'exécution demandant une propriété du descripteur de message, sinon l'appel échoue avec MQRC_CONNECTION_BROKEN.

Msg
Type: MQHMSG-entrée

Il s'agit du descripteur de message à contrôler. La valeur a été renvoyée par un appel **MQCRTMH** précédent.

InqProp
Type: MQIMPO-entrée / sortie

Pour plus de détails, voir le type de données MQIMPO .

Nom
Type: MQCHARV-entrée / sortie

Nom de la propriété à interroger.

Si aucune propriété portant ce nom n'est trouvée, l'appel échoue avec le motif MQRC_PROPERTY_NOT_AVAILABLE.

Vous pouvez utiliser le caractère générique pourcentage (%) à la fin du nom de la propriété. Le caractère générique correspond à zéro ou plusieurs caractères, y compris le point (.). Cela permet à une application d'interroger la valeur de nombreuses propriétés. Appelez MQINQMP avec l'option MQIMPO_INQ_FIRST pour obtenir la première propriété correspondante et à nouveau avec l'option MQIMPO_INQ_NEXT pour obtenir la propriété correspondante suivante. Lorsqu'aucune propriété correspondante n'est disponible, l'appel échoue avec MQRC_PROPERTY_NOT_AVAILABLE. Si la zone *ReturnedName* de la structure InqPropOpts est initialisée avec une adresse ou un décalage pour le nom renvoyé de la propriété, cette opération est effectuée lors du retour de MQINQMP avec le nom de la propriété qui a été mise en correspondance. Si la zone *VSBufSize* de *ReturnedName* dans la structure InqPropOpts est inférieure à la longueur du nom de propriété renvoyé, le code achèvement est défini sur MQCC_FAILED avec le motif MQRC_PROPERTY_NAME_TOO_BIG.

Les propriétés qui possèdent des synonymes connus sont renvoyées comme suit:

1. Propriétés avec le préfixe "mqps." sont renvoyés en tant que nom de propriété WebSphere MQ . Par exemple, "MQTopicString" est le nom renvoyé et non "mqps.Top"

2. Propriétés avec le préfixe "jms." ou "mcd." sont renvoyés en tant que nom de zone d'en-tête JMS, par exemple, "JMSExpiration" est le nom renvoyé et non "jms.Exp".
3. Propriétés avec le préfixe "usr." sont renvoyés sans ce préfixe, par exemple, "Color" est renvoyé à la place de "usr.Color".

Les propriétés avec des synonymes ne sont renvoyées qu'une seule fois.

Dans le langage de programmation C, les variables de macro suivantes sont définies pour l'interrogation de toutes les propriétés, puis de toutes les propriétés qui commencent par "usr.":

MQPROP_INQUIRE_ALL
Consultez toutes les propriétés du message.

MQPROP_INQUIRE_ALL peut être utilisé de la manière suivante:

```
MQCHARV Name = {MQPROP_INQUIRE_ALL};
```

MQPROP_INQUIRE_ALL_USR
Consultez toutes les propriétés du message qui démarrent "usr.". Le nom renvoyé est renvoyé sans la valeur "usr." .

Si MQIMP_INQ_NEXT est spécifié mais que le nom a changé depuis l'appel précédent ou qu'il s'agit du premier appel, MQIMPO_INQ_FIRST est implicite.

Pour plus d'informations sur l'utilisation des noms de propriété, voir Noms de propriété et Restrictions relatives aux noms de propriété .

PropDesc
Type: MQPD-sortie

Cette structure est utilisée pour définir les attributs d'une propriété, y compris ce qui se produit si la propriété n'est pas prise en charge, le contexte de message auquel la propriété appartient et les messages dans lesquels la propriété doit être copiée. Pour plus d'informations sur cette structure, voir MQPD .

type
Type: MQLONG-entrée / sortie

En cas de retour de l'appel MQINQMP, ce paramètre est défini sur le type de données *Valeur*. Le type de données peut être l'un des suivants:

MQTYPE_BOOLÉEN
Une valeur booléenne.

MQTYPE_BYTE_STRING
une chaîne d'octets.

MQTYPE_INT8
Entier signé de 8 bits.

MQTYPE_INT16
Entier signé 16 bits.

MQTYPE_INT32
Entier signé 32 bits.

MQTYPE_INT64
Entier signé 64 bits.

MQTYPE_FLOAT32
Nombre à virgule flottante 32 bits.

MQTYPE_FLOAT64
Nombre en virgule flottante 64 bits.

MQTYPE_CHAINE
Chaîne de caractères.

MQTYPE_NULL
La propriété existe mais a une valeur nulle.

Si le type de données de la valeur de propriété n'est pas reconnu, MQTYPE_STRING est renvoyé et une représentation de chaîne de la valeur est placée dans la zone *Valeur* . Une représentation sous forme de chaîne du type de données est disponible dans la zone *TypeString* du paramètre *InqPropOpts* . Un code achèvement d'avertissement est renvoyé avec la raison MQRC_PROP_TYPE_NOT_SUPPORTED.

De plus, si l'option MQIMPO_CONVERT_TYPE est spécifiée, la conversion de la valeur de propriété est demandée. Utilisez *Type* comme entrée pour spécifier le type de données sous lequel vous souhaitez que la propriété soit renvoyée. Pour plus de détails sur la conversion de type de données, voir la description de l'option MQIMPO_CONVERT_TYPE de la structure MQIMPO .

Si vous ne demandez pas de conversion de type, vous pouvez utiliser la valeur suivante en entrée:

MQTYPE_AS_SET
La valeur de la propriété est renvoyée sans convertir son type de données.

ValueLength
Type : MQLONG - entrée

Longueur en octets de la zone Valeur. Spécifiez zéro pour les propriétés pour lesquelles vous n'avez pas besoin de la valeur renvoyée. Il peut s'agir de propriétés conçues par une application pour avoir une valeur nulle ou une chaîne vide. Spécifiez également zéro si l'option MQIMPO_QUERY_LENGTH a été spécifiée ; dans ce cas, aucune valeur n'est renvoyée.

valeur
Type: MQBYTEx*ValueLength* -sortie

Il s'agit de la zone devant contenir la valeur de la propriété interrogée. La mémoire tampon doit être alignée sur une limite appropriée pour la valeur renvoyée. Si vous ne le faites pas, une erreur peut se produire lors de l'accès ultérieur à la valeur.

Si *ValueLength* est inférieur à la longueur de la valeur de la propriété, la plus grande partie de la valeur de la propriété est déplacée dans *Valeur* et l'appel échoue avec le code achèvement MQCC_FAILED et la raison MQRC_PROPERTY_VALEUR_TOO_BIG.

Le jeu de caractères des données dans *Valeur* est fourni par la zone ReturnedCCSID dans le paramètre Opts InqProp. Le codage des données dans *Valeur* est donné par la zone ReturnedEncoding dans le paramètre Opts InqProp.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si le paramètre *ValueLength* est égal à zéro, *Value* n'est pas référencé et sa valeur transmise par les programmes écrits en assembleur C ou System/390 peut être null.

DataLength
Type : MQLONG - sortie

Il s'agit de la longueur en octets de la valeur de propriété réelle renvoyée dans la zone *Valeur* .

Si *DataLength* est inférieur à la longueur de la valeur de la propriété, *DataLength* est toujours renseigné en cas de retour de l'appel MQINQMP. Cela permet à l'application de déterminer la taille de la mémoire tampon requise pour la valeur de propriété, puis d'émettre à nouveau l'appel avec une mémoire tampon de la taille appropriée.

Les valeurs suivantes peuvent également être renvoyées.

Si le paramètre *Type* est défini sur MQTYPE_STRING ou MQTYPE_BYTE_STRING:

MQVL_EMPTY_CHAINE
La propriété existe mais ne contient pas de caractères ni d'octets.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_PROP_NOM_NON_CONVERTI
(2492, X'09BC') Nom de propriété renvoyé non converti.

MQRC_PROP_VALEUR_NOT_CONVERTED
(2466, X'09A2') Valeur de propriété non convertie.

MQRC_PROP_TYPE_NON_PRIS en charge
(2467, X'09A3') Le type de données de propriété n'est pas pris en charge.

MQRC_RFH_FORMAT_ERREUR
(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Carte non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'0852') Impossible de charger le module de service de l'adaptateur.

MQRC_ASID_MISMATCH
(2157, X'086D') Les ASID principal et principal diffèrent.

MQRC_BUFFER_ERROR
(2004, X'07D4') Paramètre de valeur non valide.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Paramètre de longueur de valeur non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

MQRC_DATA_LENGTH_ERROR
(2010, X'07DA') Paramètre de longueur de données non valide.

MQRC_IMPO_ERREUR
(2464, X'09A0') La structure des options de propriété de message d'interrogation n'est pas valide.

ERREUR MQRC_HMSG_ERROR
(2460, X'099C') Descripteur de message non valide.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Descripteur de message déjà utilisé.

MQRC_OPTIONS_ERROR
(2046, X'07F8') Options non valides ou non cohérentes.

MQRC_PD_ERREUR
(2482, X'09B2') Structure de descripteur de propriété non valide.

MQRC_PROP_CONV_NOT_SUPPORTED
(2470, X'09A6') La conversion du type de données réel au type de données demandé n'est pas prise en charge.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Nom de propriété non valide.

MQRC_PROPERTY_NAME_TOO_BIG
(2465, X'09A1') Nom de propriété trop grand pour la mémoire tampon de nom renvoyée.

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7) Propriété non disponible.

MQRC_PROPERTY_VALEUR_TOO_BIG
(2469, X'09A5') Valeur de propriété trop grande pour la zone Valeur.

MQRC_PROP_NUMBER_FORMAT_ERROR
(2472, X'09A8') Erreur de format numérique détectée dans les données de valeur.

MQRC_PROPERTY_TYPE_ERROR
(2473, X'09A9') Type de propriété demandé non valide.

MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'0871') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'0893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Appel C

```
MQINQMP (Hconn, Hmsg, &InqPropOpts, &Name, &PropDesc, &Type,
ValueLength, Value, &DataLength, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQINQMP' USING HCONN, HMSG, INQMSGOPTS, NAME, PROPDESC, TYPE,
VALUELENGTH, VALUE, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQINQMP (Hconn, Hmsg, InqPropOpts, Name, PropDesc, Type,
ValueLength, Value, DataLength, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQINQMP,(HCONN,HMSG,INQMSGOPTS,NAME,PROPDESC,TYPE,
VALUELENGTH,VALUE,DATALENGTH,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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-Conversion du descripteur de message en mémoire tampon

L'appel MQMHBUF convertit un descripteur de message en mémoire tampon et est l'inverse de l'appel MQBUFMH.

Syntaxe

MQMHBUF (*Hconn*, *Hmsg*, *MsgHBufOpts*, *Nom*, *MsgDesc*, *BufferLength*, *Buffer*, *DataLength*, *CompCode*, *Motif*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* doit correspondre au descripteur de connexion qui a été utilisé pour créer le descripteur de message spécifié dans le paramètre *Hmsg* .

Si le descripteur de message a été créé à l'aide de MQHC_UNASSOCIATED_HCONN, une connexion valide doit être établie sur l'unité d'exécution qui supprime le descripteur de message. Si aucune connexion valide n'est établie, l'appel échoue avec MQRC_CONNECTION_BROKEN.

Msg
Type: MQHMSG-entrée

Il s'agit du descripteur de message pour lequel une mémoire tampon est requise. La valeur a été renvoyée par un appel MQCRTMH précédent.

MsgHBufOptions
Type: MQMHBO-entrée

La structure MQMHBO permet aux applications de spécifier des options qui contrôlent la façon dont les mémoires tampon sont produites à partir des descripteurs de message.

Pour plus d'informations, voir «MQMHBO-Options de traitement des messages vers la mémoire tampon», à la page 454.

Nom
Type: MQCHARV-entrée

Nom de la ou des propriétés à placer dans la mémoire tampon.

Si aucune propriété correspondant au nom ne peut être trouvée, l'appel échoue avec MQRC_PROPERTY_NOT_AVAILABLE.

Vous pouvez utiliser un caractère générique pour placer plusieurs propriétés dans la mémoire tampon. Pour ce faire, utilisez le caractère générique'%'à la fin du nom de la propriété. Ce caractère générique correspond à zéro ou plusieurs caractères, y compris le caractère'.' (?).

Dans le langage de programmation C, les variables macro suivantes sont définies pour l'interrogation de toutes les propriétés et de toutes les propriétés qui commencent par'usr':

MQPROP_INQUIRE_ALL
Placer toutes les propriétés du message dans la mémoire tampon

MQPROP_INQUIRE_ALL_USR
Placez toutes les propriétés du message qui commencent par les caractères'usr.' dans la mémoire tampon.

Pour plus d'informations sur l'utilisation des noms de propriété, voir Noms de propriété et Restrictions relatives aux noms de propriété .

MsgDesc
Type : MQMD - entrée/sortie

La structure *MsgDesc* décrit le contenu de la zone tampon.

Dans la sortie, les zones *Encoding*, *CodedCharSetId* et *Format* sont définies pour décrire correctement le codage, l'identificateur de jeu de caractères et le format des données dans la zone tampon, tels qu'ils sont écrits par l'appel.

Les données de cette structure se trouvent dans le jeu de caractères et le codage de l'application.

BufferLength
Type : MQLONG - entrée

BufferLength est la longueur de la zone tampon, en octets.

Tampon
Type : MQBYTExBufferLength - sortie

Buffer définit la zone devant contenir les propriétés de message. Vous devez aligner la mémoire tampon sur une limite de 4 octets.

Si *BufferLength* est inférieur à la longueur requise pour stocker les propriétés dans *Buffer*, MQMHBUF échoue avec MQRC_PROPERTY_VALUE_TOO_BIG.

Le contenu de la mémoire tampon peut changer même si l'appel échoue.

DataLength
Type : MQLONG - sortie

DataLength est la longueur, en octets, des propriétés renvoyées dans la mémoire tampon. Si la valeur est zéro, aucune propriété ne correspond à la valeur indiquée dans *Name* et l'appel échoue avec le code anomalie MQRC_PROPERTY_NOT_AVAILABLE.

Si *BufferLength* est inférieur à la longueur requise pour stocker les propriétés dans la mémoire tampon, l'appel MQMHBUF échoue avec MQRC_PROPERTY_VALUE_TOO_BIG, mais une valeur est toujours entrée dans *DataLength*. Cela permet à l'application de déterminer la taille de la mémoire tampon requise pour les propriétés, puis d'émettre à nouveau l'appel avec le *BufferLength*requis.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Carte non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_MHBO_ERREUR
(2501, X'095C') Le descripteur de message vers la structure des options de mémoire tampon n'est pas valide.

MQRC_BUFFER_ERROR
(2004, X'07D4') Paramètre de mémoire tampon non valide.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Paramètre de longueur de mémoire tampon non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') La connexion au gestionnaire de files d'attente a été perdue.

MQRC_DATA_LENGTH_ERROR
(2010, X'07DA') Paramètre de longueur de données non valide.

ERREUR MQRC_HMSG_ERROR
(2460, X'099C') Descripteur de message non valide.

MQRC_MD_ERROR
(2026, X'07EA') Descripteur de message non valide.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Descripteur de message déjà utilisé.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Options non valides ou non cohérentes.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Le nom de propriété n'est pas valide.

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7') Propriété non disponible.

MQRC_PROPERTY_VALEUR_TOO_BIG
(2469, X'09A5') La valeur BufferLength est trop petite pour contenir les propriétés spécifiées.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Appel C

```
MQMHBUF (Hconn, Hmsg, &MsgHBufOpts, &Name, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Notes d'utilisation

MQMHBUF convertit un descripteur de message en mémoire tampon.

Vous pouvez l'utiliser avec un exit d'API MQGET pour accéder à certaines propriétés, à l'aide des API de propriété de message, puis les transmettre dans une mémoire tampon à une application conçue pour utiliser des en-têtes MQRFH2 plutôt que des descripteurs de message.

Cet appel est l'inverse de l'appel MQBUFMH, que vous pouvez utiliser pour analyser les propriétés de message d'une mémoire tampon dans un descripteur de message.

Appel COBOL

```
     CALL 'MQMHBUF' USING HCONN, HMSG, MSGHBUFOPTS, NAME, MSGDESC,
                          BUFFERLENGTH, BUFFER, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQMHBUF (Hconn, Hmsg, MsgHBufOpts, Name, MsgDesc, BufferLength, Buffer,
              DataLength, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQMHBUF,(HCONN,HMSG,MSGHBUFOPTS,NAME,MSGDESC,BUFFERLENGTH,
                BUFFER,DATALENGTH,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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-Objet ouvert

L'appel MQOPEN établit l'accès à un objet.

Les types d'objet suivants sont valides:

- File d'attente (y compris les listes de distribution)
- Liste de noms
- Définition de processus
- Gestionnaire de files d'attente

- Topic

Syntaxe

MQOPEN (*Hconn*, *ObjDesc*, *Options*, *Hobj*, *CompCode*, *Raison*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

ObjDesc
Type: MQOD-entrée/sortie

Il s'agit d'une structure qui identifie l'objet à ouvrir. Pour plus de détails, voir «MQOD-Descripteur d'objet», à la page 456 .

Si la zone *ObjectName* du paramètre *ObjDesc* correspond au nom d'une file d'attente modèle, une file d'attente locale dynamiqueest créé avec les attributs de la file d'attente modèle ; cela se produit quelles que soient les options que vous spécifiez dans le paramètre *Options* . Les opérations suivantes utilisant le *Hobj* renvoyé par l'appel MQOPEN sont effectuées sur la nouvelle file d'attente dynamique et non sur la file d'attente modèle. Cela est vrai même pour les appels MQINQ et MQSET. Le nom de la file d'attente modèle dans le paramètre *ObjDesc* est remplacé par le nom de la file d'attente dynamique créée. Le type de la file d'attente dynamique est déterminé par la valeur de l'attribut *DefinitionType* de la file d'attente modèle (voir «Attributs des files d'attente», à la page 824). Pour plus d'informations sur les options de fermeture applicables aux files d'attente dynamiques, voir la description de l'appel MQCLOSE.

OPTIONS
Type : MQLONG - entrée

Vous devez spécifier au moins l'une des options suivantes:

- MQOO_BROWSE
- MQOO_INPUT_ * (un seul de ces éléments)
- MQOO_INTERROGATION
- MQOO_SORTIE
- MQOO_SET
- MQOO_BIND_ * (un seul de ces éléments)

Consultez le tableau suivant pour plus de détails sur ces options ; d'autres options peuvent être spécifiées selon les besoins. Si plusieurs options sont requises, les valeurs peuvent être:

- Ajouté ensemble (ne pas ajouter la même constante plus d'une fois), ou
- combinées par le biais de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations par bit).

Les combinaisons qui ne sont pas valides sont notées ; toutes les autres combinaisons sont valides. Seules les options applicables au type d'objet spécifié par *ObjDesc* sont autorisées. Le tableau suivant présente les options MQOPEN valides pour les requêtes et les rubriques.

Option	Alias [1]	Local et modèle	Eloignée	Cluster non local	Liste de distribution	Topic
MQOO_INPUT_AS_Q_DEF	Oui	Oui	Non	Non	Non	Non
MQOO_INPUT_SHARED	Oui	Oui	Non	Non	Non	Non
MQOO_INPUT_EXCLUSIVE	Oui	Oui	Non	Non	Non	Non
MQOO_SORTIE	Oui	Oui	Oui	Oui	Oui	Oui
MQOO_BROWSE	Oui	Oui	Non	Non	Non	Non
MQOO_CO_OP	Oui	Oui	Non	Non	Non	Non
MQOO_INTERROGATION	Oui	Oui	[2]	Oui	Non	Non
MQOO_SET	Oui	Oui	[2]	Non	Non	Non
MQOO_BIND_ON_OPEN [3]	Oui	Oui	Oui	Oui	Oui	Non
MQOO_BIND_NOT_FIXED [3]	Oui	Oui	Oui	Oui	Oui	Non
MQOO_BIND_ON_GROUP [3]	Oui	Oui	Oui	Oui	Oui	Non
MQOO_BIND_AS_Q_DEF [3]	Oui	Oui	Oui	Oui	Oui	Non
MQOO_SAVE_ALL_CONTEXT	Oui	Oui	Non	Non	Non	Non
MQOO_PASS_IDENTITY_CONTEXT	Oui	Oui	Oui	Oui	Oui	[4]
MQOO_PASS_ALL_CONTEXT	Oui	Oui	Oui	Oui	Oui	Oui
MQOO_SET_IDENTITY_CONTEXT	Oui	Oui	Oui	Oui	Oui	[4]
MQOO_SET_ALL_CONTEXT	Oui	Oui	Oui	Oui	Oui	Oui
MQOO_NO_READ_AHEAD	Oui	Oui	Non	Non	Non	Non
MQOO_READ_AHEAD	Oui	Oui	Non	Non	Non	Non
MQOO_READ_AHEAD_AS_Q_DEF	Oui	Oui	Non	Non	Non	Non
MQOO_ALTERNATE_USER_AUTHORITY	Oui	Oui	Oui	Oui	Oui	Oui
MQOO_FAIL_IF_QUIESCING	Oui	Oui	Oui	Oui	Oui	Oui
MQOO_RESOLVE_LOCAL_Q	Oui	Oui	Oui	Oui	Non	Non
MQOO_RESOLVE_LOCAL_TOPIC	Non	Non	Non	Non	Non	Oui
MQOO_NO_MULTICAST	Non	Non	Non	Non	Non	Oui

Remarque :

1. La validité des options pour les alias dépend de la validité de l'option pour la file d'attente dans laquelle l'alias est résolu.
2. Cette option est valide uniquement pour la définition locale d'une file d'attente éloignée.
3. Cette option peut être spécifiée pour n'importe quel type de file d'attente, mais elle est ignorée si la file d'attente n'est pas une file d'attente de cluster. Toutefois, l'attribut de file d'attente *DefBind* remplace la file d'attente de base même si la file d'attente alias ne se trouve pas dans un cluster.
4. Ces attributs peuvent être utilisés avec une rubrique, mais affectent uniquement le contexte défini pour le message conservé, et non les zones de contexte envoyées à un abonné.

Options d'accès: Les options suivantes contrôlent le type d'opérations pouvant être effectuées sur l'objet:

MQOO_INPUT_AS_Q_DEF
Ouvrez la file d'attente pour obtenir les messages en utilisant la valeur par défaut définie par la file d'attente.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. Le type d'accès est partagé ou exclusif, en fonction de la valeur de l'attribut de file d'attente *DefInputOpenOption* ; voir «Attributs des files d'attente», à la page 824 pour plus de détails.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

MQOO_INPUT_SHARED
Ouvrez la file d'attente pour obtenir les messages avec accès partagé.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel peut aboutir si la file d'attente est actuellement ouverte par cette application ou par une autre application avec MQOO_INPUT_SHARED, mais échoue avec le code anomalie MQRC_OBJECT_IN_USE si la file d'attente est actuellement ouverte avec MQOO_INPUT_EXCLUSIVE.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

MQOO_INPUT_EXCLUSIVE
Ouvrez la file d'attente pour obtenir les messages avec un accès exclusif.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel échoue avec le code anomalie MQRC_OBJECT_IN_USE si la file d'attente est actuellement ouverte par cette application ou par une autre application pour l'entrée d'un type quelconque (MQOO_INPUT_SHARED ou MQOO_INPUT_EXCLUSIVE).

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

MQOO_SORTIE

Ouvrez une file d'attente pour insérer des messages, ou une rubrique ou une chaîne de rubrique pour publier des messages.

La file d'attente ou la rubrique est ouverte pour être utilisée avec les appels MQPUT suivants.

Un appel MQOPEN avec cette option peut aboutir même si l'attribut de file d'attente *InhibitPut* est défini sur MQQA_PUT_INHIBÉ (bien que les appels MQPUT suivants échouent alors que l'attribut est défini sur cette valeur).

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution et les rubriques.

Les remarques suivantes s'appliquent à ces options:

- Une seule de ces options peut être spécifiée.
- Un appel MQOPEN avec l'une de ces options peut aboutir même si l'attribut de file d'attente *InhibitGet* est défini sur MQQA_GET_INHIBÉ (bien que les appels MQGET suivants échouent alors que l'attribut est défini sur cette valeur).
- Si la file d'attente est définie comme n'étant pas partageable (c'est-à-dire que l'attribut de file d'attente *Shareability* a la valeur MQQA_NOT_SHAREABLE), les tentatives d'ouverture de la file d'attente pour un accès partagé sont traitées comme des tentatives d'ouverture de la file d'attente avec un accès exclusif.
- Si une file d'attente alias est ouverte avec l'une de ces options, le test d'utilisation exclusive (ou de l'utilisation exclusive d'une autre application) est effectué sur la file d'attente de base dans laquelle l'alias est résolu.
- Ces options ne sont pas valides si *ObjectQMgrName* est le nom d'un alias de gestionnaire de files d'attente ; cela est vrai même si la valeur de l'attribut *RemoteQMgrName* dans la définition locale d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

MQOO_BROWSE
Ouvrez la file d'attente pour parcourir les messages.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants avec l'une des options suivantes:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_BROWSE_MSG_UNDER_CURSOR

Ceci est autorisé même si la file d'attente est actuellement ouverte pour MQOO_INPUT_EXCLUSIVE. Un appel MQOPEN avec l'option MQOO_BROWSE établit un curseur de navigation et le positionne logiquement avant le premier message de la file d'attente ; voir Zone MQGMO-Options pour plus d'informations.

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente. Elle n'est pas non plus valide si *ObjectQMgrName* est le nom d'un alias de gestionnaire de files d'attente ; cela est vrai même si la valeur de l'attribut *RemoteQMgrName* dans la définition locale d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

MQOO_CO_OP
Ouvert en tant que membre coopérant de l'ensemble de poignées.

Cette option est valide uniquement avec l'option MQOO_BROWSE. S'il est spécifié sans MQOO_BROWSE, MQOPEN renvoie MQRC_OPTIONS_ERROR.

Le descripteur renvoyé est considéré comme étant membre d'un ensemble de descripteurs coopérant pour les appels MQGET suivants avec l'une des options suivantes:

- MQGMO_MARK_BROWSE_CO_OP
- MQGMO_UNMARKED_BROWSE_MSG
- MQGMO_UNMARK_BROWSE_CO_OP

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente.

MQOO_INTERROGATION
Ouvrez l'objet pour interroger les attributs.

La file d'attente, la liste de noms, la définition de processus ou le gestionnaire de files d'attente est ouvert pour être utilisé avec les appels MQINQ suivants.

Cette option est valide pour tous les types d'objet autres que les listes de distribution. Elle n'est pas valide si *ObjectQMgrName* est le nom d'un alias de gestionnaire de files d'attente ; ceci est vrai même si la valeur de l'attribut *RemoteQMgrName* dans la définition locale d'une file d'attente éloignée utilisée pour l'alias de gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

MQOO_SET
Ouvrez la file d'attente pour définir les attributs.

La file d'attente est ouverte pour être utilisée avec les appels MQSET suivants.

Cette option est valide pour tous les types de file d'attente autres que les listes de distribution. Elle n'est pas valide si *ObjectQMgrName* est le nom d'une définition locale d'une file d'attente éloignée ; cela est vrai même si la valeur de l'attribut *RemoteQMgrName* dans la définition locale d'une file d'attente éloignée utilisée pour l'attribution d'alias au gestionnaire de files d'attente est le nom du gestionnaire de files d'attente local.

Options de liaison: Les options suivantes s'appliquent lorsque l'objet ouvert est une file d'attente de cluster ; ces options contrôlent la liaison du descripteur de file d'attente à une instance de la file d'attente de cluster:

MQOO_BIND_ON_OPEN
Le gestionnaire de files d'attente local lie l'identificateur de file d'attente à une instance de la file d'attente de destination lorsque la file d'attente est ouverte. Par conséquent, tous les messages insérés à l'aide de cet identificateur sont envoyés à la même instance de la file d'attente de destination et par la même route.

Cette option est valide uniquement pour les files d'attente et n'affecte que les files d'attente de cluster. Si cette option est spécifiée pour une file d'attente qui n'est pas une file d'attente de cluster, elle est ignorée.

MQOO_BIND_NOT_FIXED
Cette opération arrête le gestionnaire de files d'attente local qui lie le descripteur de file d'attente à une instance de la file d'attente de destination. Par conséquent, les appels MQPUT successifs utilisant ce descripteur envoient les messages à *différentes* instances de la file d'attente de destination ou à la même instance mais par des routes différentes. Elle permet également à l'instance sélectionnée d'être modifiée ultérieurement par le gestionnaire de files d'attente local, par un gestionnaire de files d'attente éloignées ou par un agent MCA, en fonction des conditions du réseau.

Remarque : Les applications client et serveur qui doivent échanger une *série* de messages pour terminer une transaction ne doivent pas utiliser MQOO_BIND_NOT_FIXED (ou MQOO_BIND_AS_Q_DEF lorsque *DefBind* a la valeur MQBND_BIND_NOT_FIXED), car les messages successifs de la série peuvent être envoyés à différentes instances de l'application serveur.

Si MQOO_BROWSE ou l'une des options MQOO_INPUT_ * est spécifiée pour une file d'attente de cluster, le gestionnaire de files d'attente est forcé de sélectionner l'instance locale de la file d'attente de cluster. Par conséquent, la liaison du descripteur de file d'attente est fixe, même si MQOO_BIND_NOT_FIXED est spécifié.

Si MQOO_INQUIRE est spécifié avec MQOO_BIND_NOT_FIXED, les appels MQINQ successifs utilisant ce descripteur peuvent interroger différentes instances de la file d'attente de cluster, bien que toutes les instances aient généralement les mêmes valeurs d'attribut.

MQOO_BIND_NOT_FIXED est valide uniquement pour les files d'attente et affecte uniquement les files d'attente de cluster. Si cette option est spécifiée pour une file d'attente qui n'est pas une file d'attente de cluster, elle est ignorée.

MQOO_BIND_ON_GROUPE
Permet à une application de demander qu'un groupe de messages soit alloué à la même instance de destination.

Cette option est valide uniquement pour les files d'attente et n'affecte que les files d'attente de cluster. Si cette option est spécifiée pour une file d'attente qui n'est pas une file d'attente de cluster, elle est ignorée.

MQOO_BIND_AS_Q_DEF
Le gestionnaire de files d'attente local lie le descripteur de file d'attente de la manière définie par l'attribut de file d'attente *DefBind* . La valeur de cet attribut est MQBND_BIND_ON_OPEN, MQBND_BIND_NOT_FIXED ou MQBND_BIND_ON_GROUP.

MQOO_BIND_AS_Q_DEF est la valeur par défaut lorsque MQOO_BIND_ON_OPEN, MQOO_BIND_NOT_FIXED ou MQOO_BIND_ON_GROUP n'est pas spécifié.

MQOO_BIND_AS_Q_DEF aide à la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec l'une des deux autres options de liaison, mais comme sa valeur est égale à zéro, cette utilisation ne peut pas être détectée.

Options de contexte: Les options suivantes contrôlent le traitement du contexte de message:

MQOO_SAVE_ALL_CONTEXT
Les informations de contexte sont associées à cet identificateur de file d'attente. Ces informations sont définies à partir du contexte de tout message extrait à l'aide de cet identificateur. Pour plus

d'informations sur le contexte de message, voir Contexte de message et Contrôle des informations de contexte.

Ces informations de contexte peuvent être transmises à un message qui est ensuite inséré dans une file d'attente à l'aide des appels MQPUT ou MQPUT1 . Voir les options MQPMO_PASS_IDENTITY_CONTEXT et MQPMO_PASS_ALL_CONTEXT décrites dans «MQPMO- Options d'insertion de message», à la page 477.

Tant qu'un message n'a pas été extrait, le contexte ne peut pas être transmis à un message en cours d'insertion dans une file d'attente.

Les informations de contexte d'un message extrait à l'aide de l'une des options de navigation MQGMO_BROWSE_ * ne sont pas sauvegardées (bien que les zones de contexte du paramètre *MsgDesc* soient définies après une navigation).

Cette option est valide uniquement pour les files d'attente locales, d'alias et modèles ; elle n'est pas valide pour les files d'attente éloignées, les listes de distribution et les objets qui ne sont pas des files d'attente. L'une des options MQOO_INPUT_ * doit être spécifiée.

MQOO_PASS_IDENTITY_CONTEXT
Cela permet à l'option MQPMO_PASS_IDENTITY_CONTEXT d'être spécifiée dans le paramètre *PutMsgOpts* lorsqu'un message est inséré dans une file d'attente. Cette option fournit au message les informations de contexte d'identité d'une file d'attente d'entrée qui a été ouverte avec l'option MQOO_SAVE_ALL_CONTEXT. Pour plus d'informations sur le contexte de message, voir Contexte de message et Contrôle des informations de contexte.

L'option MQOO_OUTPUT doit être spécifiée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

MQOO_PASS_ALL_CONTEXT
Cela permet à l'option MQPMO_PASS_ALL_CONTEXT d'être spécifiée dans le paramètre *PutMsgOpts* lorsqu'un message est inséré dans une file d'attente ; cela donne au message les informations d'identité et de contexte d'origine à partir d'une file d'attente d'entrée qui a été ouverte avec l'option MQOO_SAVE_ALL_CONTEXT. Pour plus d'informations sur le contexte de message, voir Contexte de message et Contrôle des informations de contexte .

Cette option implique MQOO_PASS_IDENTITY_CONTEXT, qui n'a donc pas besoin d'être spécifié. L'option MQOO_OUTPUT doit être spécifiée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

MQOO_SET_IDENTITY_CONTEXT
Cela permet de spécifier l'option MQPMO_SET_IDENTITY_CONTEXT dans le paramètre *PutMsgOpts* lorsqu'un message est inséré dans une file d'attente ; cela fournit au message les informations de contexte d'identité contenues dans le paramètre *MsgDesc* spécifié dans l'appel MQPUT ou MQPUT1 . Pour plus d'informations sur le contexte de message, voir Contexte de message et Contrôle des informations de contexte .

Cette option implique MQOO_PASS_IDENTITY_CONTEXT, qui n'a donc pas besoin d'être spécifié. L'option MQOO_OUTPUT doit être spécifiée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

MQOO_SET_ALL_CONTEXT
Cela permet à l'option MQPMO_SET_ALL_CONTEXT d'être spécifiée dans le paramètre *PutMsgOpts* lorsqu'un message est inséré dans une file d'attente ; cela fournit au message les informations de contexte d'identité et d'origine contenues dans le paramètre *MsgDesc* spécifié dans l'appel MQPUT ou MQPUT1 . Pour plus d'informations sur le contexte de message, voir Contexte de message et Contrôle des informations de contexte .

Cette option implique les options suivantes, qui ne doivent donc pas être spécifiées:

- MQOO_PASS_IDENTITY_CONTEXT
- MQOO_PASS_ALL_CONTEXT

- MQOO_SET_IDENTITY_CONTEXT

L'option MQOO_OUTPUT doit être spécifiée.

Cette option est valide pour tous les types de file d'attente, y compris les listes de distribution.

Options de lecture anticipée:

Lorsque vous appelez MQOPEN avec MQOO_READ_AHEAD, le client WebSphere MQ n'active la lecture anticipée que si certaines conditions sont remplies. Ces conditions sont les suivantes :

- La version du client et la version du gestionnaire de files d'attente éloignées doivent être la version 7 ou une version ultérieure de WebSphere MQ.
- L'application client doit être compilée et liée dans les bibliothèques client WebSphere MQ MQI à unités d'exécution.
- Le canal client doit utiliser le protocole TCP/IP.
- Le paramètre SharingConversations (SHARECNV) du canal doit avoir une valeur différente de zéro dans la définition de canal serveur et dans la définition de canal client.

Les options suivantes contrôlent si les messages non persistants sont envoyés au client avant qu'une application ne les demande. Les remarques suivantes s'appliquent aux options de lecture anticipée:

- Une seule de ces options peut être spécifiée.
- Ces options sont valides uniquement pour les files d'attente locales, d'alias et modèles. Ils ne sont pas valides pour les files d'attente éloignées, les listes de distribution, les rubriques ou les gestionnaires de files d'attente.
- Ces options ne sont applicables que lorsque l'une des options MQOO_BROWSE, MQOO_INPUT_SHARED et MQOO_INPUT_EXCLUSIVE est également spécifiée, bien qu'il ne s'agisse pas d'une erreur de spécification de ces options avec MQOO_INQUIRE ou MQOO_SET.
- Si l'application n'est pas exécutée en tant que client IBM WebSphere MQ , ces options sont ignorées.

MQOO_NO_READ_AHEAD
Les messages non persistants ne sont pas envoyés au client avant qu'une application ne les demande.

MQOO_READ_AHEAD
Les messages non persistants sont envoyés au client avant qu'une application ne les demande.

MQOO_READ_AHEAD_AS_Q_DEF
Le comportement de lecture anticipée est déterminé par l'attribut de lecture anticipée par défaut de la file d'attente en cours d'ouverture. Il s'agit de la valeur par défaut.

Autres options: Les options suivantes contrôlent la vérification des autorisations, ce qui se produit lorsque le gestionnaire de files d'attente est mis au repos, s'il faut ou non résoudre le nom de la file d'attente locale et la multidiffusion:

MQOO_ALTERNATE_USER_AUTHORITY
La zone *AlternateUserId* du paramètre *ObjDesc* contient un identificateur utilisateur à utiliser pour valider cet appel MQOPEN. L'appel peut aboutir uniquement si ce *AlternateUserId* est autorisé à ouvrir l'objet avec les options d'accès spécifiées, que l'ID utilisateur sous lequel l'application est exécutée soit ou non autorisé à le faire. Cela ne s'applique pas aux options de contexte spécifiées, qui sont toujours vérifiées par rapport à l'identificateur utilisateur sous lequel l'application s'exécute.

Cette option est valide pour tous les types d'objet.

MQOO_FAIL_IF_QUIESCING
L'appel MQOPEN échoue si le gestionnaire de files d'attente est à l'état de mise au repos.

Sous z/OS, pour une application CICS ou IMS , cette option force également l'échec de l'appel MQOPEN si la connexion est à l'état de mise au repos.

Cette option est valide pour tous les types d'objet.

Pour plus d'informations sur les canaux client, voir Présentation des clients IBM WebSphere MQ MQI .

MQOO_RÉSOLVE_LOCAL_Q
Remplissez ResolvedQName dans la structure MQOD avec le nom de la file d'attente locale qui a été ouverte. De même, le nom ResolvedQMgrest renseigné avec le nom du gestionnaire de files d'attente local qui héberge la file d'attente locale. Si la structure MQOD est inférieure à la version 3, MQOO_RESOLVE_LOCAL_Q est ignoré et aucune erreur n'est renvoyée.

La file d'attente locale est toujours renvoyée lorsqu'une file d'attente locale, alias ou modèle est ouverte, mais ce n'est pas le cas lorsque, par exemple, une file d'attente éloignée ou une file d'attente de cluster non locale est ouverte sans l'option MQOO_RESOLVE_LOCAL_Q ; les noms ResolvedQName et ResolvedQMgrsont renseignés avec les noms RemoteQName et RemoteQMgrtrouvés dans la définition de file d'attente éloignée, ou de la même manière avec la file d'attente de cluster éloignée choisie.

Si vous spécifiez MQOO_RESOLVE_LOCAL_Q lors de l'ouverture, par exemple, d'une file d'attente éloignée, ResolvedQName est la file d'attente de transmission dans laquelle les messages sont insérés. Le nom ResolvedQMgrest renseigné avec le nom du gestionnaire de files d'attente local hébergeant la file d'attente de transmission.

Si vous êtes autorisé à parcourir, entrer ou sortir dans une file d'attente, vous disposez des droits requis pour spécifier cet indicateur dans l'appel MQOPEN. Aucun droit spécial n'est requis.

Cette option est valide uniquement pour les files d'attente et les gestionnaires de files d'attente.

MQOO_RESOLVE_LOCAL_TOPIC
Renseignez ResolvedQName dans la structure MQOD en indiquant le nom de la rubrique d'administration ouverte.

MQOO_NO_MULTIDIFFUSION
Les messages de publication ne sont pas envoyés à l'aide de la multidiffusion.

Cette option est valide uniquement avec l'option MQOO_OUTPUT. S'il est spécifié sans MQOO_OUTPUT, MQOPEN est renvoyé avec MQRC_OPTIONS_ERROR.

Cette option est valide uniquement pour une rubrique.

Hobj
Type: MQHOBJ-sortie

Ce descripteur représente l'accès qui a été établi à l'objet. Il doit être spécifié lors des appels IBM WebSphere MQ suivants qui fonctionnent sur l'objet. Elle cesse d'être valide lorsque l'appel MQCLOSE est émis ou lorsque l'unité de traitement qui définit la portée de l'identificateur s'arrête.

La portée du descripteur d'objet renvoyé est identique à celle du descripteur de connexion spécifié dans l'appel. Pour plus d'informations sur la portée du descripteur, voir MQCONN-Hconn parameter .

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_MULTIPLE_MOTIFS
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ALIAS_BASE_Q_TYPE_ERROR
(2001, X'7D1') Le type de la file d'attente de base de l'alias n'est pas valide.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CF_NOT_XX_ENCODE_CASE_ONE disponible
(2345, X' 929') Unité de couplage non disponible.

Echec de MQRC_CF_STRUC_AUTH_FAILED
(2348, X'92C') La vérification d'autorisation de la structure d'unité de couplage a échoué.

MQRC_CF_STRUC_ERREUR
(2349, X'92D') Structure d'unité de couplage non valide.

MQRC_CF_STRUC_FAILED
(2373, X'945') Echec de la structure de l'unité de couplage.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347, X'92B') En-tête de liste de la structure d'unité de couplage en cours d'utilisation.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CLUSTER_EXIT_ERREUR
(2266, X'8DA') L'exit de charge de travail de cluster a échoué.

MQRC_CLUSTER_PUT_INHIBÉ
(2268, X'8DC') Appels d'insertion bloqués pour toutes les files d'attente du cluster.

Erreur de résolution MQRC_CLUSTER_RESOLUTION_ERROR
(2189, X'88D') La résolution du nom de cluster a échoué.

MQRC_CLUSTER_RESOURCE_ERROR
(2269, X'8DD') Erreur de ressource de cluster.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Connexion en cours de mise au repos.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926') Sous-système Db2 non disponible.

MQRC_DEF_XMIT_Q_TYPE_ERREUR
(2198, X'896') La file d'attente de transmission par défaut n'est pas locale.

MQRC_DEF_XMIT_Q_USAGE_ERROR
(2199, X'897') Erreur d'utilisation de la file d'attente de transmission par défaut.

MQRC_DYNAMIC_Q_NAME_ERROR
(2011, X'7DB') Nom de la file d'attente dynamique non valide.

MQRC_HANDLE_NOT_AVAILABLE
(2017, X'7E1') Plus de descripteurs disponibles.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HOBJ_ERROR
(2019, X'7E3') Descripteur d'objet non valide.

MQRC_MULTIPLE_MOTIFS
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

MQRC_NAME_IN_USE
(2201, X'899') Nom utilisé.

MQRC_NAME_NOT_VALID_FOR_TYPE
(2194, X'892') Nom d'objet incorrect pour le type d'objet.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_OBJECT_ALREADY_EXISTS
(2100, X'834') L'objet existe.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

MQRC_OBJECT_IN_USE
(2042, X'7FA') Objet déjà ouvert avec des options en conflit.

MQRC_OBJET_NIVEAU_INCOMPATIBLE
(2360, X' 938') Niveau d'objet non compatible.

MQRC_OBJECT_NAME_ERROR
(2152, X'868') Nom d'objet incorrect.

MQRC_OBJECT_NOT_UNIQUE
(2343, X' 927') Objet non unique.

MQRC_OBJECT_Q_MGR_NAME_ERROR
(2153, X'869') Nom de gestionnaire de files d'attente d'objet incorrect.

MQRC_OBJECT_RECORDS_ERROR
(2155, X'86B') Enregistrements d'objet non valides.

MQRC_OBJECT_STRING_ERROR
(2441, X'0989') Zone Objectstring non valide

MQRC_OBJECT_TYPE_ERREUR
(2043, X'7FB') Type d'objet non valide.

MQRC_ERREUR D'OD_DU
(2044, X'7FC') Structure de descripteur d'objet non valide.

MQRC_OPTION_NOT_VALID_FOR_TYPE
(2045, X'7FD') Option non valide pour le type d'objet.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_PAGESET_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_Q_DELETED
(2052, X'804') File d'attente supprimée.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_QUIESCING
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_Q_TYPE_ERREUR
(2057, X'809') Type de file d'attente non valide.

MQRC_RECS_Présentateur-ERREUR
(2154, X'86A') Nombre d'enregistrements présents non valide.

MQRC_REMOTE_Q_NAME_ERROR
(2184, X'888') Nom de file d'attente éloignée incorrect.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Enregistrements de réponse non valides.

MQRC_SECURITY_ERROR (ERREUR DE SECURITE MQ)
(2063, X'80F') Une erreur de sécurité s'est produite.

MQRC_SELECTOR_SYNTAX_ERREUR
2459 (X'099B') Un appel MQOPEN, MQPUT1 ou MQSUB a été émis mais une chaîne de sélection contenant une erreur de syntaxe a été spécifiée.

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Appel rejeté par l'exit de charge de travail du cluster.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082, X'822') File d'attente de base alias inconnue.

MQRC_UNKNOWN_DEF_XMIT_Q
(2197, X'895') File d'attente de transmission par défaut inconnue.

MQRC_UNKNOWN_NOM_OBJET
(2085, X'825') Nom d'objet inconnu.

MQRC_UNKNOWN_OBJET_Q_MGR
(2086, X'826') Gestionnaire de files d'attente d'objets inconnu.

MQRC_UNKNOWN_REMOTE_Q_MGR
(2087, X'827') Gestionnaire de files d'attente éloignées inconnu.

MQRC_UNKNOWN_XMIT_Q
(2196, X'894') File d'attente de transmission inconnue.

MQRC_WRONG_CF_NIVEAU
(2366, X'93E') La structure de l'unité de couplage est de niveau incorrect.

MQRC_XMIT_Q_TYPE_ERROR
(2091, X'82B') File d'attente de transmission non locale.

MQRC_XMIT_Q_USAGE_ERROR
(2092, X'82C') File d'attente de transmission avec une utilisation incorrecte.

Pour plus d'informations sur ces codes, voir:

- Codes anomalie pour toutes les autres plateformes IBM WebSphere MQ à l'exception de z/OS.

Remarques générales sur l'utilisation

1. L'objet ouvert est l'un des suivants:
   - Une file d'attente pour:
     - Extraire ou parcourir les messages (à l'aide de l'appel MQGET)
     - Messages d'insertion (à l'aide de l'appel MQPUT)
     - Renseignez-vous sur les attributs de la file d'attente (à l'aide de l'appel MQINQ)
     - Définir les attributs de la file d'attente (à l'aide de l'appel MQSET)

     Si la file d'attente nommée est une file d'attente modèle, une file d'attente locale dynamique est créée. Voir le paramètre *ObjDesc* décrit dans «MQOPEN-Objet ouvert», à la page 719.

     Une liste de distribution est un type spécial d'objet file d'attente qui contient une liste de files d'attente. Il peut être ouvert pour insérer des messages, mais pas pour extraire ou parcourir des messages, ni pour interroger ou définir des attributs. Voir la note d'utilisation 8 pour plus de détails.

     Une file d'attente `QSGDISP(GROUP)` est un type spécial de définition de file d'attente qui ne peut pas être utilisé avec les appels MQOPEN ou MQPUT1 .
   - Liste de noms permettant de connaître les noms des files d'attente de la liste (à l'aide de l'appel MQINQ).
   - Une définition de processus pour l'interrogation des attributs de processus (à l'aide de l'appel MQINQ).
   - Le gestionnaire de files d'attente pour demander les attributs du gestionnaire de files d'attente local (à l'aide de l'appel MQINQ).
   - Rubrique permettant de publier un message (à l'aide de l'appel MQPUT)
2. Une application peut ouvrir le même objet plusieurs fois. Un descripteur d'objet différent est renvoyé pour chaque ouverture. Chaque descripteur renvoyé peut être utilisé pour les fonctions pour lesquelles l'ouverture correspondante a été effectuée.
3. Si l'objet ouvert est une file d'attente autre qu'une file d'attente de cluster, la résolution de tous les noms dans le gestionnaire de files d'attente local est effectuée au moment de l'appel MQOPEN. Cet audit peut porter sur l'un ou plusieurs des éléments suivants :
   - Résolution du nom d'une définition locale d'une file d'attente éloignée en nom du gestionnaire de files d'attente éloignées et nom sous lequel la file d'attente est connue sur le gestionnaire de files d'attente éloignées
   - Résolution du nom du gestionnaire de files d'attente éloignées en nom de file d'attente de transmission locale
   - (z/OS uniquement) Résolution du nom du gestionnaire de files d'attente éloignées par le nom de la file d'attente de transmission partagée utilisée par l'agent de mise en file d'attente intra-groupe (s'applique uniquement si les gestionnaires de files d'attente locales et éloignées appartiennent au même groupe de partage de files d'attente)
   - Résolution d'alias sur le nom d'une file d'attente de base ou d'un objet de rubrique.

Cependant, sachez que les appels MQINQ ou MQSET ultérieurs pour le descripteur se rapportent uniquement au nom qui a été ouvert et non à l'objet résultant de la résolution de nom. Par exemple, si l'objet ouvert est un alias, les attributs renvoyés par l'appel MQINQ sont les attributs de l'alias, et non les attributs de la file d'attente de base ou un objet de rubrique auquel l'alias est résolu.

Si l'objet en cours d'ouverture est une file d'attente de cluster, la résolution de nom peut se produire au moment de l'appel MQOPEN ou être différée jusqu'à une date ultérieure. Le point de résolution est contrôlé par les options MQOO_BIND_ * spécifiées dans l'appel MQOPEN:

- MQOO_BIND_ON_OPEN
- MQOO_BIND_NOT_FIXED
- MQOO_BIND_AS_Q_DEF
- MQOO_BIND_ON_GROUPE

Voir Résolution de nom pour plus d'informations sur la résolution de nom pour les files d'attente de cluster.

4. Un appel MQOPEN avec l'option MQOO_BROWSE établit un curseur de navigation, à utiliser avec les appels MQGET qui spécifient le descripteur d'objet et l'une des options de navigation. Cela permet d'analyser la file d'attente sans modifier son contenu. Un message trouvé lors de la consultation peut être supprimé de la file d'attente à l'aide de l'option MQGMO_MSG_UNDER_CURSOR.

   Plusieurs curseurs de navigation peuvent être actifs pour une seule application en émettant plusieurs demandes MQOPEN pour la même file d'attente.
5. Les applications démarrées par un moniteur de déclenchement sont transmises avec le nom de la file d'attente associée à l'application lors de son démarrage. Ce nom de file d'attente peut être spécifié dans le paramètre *ObjDesc* pour ouvrir la file d'attente. Pour plus de détails, voir «MQTMC2 -Message de déclenchement 2 (format de caractère)», à la page 588.
6. Sous IBM i, les applications qui s'exécutent en mode compatibilité sont connectées automatiquement au gestionnaire de files d'attente par le premier appel MQOPEN émis par l'application (si l'application ne s'est pas déjà connectée au gestionnaire de files d'attente à l'aide de l'appel MQCONN).

   Les applications qui ne s'exécutent pas en mode compatibilité doivent émettre l'appel MQCONN ou MQCONNX pour se connecter au gestionnaire de files d'attente de manière explicite, avant d'utiliser l'appel MQOPEN pour ouvrir un objet.

Options de lecture anticipée

Lorsque vous appelez MQOPEN avec MQOO_READ_AHEAD, le client WebSphere MQ n'active la lecture anticipée que si certaines conditions sont remplies. Ces conditions sont les suivantes :

- La version du client et la version du gestionnaire de files d'attente éloignées doivent être la version 7 ou une version ultérieure de WebSphere MQ.
- L'application client doit être compilée et liée dans les bibliothèques client WebSphere MQ MQI à unités d'exécution.
- Le canal client doit utiliser le protocole TCP/IP.
- Le paramètre SharingConversations (SHARECNV) du canal doit avoir une valeur différente de zéro dans la définition de canal serveur et dans la définition de canal client.

Les remarques suivantes s'appliquent à l'utilisation des options de lecture anticipée.

1. Les options de lecture anticipée s'appliquent uniquement lorsqu'une seule des options MQOO_BROWSE, MQOO_INPUT_SHARED et MQOO_INPUT_EXCLUSIVE est également spécifiée. Une erreur n'est pas générée si des options de lecture anticipée sont spécifiées avec les options MQOO_INQUIRE ou MQOO_SET.
2. La lecture anticipée n'est pas activée lorsqu'elle est demandée si les options utilisées sur le premier appel MQGET ne sont pas prises en charge pour une utilisation avec la lecture anticipée. En outre, la lecture anticipée est désactivée lorsque le client se connecte à un gestionnaire de files d'attente qui ne prend pas en charge la lecture anticipée.

3. Si l'application n'est pas exécutée en tant que client IBM WebSphere MQ , les options de lecture anticipée sont ignorées.

Files d'attente de cluster

Les remarques suivantes s'appliquent à l'utilisation des files d'attente de cluster.

1. Lorsqu'une file d'attente de cluster est ouverte pour la première fois et que le gestionnaire de files d'attente local n'est pas un gestionnaire de files d'attente de référentiel complet, le gestionnaire de files d'attente local obtient des informations sur la file d'attente de cluster à partir d'un gestionnaire de files d'attente de référentiel complet. Lorsque le réseau est occupé, le gestionnaire de files d'attente local peut mettre plusieurs secondes à recevoir les informations requises du gestionnaire de files d'attente de référentiel. Par conséquent, l'application qui émet l'appel MQOPEN peut avoir à attendre jusqu'à 10 secondes avant que le contrôle ne soit renvoyé à partir de l'appel MQOPEN. Si le gestionnaire de files d'attente local ne reçoit pas les informations nécessaires sur la file d'attente de cluster dans ce délai, l'appel échoue avec le code anomalie MQRC_CLUSTER_RESOLUTION_ERROR.
2. Lorsqu'une file d'attente de cluster est ouverte et qu'il existe plusieurs instances de la file d'attente dans le cluster, l'instance ouverte dépend des options spécifiées dans l'appel MQOPEN:
   - Si les options spécifiées incluent l'une des options suivantes:
     - MQOO_BROWSE
     - MQOO_INPUT_AS_Q_DEF
     - MQOO_INPUT_EXCLUSIVE
     - MQOO_INPUT_SHARED
     - MQOO_SET

     l'instance de la file d'attente de cluster ouverte doit être l'instance locale. S'il n'existe aucune instance locale de la file d'attente, l'appel MQOPEN échoue.
   - Si les options spécifiées n'incluent aucune des options décrites précédemment, mais incluent l'une des options suivantes ou les deux:
     - MQOO_INTERROGATION
     - MQOO_SORTIE

     l'instance ouverte est l'instance locale, s'il en existe une, et une instance distante dans le cas contraire (si vous utilisez les valeurs par défaut de CLWLUSEQ). L'instance choisie par le gestionnaire de files d'attente peut toutefois être modifiée par un exit de charge de travail de cluster (s'il en existe un).
3. S'il existe un abonnement pour la file d'attente, mais qu'elle n'est pas reconnue par un référentiel complet, l'objet n'est pas présent dans le cluster et l'appel échoue avec le code anomalie MQRC_OBJECT_NAME.

Pour plus d'informations sur les files d'attente de cluster, voir Files d'attente de cluster.

Listes de diffusion

Les remarques suivantes s'appliquent à l'utilisation des listes de distribution.

Les listes de distribution sont prises en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients IBM WebSphere MQ MQI connectés à ces systèmes.

1. Les zones de la structure MQOD doivent être définies comme suit lors de l'ouverture d'une liste de distribution:
   - *Version* doit être MQOD_VERSION_2 ou supérieur.
   - *ObjectType* doit être MQOT_Q.
   - *ObjectName* doit être vide ou la chaîne null.
   - *ObjectQMgrName* doit être vide ou la chaîne null.

- *RecsPresent* être supérieur à zéro.
- L'un des éléments *ObjectRecOffset* et *ObjectRecPtr* doit être égal à zéro et l'autre doit être différent de zéro.
- *ResponseRecOffset* et *ResponseRecPtr* ne peuvent pas être différents de zéro.
- Il doit y avoir des enregistrements d'objet *RecsPresent* , traités par *ObjectRecOffset* ou *ObjectRecPtr*. Les enregistrements d'objet doivent être définis sur les noms des files d'attente de destination à ouvrir.
- Si l'un des éléments *ResponseRecOffset* et *ResponseRecPtr* est différent de zéro, des enregistrements de réponse *RecsPresent* doivent être présents. Ils sont définis par le gestionnaire de files d'attente si l'appel se termine avec le code anomalie MQRC_MULTIPLE_MOTIFS.

Un MQOD version-2 peut également être utilisé pour ouvrir une file d'attente unique qui ne figure pas dans une liste de distribution, en veillant à ce que *RecsPresent* soit égal à zéro.

2. Seules les options d'ouverture suivantes sont valides dans le paramètre *Options* :
   - MQOO_SORTIE
   - MQOO_PASS_ * _CONTEXTE
   - MQOO_SET_ * _CONTEXTE
   - MQOO_ALTERNATE_USER_AUTHORITY
   - MQOO_FAIL_IF_QUIESCING
3. Les files d'attente de destination de la liste de distribution peuvent être des files d'attente locales, des files d'attente alias ou des files d'attente éloignées, mais elles ne peuvent pas être des files d'attente modèles. Si une file d'attente modèle est spécifiée, l'ouverture de cette file d'attente échoue avec le code anomalie MQRC_Q_TYPE_ERROR. Toutefois, cela n'empêche pas l'ouverture d'autres files d'attente de la liste.
4. Les paramètres de code achèvement et de code anomalie sont définis comme suit:
   - Si les opérations d'ouverture des files d'attente de la liste de distribution aboutissent ou échouent de la même manière, les paramètres de code achèvement et de code anomalie sont définis pour décrire le résultat commun. Les enregistrements de réponse MQRR (s'ils sont fournis par l'application) ne sont pas définis dans ce cas.

     Par exemple, si chaque ouverture aboutit, le code achèvement est défini sur MQCC_OK et le code anomalie est défini sur MQRC_NON; si chaque ouverture échoue car aucune des files d'attente n'existe, les paramètres sont définis sur MQCC_FAILED et MQRC_UNKNOWN_OBJECT_NAME.
   - Si les opérations d'ouverture des files d'attente de la liste de distribution n'aboutissent pas toutes ou échouent de la même manière:
     - Le paramètre de code achèvement est défini sur MQCC_WARNING si au moins une ouverture a abouti et sur MQCC_FAILED si toutes les opérations ont échoué.
     - Le paramètre de code anomalie est défini sur MQRC_MULTIPLE_MOTIFS.
     - Les enregistrements de réponse (s'ils sont fournis par l'application) sont définis sur les codes achèvement individuels et les codes raison des files d'attente de la liste de distribution.
5. Lorsqu'une liste de distribution a été ouverte avec succès, le descripteur *Hobj* renvoyé par l'appel peut être utilisé sur les appels MQPUT suivants pour placer des messages dans les files d'attente de la liste de distribution et sur un appel MQCLOSE pour abandonner l'accès à la liste de distribution. La seule option de fermeture valide pour une liste de distribution est MQCO_NONE.

   L'appel MQPUT1 peut également être utilisé pour placer un message dans une liste de distribution ; la structure MQOD définissant les files d'attente dans la liste est spécifiée en tant que paramètre sur cet appel.
6. Chaque destination correctement ouverte dans la liste de distribution compte comme un descripteur distinct lors de la vérification du dépassement du nombre maximal de descripteurs autorisé par l'application (voir l'attribut de gestionnaire de files d'attente *MaxHandles*). Cela est vrai même lorsque deux ou plusieurs des destinations de la liste de distribution sont résolues dans la même

file d'attente physique. Si l'appel MQOPEN ou MQPUT1 pour une liste de distribution entraîne le dépassement du nombre de descripteurs utilisés par l'application *MaxHandles*, l'appel échoue avec le code anomalie MQRC_HANDLE_NOT_AVAILABLE.

7. Chaque destination ouverte avec succès a la valeur de son attribut *OpenOutputCount* incrémentée d'un. Si au moins deux des destinations de la liste de distribution sont résolues dans la même file d'attente physique, l'attribut *OpenOutputCount* de cette file d'attente est incrémenté du nombre de destinations de la liste de distribution qui sont résolues dans cette file d'attente.
8. Toute modification apportée aux définitions de file d'attente qui aurait entraîné la non-validité d'un descripteur si les files d'attente avaient été ouvertes individuellement (par exemple, une modification du chemin de résolution) n'entraîne pas la non-validité du descripteur de liste de distribution. Toutefois, elle entraîne un échec pour cette file d'attente particulière lorsque le descripteur de liste de distribution est utilisé lors d'un appel MQPUT ultérieur.
9. Une liste de distribution ne peut contenir qu'une seule destination.

Files d'attente éloignées

Les remarques suivantes s'appliquent à l'utilisation des files d'attente éloignées.

Une file d'attente éloignée peut être spécifiée de deux manières dans le paramètre *ObjDesc* de cet appel.

- En spécifiant pour *ObjectName* le nom d'une définition locale de la file d'attente éloignée. Dans ce cas, *ObjectQMgrName* fait référence au gestionnaire de files d'attente local et peut être spécifié sous forme de blancs ou (dans le langage de programmation C) sous la forme d'une chaîne nulle.

  La validation de sécurité effectuée par le gestionnaire de files d'attente local vérifie que l'utilisateur est autorisé à ouvrir la définition locale de la file d'attente éloignée.
- En spécifiant pour *ObjectName* le nom de la file d'attente éloignée, tel qu'il est connu du gestionnaire de files d'attente éloignées. Dans ce cas, *ObjectQMgrName* est le nom du gestionnaire de files d'attente éloignées.

  La validation de sécurité effectuée par le gestionnaire de files d'attente local vérifie que l'utilisateur est autorisé à envoyer des messages à la file d'attente de transmission résultant du processus de résolution de nom.

Dans les deux cas:

- Aucun message n'est envoyé par le gestionnaire de files d'attente local au gestionnaire de files d'attente éloignées pour vérifier que l'utilisateur est autorisé à placer des messages dans la file d'attente.
- Lorsqu'un message arrive sur le gestionnaire de files d'attente éloignées, ce dernier peut le rejeter car l'utilisateur à l'origine du message n'est pas autorisé.

Pour plus d'informations, voir les zones *ObjectName* et *ObjectQMgrName* décrites dans «MQOD-Descripteur d'objet», à la page 456 .

Objets

Sécurité

Les remarques suivantes concernent les aspects de sécurité liés à l'utilisation de MQOPEN.

Le gestionnaire de files d'attente effectue des contrôles de sécurité lorsqu'un appel MQOPEN est émis, afin de vérifier que l'ID utilisateur sous lequel l'application s'exécute dispose du niveau de droits approprié avant que l'accès ne soit autorisé. La vérification des droits est effectuée sur le nom de l'objet en cours d'ouverture et non sur le ou les noms, ce qui se produit après la résolution d'un nom.

Si l'objet ouvert est une file d'attente alias qui pointe vers un objet de rubrique, le gestionnaire de files d'attente effectue un contrôle de sécurité sur le nom de la file d'attente alias, avant d'effectuer un contrôle de sécurité pour la rubrique comme si l'objet de rubrique avait été utilisé directement.

Si l'objet en cours d'ouverture est un objet de rubrique, que ce soit avec *ObjectName* seul ou à l'aide de *ObjectString* (avec ou sans base *ObjectName*), le gestionnaire de files d'attente effectue le contrôle de sécurité à l'aide de la chaîne de rubrique résultante, extraite de l'objet de rubrique spécifié dans *ObjectName*, et, si nécessaire, en le concaténant avec celui fourni dans *ObjectString*, puis en recherchant l'objet de rubrique le plus proche à ce point ou au-dessus de ce point dans l'arborescence de rubriques pour effectuer le contrôle de sécurité. Il se peut qu'il ne s'agit pas du même objet de rubrique que celui spécifié dans *ObjectName*.

Si l'objet ouvert est une file d'attente modèle, le gestionnaire de files d'attente effectue une vérification de sécurité complète par rapport au nom de la file d'attente modèle et au nom de la file d'attente dynamique créée. Si la file d'attente dynamique résultante est ouverte de manière explicite, une vérification supplémentaire de la sécurité des ressources est effectuée par rapport au nom de la file d'attente dynamique.

Attribut

Les remarques suivantes concernent les attributs.

Les attributs d'un objet peuvent changer lorsque l'objet est ouvert dans une application. Dans de nombreux cas, l'application ne le remarque pas, mais pour certains attributs, le gestionnaire de files d'attente marque l'identificateur comme n'étant plus valide. Ces attributs sont les suivants:

- Tout attribut qui affecte la résolution de nom de l'objet. Cela s'applique indépendamment des options ouvertes utilisées et inclut les éléments suivants:
  - Modification de l'attribut *BaseQName* d'une file d'attente alias ouverte.
  - Modification de l'attribut *TargetType* d'une file d'attente alias ouverte.
  - Modification des attributs de file d'attente *RemoteQName* ou *RemoteQMgrName* , pour tout descripteur ouvert pour cette file d'attente ou pour une file d'attente résolue via cette définition en tant qu'alias de gestionnaire de files d'attente.
  - Toute modification entraînant la résolution d'un descripteur actuellement ouvert pour une file d'attente éloignée dans une autre file d'attente de transmission ou l'échec de la résolution en une seule file d'attente. Par exemple, il peut s'agir des éléments suivants:
    - Modification de l'attribut *XmitQName* de la définition locale d'une file d'attente éloignée, que la définition soit utilisée pour une file d'attente ou pour un alias de gestionnaire de files d'attente.
    - (z/OS uniquement) Modification de la valeur de l'attribut de gestionnaire de files d'attente *IntraGroupQueuing* ou modification de la définition de la file d'attente de transmission partagée (SYSTEM.QSG.TRANSMIT.QUEUE) utilisée par l'agent IGQ.

    Il y a une exception à cela: la création d'une nouvelle file d'attente de transmission. Un descripteur qui aurait été résolu dans cette file d'attente s'il avait été présent lors de l'ouverture du descripteur, mais qui aurait été résolu dans la file d'attente de transmission par défaut, n'est pas rendu non valide.
  - Modification de l'attribut de gestionnaire de files d'attente *DefXmitQName* . Dans ce cas, tous les descripteurs ouverts résolus dans la file d'attente précédemment nommée (qui ont été résolus uniquement parce qu'il s'agissait de la file d'attente de transmission par défaut) sont marqués comme non valides. Les descripteurs résolus dans cette file d'attente pour d'autres raisons ne sont pas affectés.
- L'attribut de file d'attente *Shareability* , s'il existe au moins deux descripteurs qui fournissent actuellement un accès MQOO_INPUT_SHARED pour cette file d'attente ou pour une file d'attente qui se résout en cette file d'attente. Si tel est le cas, *tous* les descripteurs qui sont ouverts pour cette file d'attente, ou pour une file d'attente qui se résout en cette file d'attente, sont marqués comme non valides, quelles que soient les options d'ouverture.

  Sous z/OS, les descripteurs décrits précédemment sont marqués comme non valides si un ou plusieurs descripteurs fournissent actuellement un accès MQOO_INPUT_SHARED ou MQOO_INPUT_EXCLUSIVE à la file d'attente.
- L'attribut de file d'attente *Usage* , pour tous les descripteurs ouverts pour cette file d'attente, ou pour une file d'attente qui se résout en cette file d'attente, quelles que soient les options d'ouverture.

Lorsqu'un descripteur est marqué comme non valide, tous les appels ultérieurs (autres que MQCLOSE) utilisant ce descripteur échouent avec le code anomalie MQRC_OBJECT_CHANGED. L'application doit émettre un appel MQCLOSE (à l'aide du descripteur d'origine), puis rouvrir la file d'attente. Toutes les mises à jour non validées par rapport à l'ancien descripteur des appels réussis précédents peuvent toujours être validées ou annulées, comme requis par la logique de l'application.

Si la modification d'un attribut entraîne cette situation, utilisez une version forcée spéciale de l'appel.

Appel C

```
MQOPEN (Hconn, &ObjDesc, Options, &Hobj, &CompCode,
       &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQOPEN' USING HCONN, OBJDESC, OPTIONS, HOBJ, COMPCODE, REASON
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQOPEN (Hconn, ObjDesc, Options, Hobj, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQOPEN,(HCONN,OBJDESC,OPTIONS,HOBJ,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Appel Visual Basic

```
MQOPEN Hconn, ObjDesc, Options, Hobj, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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-Insertion d'un message

L'appel MQPUT place un message dans une file d'attente ou une liste de distribution, ou dans une rubrique. La file d'attente, la liste de distribution ou la rubrique doit déjà être ouverte.

Syntaxe

MQPUT (*Hconn*, *Hobj*, *MsgDesc*, *PutMsgOpts*, *BufferLength*, *Buffer*, *CompCode*, *Raison*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

Hobj
Type : MQHOBJ - entrée

Cet identificateur représente la file d'attente à laquelle le message est ajouté ou la rubrique dans laquelle le message est publié. La valeur de *Hobj* a été renvoyée par un appel MQOPEN précédent qui a spécifié l'option MQOO_OUTPUT.

MsgDesc
Type : MQMD - entrée/sortie

Cette structure décrit les attributs du message envoyé et reçoit des informations sur le message une fois la demande d'insertion terminée. Voir «MQMD-Descripteur de message», à la page 394 pour plus de détails.

Si l'application fournit un MQMD version-1 , les données de message peuvent être préfixées avec une structure MQMDE afin de spécifier des valeurs pour les zones qui existent dans le MQMD version-2 mais pas dans version-1. La zone *Format* du MQMD doit être définie sur MQFMT_MD_EXTENSION

pour indiquer qu'un MQMDE est présent. Pour plus d'informations, voir «MQMDE-Extension de descripteur de message», à la page 447.

L'application n'a pas besoin de fournir une structure MQMD si un descripteur de message valide est fourni dans les zones *OriginalMsgHandle* ou *NewMsgHandle* de la structure MQPMO. Si aucune donnée n'est fournie dans l'un de ces champs, le descripteur du message est extrait du descripteur associé aux descripteurs de message.

Si vous utilisez ou prévoyez d'utiliser des exits API, nous vous recommandons de fournir explicitement une structure MQMD et de ne pas utiliser les descripteurs de message associés aux descripteurs de message. En effet, l'exit API associé à l'appel MQPUT ou MQPUT1 ne parvient pas à déterminer les valeurs MQMD utilisées par le gestionnaire de files d'attente pour exécuter la demande MQPUT ou MQPUT1 .

PutMsgOpts
Type: MQPMO-entrée/sortie

Voir «MQPMO-Options d'insertion de message», à la page 477 pour plus de détails.

BufferLength
Type : MQLONG - entrée

Longueur du message dans *Buffer*. La valeur zéro est valide et indique que le message ne contient aucune donnée d'application. La limite supérieure de *BufferLength* dépend de différents facteurs:

- Si la destination est une file d'attente locale ou est convertie en file d'attente locale, la limite supérieure varie selon que:
  - Le gestionnaire de files d'attente local prend en charge la segmentation.
  - L'application émettrice spécifie l'indicateur qui permet au gestionnaire de files d'attente de segmenter le message. Cet indicateur est MQMF_SEGMENTATION_ALLOWED et peut être spécifié dans un MQMD version-2 ou dans un MQMDE utilisé avec un MQMD version-1 .

  Si ces deux conditions sont satisfaites, *BufferLength* ne peut pas dépasser 999 999 999 moins la valeur de la zone *Offset* dans MQMD. Le message logique le plus long pouvant être inséré est donc 999 999 999 octets (lorsque *Offset* est égal à zéro). Toutefois, les contraintes de ressources imposées par le système d'exploitation ou l'environnement dans lequel l'application s'exécute peuvent entraîner une limite inférieure.

  Si l'une des conditions ci-dessus ou les deux conditions ci-dessus ne sont pas remplies, *BufferLength* ne peut pas dépasser la plus petite valeur de l'attribut *MaxMsgLength* de la file d'attente et de l'attribut *MaxMsgLength* du gestionnaire de files d'attente.
- Si la destination est une file d'attente éloignée ou est convertie en file d'attente éloignée, les conditions des files d'attente locales s'appliquent, *mais au niveau de chaque gestionnaire de files d'attente via lequel le message doit être transmis pour atteindre la file d'attente de destination*; en particulier:
  1. File d'attente de transmission locale utilisée pour stocker temporairement le message sur le gestionnaire de files d'attente local
  2. Files d'attente de transmission intermédiaires (le cas échéant) utilisées pour stocker les messages au niveau des gestionnaires de files d'attente sur la route entre les gestionnaires de files d'attente local et de destination
  3. File d'attente de destination sur le gestionnaire de files d'attente de destination

  Le message le plus long pouvant être inséré est donc régi par les files d'attente et les gestionnaires de files d'attente les plus restrictifs.

  Lorsqu'un message se trouve dans une file d'attente de transmission, des informations supplémentaires se trouvent avec les données de message, ce qui réduit la quantité de données d'application pouvant être transférées. Dans cette situation, soustrayez les octets MQ_MSG_HEADER_LENGTH des valeurs *MaxMsgLength* des files d'attente de transmission lors de la détermination de la limite pour *BufferLength*.

Remarque : Seul le non-respect de la condition 1 peut être diagnostiqué de manière synchrone (avec le code anomalie MQRC_MSG_TOO_BIG_FOR_Q ou MQRC_MSG_TOO_BIG_FOR_Q_MGR) lorsque le message est inséré. Si les conditions 2 ou 3 ne sont pas satisfaites, le message est redirigé vers une file d'attente de rebut (message non distribué), soit au niveau d'un gestionnaire de files d'attente intermédiaire, soit au niveau du gestionnaire de files d'attente de destination. Si cela se produit, un message de rapport est généré si l'expéditeur en a demandé un.

Tampon
Type: MQBYTExBufferLength-input

Il s'agit d'une mémoire tampon contenant les données d'application à envoyer. La mémoire tampon doit être alignée sur une limite appropriée à la nature des données du message. L'alignement sur 4 octets convient à la plupart des messages (y compris les messages contenant des structures d'en-tête WebSphere MQ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message contenant un entier binaire de 64 bits peut nécessiter un alignement de 8 octets.

Si *Buffer* contient des données de type caractère ou numérique, définissez les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* sur les valeurs appropriées aux données ; cela permet au destinataire du message de convertir les données (si nécessaire) en jeu de caractères et en codage utilisés par le destinataire.

Remarque : Tous les autres paramètres de l'appel MQPUT doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local (indiqués par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et MQENC_NATIVE).

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si le paramètre *BufferLength* a pour valeur zéro, *Buffer* n'est pas référencé ; dans ce cas, il se peut que l'adresse de paramètre transmise par les programmes écrits en langage C ou assembleur System/390 ait pour valeur NULL.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Groupe de messages non complet.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Message logique non complet.

MQRC_INCONSISTENT_PERSISTENCE
(2185, X'889') Spécification de persistance incohérente.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

MQRC_MULTIPLE_MOTIFS
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

MQRC_PRIORITY_MAXIMAL_DÉPASSEMENTS
(2049, X'801') La priorité de message dépasse la valeur maximale prise en charge.

MQRC_UNKNOWN_REPORT_OPTION
(2104, X'838') Option (s) de rapport dans le descripteur de message non reconnue.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ALIAS_TARGTYPE_CHANGED
(2480, X'09B0') Le type de cible d'abonnement a été modifié de file d'attente en objet de rubrique.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BACKED_OUT
(2003, X'7D3') Unité d'oeuvre annulée.

MQRC_BUFFER_ERROR
(2004, X'7D4') Paramètre de tampon non valide.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CALL_INTERROMPUE
(2549, X'9F5') MQPUT ou MQCMIT a été interrompu et le traitement de la reconnexion ne peut pas rétablir un résultat défini.

MQRC_CF_STRUC_FAILED
(2373, X'945') Echec de la structure de l'unité de couplage.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

MQRC_CFGR_ERREUR
(2416, X' 970') La structure de paramètre de groupe PCF MQCFGR dans les données de message n'est pas valide.

MQRC_CFH_ERREUR
(2235, X'8BB') Structure d'en-tête PCF non valide.

MQRC_CFIF_ERREUR
(2414, X'96E') La structure du paramètre de filtre d'entier PCF dans les données de message n'est pas valide.

MQRC_CFIL_ERREUR
(2236, X'8BC') Structure de paramètre de liste d'entiers PCF ou PCIF*64 structure de paramètre de liste d'entiers non valide.

MQRC_CFIN_ERREUR
(2237, X'8BD') Structure de paramètre d'entier PCF ou PCIF*64 structure de paramètre d'entier non valide.

MQRC_CFSF_ERREUR
(2415, X'96F') La structure de paramètre de filtre de chaîne PCF dans les données de message n'est pas valide.

MQRC_CFSL_ERREUR
(2238, X'8BE') Structure de paramètre de liste de chaînes PCF incorrecte.

MQRC_CFST_ERREUR
(2239, X'8BF') Structure de paramètre de chaîne PCF incorrecte.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CLUSTER_EXIT_ERREUR
(2266, X'8DA') L'exit de charge de travail de cluster a échoué.

Erreur de résolution MQRC_CLUSTER_RESOLUTION_ERROR
(2189, X'88D') La résolution du nom de cluster a échoué.

MQRC_CLUSTER_RESOURCE_ERROR
(2269, X'8DD') Erreur de ressource de cluster.

MQRC_COD_NOT_VALID_FOR_XCF_Q
(2106, X'83A') L'option de rapport COD n'est pas valide pour la file d'attente XCF.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Connexion en cours de mise au repos.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_CONTENT_ERREUR
2554 (X'09FA') Le contenu du message n'a pas pu être analysé pour déterminer si le message doit être distribué à un abonné avec un sélecteur de message étendu.

MQRC_CONTEXT_HANDLE_ERREUR
(2097, X'831') L'identificateur de file d'attente auquel il est fait référence ne sauvegarde pas le contexte.

MQRC_CONTEXT_NOT_AVAILABLE
(2098, X'832') Contexte non disponible pour le descripteur de file d'attente référencé.

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Paramètre de longueur des données non valide.

MQRC_DH_ERREUR
(2135, X'857') Structure d'en-tête de distribution incorrecte.

MQRC_DLH_ERREUR
(2141, X'85D') Structure d'en-tête de rebut non valide.

MQRC_EPH_ERREUR
(2420, X' 974') Structure PCF intégrée incorrecte.

MQRC_EXPIRY_ERROR
(2013, X'7DD') Heure d'expiration non valide.

MQRC_ERREUR_FEEDBACK_ERROR
(2014, X'7DE') Code retour non valide.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Conflit entre les unités d'oeuvre globales.

MQRC_GROUP_ID_ERREUR
(2258, X'8D2') Identificateur de groupe non valide.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HEADER_ERREUR
(2142, X'85E') Structure d'en-tête MQ non valide.

MQRC_HOBJ_ERROR
(2019, X'7E3') Descripteur d'objet non valide.

MQRC_IIH_ERREUR
(2148, X'864') La structure d'en-tête d'information IMS n'est pas valide.

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Groupe de messages non complet.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Message logique non complet.

MQRC_INCONSISTENT_PERSISTENCE
(2185, X'889') Spécification de persistance incohérente.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Spécification d'unité d'oeuvre incohérente.

MQRC_LOCAL_UOW_CONFLICT
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

MQRC_MD_ERROR
(2026, X'7EA') Descripteur de message non valide.

MQRC_MDE_ERREUR
(2248, X'8C8') Extension de descripteur de message non valide.

MQRC_MISSING_REPLY_TO_Q
(2027, X'7EB') Une file d'attente de réponse manquante ou MQPMO_SUPPRESS_REPLYTO a été utilisée

MQRC_WIH
(2332, X'91C') Les données de message ne commencent pas par MQWIH.

MQRC_MSG_FLAGS_ERREUR
(2249, X'8C9') Indicateurs de message non valides.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Numéro de séquence de message non valide.

MQRC_MSG_TOO_BIG_FOR_Q
(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

MQRC_MSG_TYPE_ERREUR
(2029, X'7ED') Le type de message dans le descripteur de message n'est pas valide.

MQRC_MULTIPLE_MOTIFS
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

MQRC_NO_DESTINATIONS_AVAILABLE
(2270, X'8DE') Aucune file d'attente de destination disponible.

MQRC_NOT_OPEN_FOR_OUTPUT
(2039, X'7F7') File d'attente non ouverte pour la sortie.

MQRC_NOT_OPEN_FOR_PASS_ALL
(2093, X'82D') File d'attente non ouverte pour le passage de tous les contextes.

MQRC_NOT_OPEN_FOR_PASS_IDENT
(2094, X'82E') File d'attente non ouverte pour le contexte d'identité de transmission.

MQRC_NOT_OPEN_FOR_SET_ALL
(2095, X'82F') File d'attente non ouverte pour tous les contextes.

MQRC_NOT_OPEN_FOR_SET_IDENT
(2096, X'830') File d'attente non ouverte pour le contexte d'identité défini.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

ERREUR MQRC_OFFSET_ERROR
(2251, X'8CB') Position de segment de message non valide.

MQRC_OPEN_FAILED
(2137, X'859') L'ouverture de l'objet a échoué.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_ORIGINAL_LENGTH_ERROR
(2252, X'8CC') Longueur d'origine incorrecte.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_PAGESET_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_PCF_ERREUR
(2149, X'865') Structures PCF incorrectes.

MQRC_XX_ENCODE_CASE_ONE err_persistant
(2047, X'7FF') Persistance non valide.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.

MQRC_PMO_ERREUR
(2173, X'87D') La structure des options d'insertion de message n'est pas valide.

MQRC_PMO_RECORD_FLAGS_ERROR
(2158, X'86E') Les indicateurs d'enregistrement de message d'insertion ne sont pas valides.

MQRC_PRIORITY_ERROR
(2050, X'802') Priorité de message non valide.

MQRC_PUBLICATION_FAILURE
(2502, X'9C6') La publication n'a été distribuée à aucun des abonnés.

MQRC_PUT_INHIBÉ
(2051, X'803') Les appels d'insertion sont interdits pour la file d'attente, pour la file d'attente dans laquelle cette file d'attente est résolue ou pour la rubrique.

MQRC_PUT_MSG_RECORDS_ERROR
(2159, X'86F') Enregistrements de message d'insertion non valides.

MQRC_PUT_NOT_RETENU
(2479, X'09AF') La publication n'a pas pu être conservée

MQRC_Q_DELETED
(2052, X'804') File d'attente supprimée.

MQRC_Q_FULL
(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_QUIESCING
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_Q_ESPACE_NON_DISPONIBLE
(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.

Echec de MQRC_RECONNECT_FAILED
(2548, X'9F4') Après la reconnexion, une erreur s'est produite lors de la réinstallation des descripteurs d'une connexion reconnectable.

MQRC_RECS_Présentateur-ERREUR
(2154, X'86A') Nombre d'enregistrements présents non valide.

MQRC_REPORT_OPTIONS_ERREUR
(2061, X'80D') Les options de rapport dans le descripteur de message ne sont pas valides.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Enregistrements de réponse non valides.

MQRC_RFH_ERREUR
(2334, X'91E') Structure MQRFH ou MQRFH2 non valide.

MQRC_RMH_ERREUR
(2220, X'8AC') La structure d'en-tête de message de référence n'est pas valide.

MQRC_SEGMENT_LONGUEUR_ZÉRO
(2253, X'8CD') La longueur des données dans le segment de message est égale à zéro.

MQRC_SEGMENTS_NON_PRIS en charge
(2365, X'93D') Segments non pris en charge.

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') Il existe un abonné possible pour la publication, mais le gestionnaire de files d'attente ne peut pas vérifier si la publication doit être envoyée à l'abonné.

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Appel rejeté par l'exit de charge de travail du cluster.

MQRC_STORAGE_CLASS_ERROR
(2105, X'839') Erreur de classe de stockage.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_SYNCPOINT_LIMIT_REACHED
(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818') Prise en charge du point de synchronisation non disponible.

MQRC_TM_ERREUR
(2265, X'8D9') La structure du message de déclenchement n'est pas valide.

MQRC_TMC_ERREUR
(2191, X'88F') La structure de message du déclencheur de caractères n'est pas valide.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

MQRC_WIH_ERREUR
(2333, X'91D') Structure MQWIH non valide.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Version MQMD fournie non valide.

MQRC_XQH_ERREUR
(2260, X'8D4') Structure d'en-tête de file d'attente de transmission non valide.

Pour plus de détails sur ces codes, voir Codes anomalie.

Remarques sur l'utilisation des rubriques

1. Les remarques suivantes s'appliquent à l'utilisation des rubriques:

   a. Lors de l'utilisation de MQPUT pour publier des messages sur une rubrique, où un ou plusieurs abonnés à cette rubrique ne peuvent pas recevoir la publication en raison d'un problème avec leur file d'attente d'abonné (par exemple, elle est saturée), le code anomalie renvoyé à l'appel MQPUT et le comportement de distribution dépendent de la définition des attributs PMSGDLV ou NPMSGDLV sur le TOPIC. La distribution d'une publication dans la file d'attente des messages non livrés lorsque MQRO_DEAD_LETTER_Q est spécifié, ou la suppression du message lorsque MQRO_DISCARD_MSG est spécifié, est considérée comme une distribution réussie du message. Si aucune des publications n'est distribuée, MQPUT est renvoyé avec MQRC_PUBLICATION_FAILURE. Cela peut se produire dans les cas suivants:

      - Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALL et tout abonnement (durable ou non) possède une file d'attente qui ne peut pas recevoir la publication.
      - Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLDUR et un abonnement durable possède une file d'attente qui ne peut pas recevoir la publication.

      MQPUT peut être renvoyé avec MQRC_NONE même si les publications n'ont pas pu être distribuées à certains abonnés dans les cas suivants:

      - Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLAVAIL et tout abonnement, durable ou non, possède une file d'attente qui ne peut pas recevoir la publication.
      - Un message est publié sur un TOPIC avec PMSGDLV ou NPMSGDLV (en fonction de la persistance du message) défini sur ALLDUR et un abonnement non durable possède une file d'attente qui ne peut pas recevoir la publication.

      Vous pouvez utiliser l'attribut de rubrique USEDLQ pour déterminer si la file d'attente de rebut est utilisée lorsque les messages de publication ne peuvent pas être distribués à leur file d'attente de souscription correcte. Pour plus d'informations sur l'utilisation de USEDLQ, voir DEFINE TOPIC.

   b. S'il n'y a pas d'abonnés à la rubrique utilisée, le message publié n'est envoyé à aucune file d'attente et est supprimé. Peu importe que le message soit persistant ou non persistant, qu'il ait une expiration illimitée ou qu'il ait une heure d'expiration, il est tout de même supprimé s'il n'y a pas d'abonnés. La seule exception à cette règle est si le message doit être conservé, auquel cas, bien qu'il ne soit pas envoyé aux files d'attente des abonnés, il est stocké par rapport à la rubrique à distribuer aux nouveaux abonnements ou aux abonnés qui demandent des publications conservées à l'aide de MQSUBRQ.

MQPUT et MQPUT1

Vous pouvez utiliser les appels MQPUT et MQPUT1 pour placer des messages dans une file d'attente ; l'appel à utiliser dépend des circonstances

- Utilisez l'appel MQPUT pour placer plusieurs messages dans la *même* file d'attente.

  Un appel MQOPEN spécifiant l'option MQOO_OUTPUT est émis en premier, suivi d'une ou de plusieurs demandes MQPUT pour ajouter des messages à la file d'attente ; enfin, la file d'attente est fermée avec un appel MQCLOSE. Cela offre de meilleures performances que l'utilisation répétée de l'appel MQPUT1 .
- Utilisez l'appel MQPUT1 pour placer *un seul* message dans une file d'attente.

  Cet appel encapsule les appels MQOPEN, MQPUT et MQCLOSE en un seul appel, ce qui réduit le nombre d'appels qui doivent être émis.

Files d'attente de destination

Les remarques suivantes s'appliquent à l'utilisation des files d'attente de destination:

1. Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages,l'ordre de ces messages est conservé si les conditions détaillées sont remplies. Certaines conditions s'appliquent aux files d'attente de destination locales et éloignées ; d'autres conditions s'appliquent uniquement aux files d'attente de destination éloignées.

   Conditions qui s'appliquent aux files d'attente de destination locales et distantes

   - Tous les appels MQPUT se trouvent dans la même unité de travail, ou aucun d'entre eux ne se trouve dans une unité de travail.

     Sachez que lorsque des messages sont placés dans une file d'attente particulière au sein d'une seule unité d'oeuvre, les messages provenant d'autres applications peuvent être associés à la séquence de messages de la file d'attente.
   - Tous les appels MQPUT sont effectués à l'aide du même descripteur d'objet *Hobj*.

     Dans certains environnements, la séquence de messages est également conservée lorsque des descripteurs d'objet différents sont utilisés, si les appels sont effectués à partir de la même application. La signification de *même application* est déterminée par l'environnement:

     – Sous z/OS, l'application est:

       - Pour CICS, la tâche CICS
       - Pour IMS, la tâche
       - Pour le traitement par lots z/OS , la tâche

     – Sous IBM i, l'application est le travail.

     – Sur les systèmes Windows et UNIX , l'application est l'unité d'exécution.
   - Les messages ont tous la même priorité.
   - Les messages ne sont pas insérés dans une file d'attente de cluster avec MQOO_BIND_NOT_FIXED spécifié (ou avec MQOO_BIND_AS_Q_DEF en vigueur lorsque l'attribut de file d'attente DefBind a la valeur MQBND_BIND_NOT_FIXED).

   Conditions supplémentaires qui s'appliquent aux files d'attente de destination éloignées

   - Il n'existe qu'un seul chemin entre le gestionnaire de files d'attente émetteur et le gestionnaire de files d'attente cible.

     Si certains messages de la séquence peuvent se trouver dans un chemin différent (par exemple, en raison de la reconfiguration, de l'équilibrage du trafic ou de la sélection du chemin en fonction de la taille des messages), l'ordre des messages sur le gestionnaire de files d'attente de destination ne peut pas être garanti.
   - Les messages ne sont pas placés temporairement dans des files d'attente de rebut au niveau des gestionnaires de files d'attente d'envoi, intermédiaire ou de destination.

Si un ou plusieurs messages sont placés temporairement dans une file d'attente de rebut (par exemple, parce qu'une file d'attente de transmission ou la file d'attente de destination est temporairement saturée), les messages peuvent arriver dans la file d'attente de destination hors séquence.

- Les messages sont tous persistants ou non persistants.

  Si l'attribut *NonPersistentMsgSpeed* d'un canal sur la route entre les gestionnaires de files d'attente d'envoi et de destination est défini sur MQNPMS_FAST, les messages non persistants peuvent prendre le pas sur les messages persistants, ce qui a pour conséquence que l'ordre des messages persistants par rapport aux messages non persistants n'est pas conservé. Cependant, l'ordre des messages persistants les uns par rapport aux autres et des messages non persistants les uns par rapport aux autres est conservé.

Si ces conditions ne sont pas remplies, vous pouvez utiliser des groupes de messages pour conserver l'ordre des messages, mais cela nécessite que les applications d'envoi et de réception utilisent la prise en charge du regroupement de messages. Pour plus d'informations sur les groupes de messages, voir :

- MQMD - Zone MsgFlags
- MQPMO_LOGICAL_ORDER
- MQGMO_LOGICAL_ORDER

Listes de distribution

Les remarques suivantes s'appliquent à l'utilisation des listes de distribution.

Les listes de distribution sont prises en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus les clients WebSphere MQ MQI connectés à ces systèmes.

1. Vous pouvez placer des messages dans une liste de distribution à l'aide d'une instance version-1 ou d'une instance version-2 MQPMO. Si vous utilisez un MQPMO version-1 (ou un MQPMO version-2 avec *RecsPresent* égal à zéro), l'application ne peut pas fournir d'enregistrements de message d'insertion ni d'enregistrements de réponse. Vous ne pouvez pas identifier les files d'attente qui rencontrent des erreurs si le message est envoyé avec succès à certaines files d'attente de la liste de distribution et non à d'autres.

   Si l'application fournit des enregistrements de message d'insertion ou des enregistrements de réponse, définissez la zone *Version* sur MQPMO_VERSION_2.

   Vous pouvez également utiliser un MQPMO version-2 pour envoyer des messages à une file d'attente unique qui ne figure pas dans une liste de distribution, en vous assurant que *RecsPresent* est égal à zéro.
2. Les paramètres de code achèvement et de code anomalie sont définis comme suit:
   - Si les insertions dans les files d'attente de la liste de distribution aboutissent ou échouent de la même manière, les paramètres de code achèvement et de code raison sont définis pour décrire le résultat commun. Les enregistrements de réponse MQRR (s'ils sont fournis par l'application) ne sont pas définis dans ce cas.

     Par exemple, si chaque insertion aboutit, le code achèvement et le code anomalie sont définis sur MQCC_OK et MQRC_NON; si chaque insertion échoue car toutes les files d'attente sont interdites pour les insertions, les paramètres sont définis sur MQCC_FAILED et MQRC_PUT_INHIBÉ.
   - Si les insertions dans les files d'attente de la liste de distribution n'aboutissent pas toutes ou échouent de la même manière:
     - Le paramètre de code achèvement est défini sur MQCC_WARNING si au moins une insertion a abouti et sur MQCC_FAILED si toutes les opérations ont échoué.
     - Le paramètre de code anomalie est défini sur MQRC_MULTIPLE_MOTIFS.
     - Les enregistrements de réponse (s'ils sont fournis par l'application) sont définis sur les codes achèvement individuels et les codes raison des files d'attente de la liste de distribution.

Si l'insertion dans une destination échoue en raison de l'échec de l'ouverture de cette destination, les zones de l'enregistrement de réponse sont définies sur MQCC_FAILED et MQRC_OPEN_FAILED; cette destination est incluse dans *InvalidDestCount*.

3. Si une destination de la liste de distribution se résout en une file d'attente locale, le message est placé dans cette file d'attente sous la forme normale (c'est-à-dire qu'il ne s'agit pas d'un message de liste de distribution). Si plusieurs destinations sont résolues dans la même file d'attente locale, un message est placé dans la file d'attente pour chaque destination.

   Si une destination de la liste de distribution est résolue en file d'attente éloignée, un message est placé dans la file d'attente de transmission appropriée. Lorsque plusieurs destinations sont résolues dans la même file d'attente de transmission, un seul message de liste de distribution contenant ces destinations peut être placé dans la file d'attente de transmission, même si ces destinations n'étaient pas adjacentes dans la liste de destinations fournie par l'application. Toutefois, cette opération ne peut être effectuée que si la file d'attente de transmission prend en charge les messages de liste de distribution (voir DistLists).

   Si la file d'attente de transmission ne prend pas en charge les listes de distribution, une copie du message au format normal est placée dans la file d'attente de transmission pour chaque destination qui utilise cette file d'attente de transmission.

   Si une liste de distribution avec les données de message d'application est trop grande pour une file d'attente de transmission, le message de liste de distribution est divisé en messages de liste de distribution plus petits, chacun contenant moins de destinations. Si les données de message d'application ne correspondent qu'à la file d'attente, les messages de liste de distribution ne peuvent pas être utilisés du tout et le gestionnaire de files d'attente génère une copie du message au format normal pour chaque destination qui utilise cette file d'attente de transmission.

   Si différentes destinations ont une priorité de message ou une persistance de message différente (cela peut se produire lorsque l'application spécifie MQPRI_PRIORITY_AS_Q_DEF ou MQPER_PERSISTENCE_AS_Q_DEF), les messages ne sont pas conservés dans le même message de liste de distribution. Au lieu de cela, le gestionnaire de files d'attente génère autant de messages de liste de distribution que nécessaire pour prendre en charge les valeurs de priorité et de persistance différentes.

4. Une insertion dans une liste de distribution peut entraîner:

   - un message de liste de distribution unique, ou
   - Un certain nombre de messages de liste de distribution plus petits, ou
   - Une combinaison de messages de liste de distribution et de messages normaux, ou
   - Messages normaux uniquement.

   Ce qui précède se produit selon que:

   - Les destinations de la liste sont locales, distantes ou un mélange.
   - Les destinations ont la même priorité de message et la même persistance de message.
   - Les files d'attente de transmission peuvent contenir des messages de liste de distribution.
   - La longueur maximale des messages des files d'attente de transmission est suffisante pour accueillir le message sous forme de liste de distribution.

   Toutefois, quel que soit le message ci-dessus, chaque message *physique* résultant (c'est-à-dire chaque message normal ou de liste de distribution résultant de l'insertion) compte comme *un seul* message lorsque:

   - Vérifier si l'application a dépassé le nombre maximal de messages autorisé dans une unité de travail (voir l'attribut de gestionnaire de files d'attente *MaxUncommittedMsgs*).
   - Vérification du respect des conditions de déclenchement.
   - Incrémentation du nombre de lignes de la file d'attente et vérification du dépassement du nombre maximal de lignes de la file d'attente.

5. Toute modification apportée aux définitions de file d'attente qui aurait entraîné la non-validité d'un descripteur si les files d'attente avaient été ouvertes individuellement (par exemple, une modification du chemin de résolution) n'entraîne pas la non-validité du descripteur de liste de distribution. Toutefois, elle entraîne un échec pour cette file d'attente particulière lorsque le descripteur de liste de distribution est utilisé lors d'un appel MQPUT ultérieur.

En-têtes

Si un message est inséré avec une ou plusieurs structures d'en-tête WebSphere MQ au début des données de message d'application, le gestionnaire de files d'attente effectue certaines vérifications sur les structures d'en-tête pour vérifier qu'elles sont valides. Si le gestionnaire de files d'attente détecte une erreur, l'appel échoue avec un code anomalie approprié. Les contrôles effectués varient en fonction des structures particulières présentes:

- Les vérifications ne sont effectuées que si une version-2 ou une version ultérieure de MQMD est utilisée sur l'appel MQPUT ou MQPUT1 . Les vérifications ne sont pas effectuées si un MQMD version-1 est utilisé, même si un MQMDE est présent au début des données du message.
- Les structures qui ne sont pas prises en charge par le gestionnaire de files d'attente local et les structures qui suivent le premier MQDLH dans le message ne sont pas validées.
- Les structures MQDH et MQMDE sont entièrement validées par le gestionnaire de files d'attente.
- Les autres structures sont partiellement validées par le gestionnaire de files d'attente (toutes les zones ne sont pas vérifiées).

Les vérifications générales effectuées par le gestionnaire de files d'attente sont les suivantes:

- La zone *StrucId* doit être valide.
- La zone *Version* doit être valide.
- La zone *StrucLength* doit spécifier une valeur suffisamment grande pour inclure la structure plus les données de longueur variable qui font partie de la structure.
- La valeur de la zone *CodedCharSetId* ne doit pas être zéro ou une valeur négative non valide (MQCCSI_DEFAULT, MQCCSI_EMBEDDED, MQCCSI_Q_MGR et MQCCSI_UNDEFINED ne sont *pas* valides dans la plupart des structures d'en-tête WebSphere MQ).
- Le paramètre *BufferLength* de l'appel doit spécifier une valeur suffisamment élevée pour inclure la structure (la structure ne doit pas s'étendre au-delà de la fin du message).

Outre les contrôles généraux des structures, les conditions suivantes doivent être remplies:

- La somme des longueurs des structures d'un message PCF doit être égale à la longueur spécifiée par le paramètre *BufferLength* dans l'appel MQPUT ou MQPUT1 . Un message PCF est un message dont le nom de format est MQFMT_ADMIN, MQFMT_EVENT ou MQFMT_PCF.
- Une structure WebSphere MQ ne doit pas être tronquée, sauf dans les cas suivants où les structures tronquées sont autorisées:
  - Messages qui sont des messages de rapport.
  - Messages PCF.
  - Messages contenant une structure MQDLH. (Les structures *suivant* le premier MQDLH peuvent être tronquées ; les structures précédant le MQDLH ne peuvent pas.)
- Une structure WebSphere MQ ne doit pas être fractionnée sur deux ou plusieurs segments ; la structure doit être entièrement contenue dans un segment.

Tampon

Pour le langage de programmation Visual Basic, les points suivants s'appliquent :

- Si la taille du paramètre *Buffer* est inférieure à la longueur spécifiée par le paramètre *BufferLength* , l'appel échoue avec le code anomalie MQRC_BUFFER_LENGTH_ERROR.

- Le paramètre *Buffer* est déclaré comme étant de type String. Si les données à placer dans la file d'attente ne sont pas de type String, utilisez laAppel MQPUTAny à la place de MQPUT.

  L'appel MQPUTAny possède les mêmes paramètres que l'appel MQPUT, sauf que le paramètre *Buffer* est déclaré comme étant de type Any, ce qui permet de placer n'importe quel type de données dans la file d'attente. Cependant, cela signifie que le système ne peut pas vérifier que le paramètre *Buffer* comporte au minimum *BufferLength* octets.

Appel C

```
MQPUT (Hconn, Hobj, &MsgDesc, &PutMsgOpts, BufferLength, Buffer,
      &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
     CALL 'MQPUT' USING HCONN, HOBJ, MSGDESC, PUTMSGOPTS, BUFFERLENGTH,
                        BUFFER, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQPUT (Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer,
           CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQPUT,(HCONN,HOBJ,MSGDESC,PUTMSGOPTS,BUFFERLENGTH, X
               BUFFER,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Appel Visual Basic

```
MQPUT Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode,
      Reason
```

Déclarez les paramètres comme suit :

```
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 -Message d'insertion unique

L'appel MQPUT1 place un message dans une file d'attente, une liste de distribution ou une rubrique.

Il n'est pas nécessaire d'ouvrir la file d'attente, la liste de distribution ou la rubrique.

Syntaxe

MQPUT1 (*Hconn*, *ObjDesc*, *MsgDesc*, *PutMsgOpts*, *BufferLength*, *Buffer*, *CompCode*, *Reason*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

ObjDesc
Type: MQOD-entrée/sortie

Il s'agit d'une structure qui identifie la file d'attente à laquelle le message est ajouté ou la rubrique à laquelle le message est publié. Voir «MQOD-Descripteur d'objet», à la page 456 pour plus de détails.

Si la structure est une file d'attente, l'utilisateur doit être autorisé à ouvrir la file d'attente pour la sortie. La file d'attente ne doit **pas** être une file d'attente modèle.

MsgDesc
Type : MQMD - entrée/sortie

Cette structure décrit les attributs du message envoyé et reçoit des informations en retour une fois la demande d'insertion terminée. Voir «MQMD-Descripteur de message», à la page 394 pour plus de détails.

Si l'application fournit un MQMD version-1 , les données de message peuvent être préfixées avec une structure MQMDE afin de spécifier des valeurs pour les zones qui existent dans le MQMD version-2 mais pas dans version-1. Définissez la zone *Format* dans le MQMD sur MQFMT_MD_EXTENSION pour indiquer qu'un MQMDE est présent. Pour plus d'informations, voir «MQMDE-Extension de descripteur de message», à la page 447.

L'application n'a pas besoin de fournir une structure MQMD si un descripteur de message valide est fourni dans la zone *MsgHandle* de la structure MQGMO ou dans les zones *OriginalMsgHandle* ou *NewMsgHandle* de la structure MQPMO. Si aucune donnée n'est fournie dans l'un de ces champs, le descripteur du message est extrait du descripteur associé aux descripteurs de message.

PutMsgOpts
Type: MQPMO-entrée/sortie

Voir «MQPMO-Options d'insertion de message», à la page 477 pour plus de détails.

BufferLength
Type : MQLONG - entrée

Longueur du message dans *Buffer*. La valeur zéro est valide et indique que le message ne contient aucune donnée d'application. La limite supérieure dépend de différents facteurs ; pour plus de détails, voir la description du paramètre *BufferLength* de l'appel MQPUT.

Tampon
Type: MQBYTExBufferLength-input

Il s'agit d'une mémoire tampon contenant les données de message d'application à envoyer. Alignez la mémoire tampon sur une limite appropriée à la nature des données dans le message. L'alignement sur 4 octets convient à la plupart des messages (y compris les messages contenant des structures d'en-tête WebSphere MQ), mais certains messages peuvent nécessiter un alignement plus strict. Par exemple, un message contenant un entier binaire de 64 bits peut nécessiter un alignement de 8 octets.

Si *Buffer* contient des données de type caractère ou numérique, définissez les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* sur les valeurs appropriées aux données ; cela permet au destinataire du message de convertir les données (si nécessaire) en jeu de caractères et en codage utilisés par le destinataire.

Remarque : Tous les autres paramètres de l'appel MQPUT1 doivent être dans le jeu de caractères et le codage du gestionnaire de files d'attente local (indiqués par l'attribut de gestionnaire de files d'attente *CodedCharSetId* et MQENC_NATIVE).

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si le paramètre *BufferLength* a pour valeur zéro, *Buffer* n'est pas référencé ; dans ce cas, il se peut que l'adresse de paramètre transmise par les programmes écrits en langage C ou assembleur System/390 ait pour valeur NULL.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_MULTIPLE_MOTIFS
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Groupe de messages non complet.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Message logique non complet.

MQRC_PRIORITY_MAXIMAL_DÉPASSEMENTS
(2049, X'801') La priorité de message dépasse la valeur maximale prise en charge.

MQRC_UNKNOWN_REPORT_OPTION
(2104, X'838') Options de rapport dans le descripteur de message non reconnues.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ALIAS_BASE_Q_TYPE_ERROR
(2001, X'7D1') Le type de la file d'attente de base de l'alias n'est pas valide.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BACKED_OUT
(2003, X'7D3') Unité d'oeuvre annulée.

MQRC_BUFFER_ERROR
(2004, X'7D4') Paramètre de tampon non valide.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CF_NOT_XX_ENCODE_CASE_ONE disponible
(2345, X' 929') unité de couplage non disponible.

Echec de MQRC_CF_STRUC_AUTH_FAILED
(2348, X'92C') La vérification d'autorisation de la structure d'unité de couplage a échoué.

MQRC_CF_STRUC_ERREUR
(2349, X'92D') Structure d'unité de couplage non valide.

MQRC_CF_STRUC_FAILED
(2373, X'945') Echec de la structure de l'unité de couplage.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347, X'92B') En-tête de liste de la structure d'unité de couplage en cours d'utilisation.

MQRC_CFGR_ERREUR
(2416, X' 970') La structure de paramètre de groupe PCF MQCFGR dans les données de message n'est pas valide.

MQRC_CFH_ERREUR
(2235, X'8BB') Structure d'en-tête PCF non valide.

MQRC_CFIF_ERREUR
(2414, X'96E') La structure du paramètre de filtre d'entier PCF dans les données de message n'est pas valide.

MQRC_CFIL_ERREUR
(2236, X'8BC') Structure de paramètre de liste d'entiers PCF ou PCIF*64 structure de paramètre de liste d'entiers non valide.

MQRC_CFIN_ERREUR
(2237, X'8BD') Structure de paramètre d'entier PCF ou PCIF*64 structure de paramètre d'entier non valide.

MQRC_CFSF_ERREUR
(2415, X'96F') La structure de paramètre de filtre de chaîne PCF dans les données de message n'est pas valide.

MQRC_CFSL_ERREUR
(2238, X'8BE') Structure de paramètre de liste de chaînes PCF incorrecte.

MQRC_CFST_ERREUR
(2239, X'8BF') Structure de paramètre de chaîne PCF incorrecte.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CLUSTER_EXIT_ERREUR
(2266, X'8DA') L'exit de charge de travail de cluster a échoué.

Erreur de résolution MQRC_CLUSTER_RESOLUTION_ERROR
(2189, X'88D') La résolution du nom de cluster a échoué.

MQRC_CLUSTER_RESOURCE_ERROR
(2269, X'8DD') Erreur de ressource de cluster.

MQRC_COD_NOT_VALID_FOR_XCF_Q
(2106, X'83A') L'option de rapport COD n'est pas valide pour la file d'attente XCF.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Connexion en cours de mise au repos.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_CONTENT_ERREUR
2554 (X'09FA') Le contenu du message n'a pas pu être analysé pour déterminer si le message peut être distribué à un abonné avec un sélecteur de message étendu.

MQRC_CONTEXT_HANDLE_ERREUR
(2097, X'831') L'identificateur de file d'attente auquel il est fait référence ne sauvegarde pas le contexte.

MQRC_CONTEXT_NOT_AVAILABLE
(2098, X'832') Contexte non disponible pour le descripteur de file d'attente référencé.

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Paramètre de longueur des données non valide.

MQRC_DB2_NOT_AVAILABLE
(2342, X'926') Sous-système DB2 non disponible.

MQRC_DEF_XMIT_Q_TYPE_ERREUR
(2198, X'896') La file d'attente de transmission par défaut n'est pas locale.

MQRC_DEF_XMIT_Q_USAGE_ERROR
(2199, X'897') Erreur d'utilisation de la file d'attente de transmission par défaut.

MQRC_DH_ERREUR
(2135, X'857') Structure d'en-tête de distribution incorrecte.

MQRC_DLH_ERREUR
(2141, X'85D') Structure d'en-tête de rebut non valide.

MQRC_EPH_ERREUR
(2420, X' 974') Structure PCF intégrée incorrecte.

MQRC_EXPIRY_ERROR
(2013, X'7DD') Heure d'expiration non valide.

MQRC_ERREUR_FEEDBACK_ERROR
(2014, X'7DE') Code retour non valide.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Conflit entre les unités d'oeuvre globales.

MQRC_GROUP_ID_ERREUR
(2258, X'8D2') Identificateur de groupe non valide.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X'931') Descripteur en cours d'utilisation pour l'unité d'oeuvre globale.

MQRC_HANDLE_NOT_AVAILABLE
(2017, X'7E1') Plus de descripteurs disponibles.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HEADER_ERREUR
(2142, X'85E') Structure d'en-tête WebSphere MQ non valide.

MQRC_IIH_ERREUR
(2148, X'864') La structure d'en-tête d'information IMS n'est pas valide.

MQRC_LOCAL_UOW_CONFLICT
(2352, X'930') Conflit entre l'unité d'oeuvre globale et l'unité d'oeuvre locale.

MQRC_MD_ERROR
(2026, X'7EA') Descripteur de message non valide.

MQRC_MDE_ERREUR
(2248, X'8C8') Extension de descripteur de message non valide.

MQRC_MISSING_REPLY_TO_Q
(2027, X'7EB') File d'attente de réponse manquante.

MQRC_WIH
(2332, X'91C') Les données de message ne commencent pas par MQWIH.

MQRC_MSG_FLAGS_ERREUR
(2249, X'8C9') Indicateurs de message non valides.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Numéro de séquence de message non valide.

MQRC_MSG_TOO_BIG_FOR_Q
(2030, X'7EE') Longueur de message supérieure au maximum pour la file d'attente.

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031, X'7EF') Longueur de message supérieure à la longueur maximale pour le gestionnaire de files d'attente.

MQRC_MSG_TYPE_ERREUR
(2029, X'7ED') Le type de message dans le descripteur de message n'est pas valide.

MQRC_MULTIPLE_MOTIFS
(2136, X'858') Plusieurs codes anomalie ont été renvoyés.

MQRC_NO_DESTINATIONS_AVAILABLE
(2270, X'8DE') Aucune file d'attente de destination disponible.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

MQRC_OBJECT_IN_USE
(2042, X'7FA') Objet déjà ouvert avec des options en conflit.

MQRC_OBJET_NIVEAU_INCOMPATIBLE
(2360, X' 938') Niveau d'objet non compatible.

MQRC_OBJECT_NAME_ERROR
(2152, X'868') Nom d'objet incorrect.

MQRC_OBJECT_NOT_UNIQUE
(2343, X' 927') Objet non unique.

MQRC_OBJECT_Q_MGR_NAME_ERROR
(2153, X'869') Nom de gestionnaire de files d'attente d'objet incorrect.

MQRC_OBJECT_RECORDS_ERROR
(2155, X'86B') Enregistrements d'objet non valides.

MQRC_OBJECT_TYPE_ERREUR
(2043, X'7FB') Type d'objet non valide.

MQRC_ERREUR D'OD_DU
(2044, X'7FC') Structure de descripteur d'objet non valide.

ERREUR MQRC_OFFSET_ERROR
(2251, X'8CB') Position de segment de message non valide.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_ORIGINAL_LENGTH_ERROR
(2252, X'8CC') Longueur d'origine incorrecte.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_PAGESET_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_PCF_ERREUR
(2149, X'865') Structures PCF incorrectes.

MQRC_XX_ENCODE_CASE_ONE err_persistant
(2047, X'7FF') Persistance non valide.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800') La file d'attente ne prend pas en charge les messages persistants.

MQRC_PMO_ERREUR
(2173, X'87D') La structure des options d'insertion de message n'est pas valide.

MQRC_PMO_RECORD_FLAGS_ERROR
(2158, X'86E') Les indicateurs d'enregistrement de message d'insertion ne sont pas valides.

MQRC_PRIORITY_ERROR
(2050, X'802') Priorité de message non valide.

MQRC_PUBLICATION_FAILURE
(2502, X'9C6') La publication n'a été distribuée à aucun des abonnés.

MQRC_PUT_INHIBÉ
(2051, X'803') Appels d'insertion bloqués pour la file d'attente.

MQRC_PUT_MSG_RECORDS_ERROR
(2159, X'86F') Enregistrements de message d'insertion non valides.

MQRC_Q_DELETED
(2052, X'804') File d'attente supprimée.

MQRC_Q_FULL
(2053, X'805') La file d'attente contient déjà le nombre maximal de messages.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_QUIESCING
(2161, X'871') Gestionnaire de files d'attente en cours de mise au repos.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_Q_ESPACE_NON_DISPONIBLE
(2056, X'808') Aucun espace disponible sur le disque pour la file d'attente.

MQRC_Q_TYPE_ERREUR
(2057, X'809') Type de file d'attente non valide.

MQRC_RECS_Présentateur-ERREUR
(2154, X'86A') Nombre d'enregistrements présents non valide.

MQRC_REMOTE_Q_NAME_ERROR
(2184, X'888') Nom de file d'attente éloignée incorrect.

MQRC_REPORT_OPTIONS_ERREUR
(2061, X'80D') Les options de rapport dans le descripteur de message ne sont pas valides.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Enregistrements de réponse non valides.

MQRC_RFH_ERREUR
(2334, X'91E') Structure MQRFH ou MQRFH2 non valide.

MQRC_RMH_ERREUR
(2220, X'8AC') La structure d'en-tête de message de référence n'est pas valide.

MQRC_SECURITY_ERROR (ERREUR DE SECURITE MQ)
(2063, X'80F') Une erreur de sécurité s'est produite.

MQRC_SEGMENT_LONGUEUR_ZÉRO
(2253, X'8CD') La longueur des données dans le segment de message est égale à zéro.

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') Il existe un abonné possible pour la publication, mais le gestionnaire de files d'attente ne peut pas vérifier si la publication doit être envoyée à l'abonné.

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Appel rejeté par l'exit de charge de travail du cluster.

MQRC_STORAGE_CLASS_ERROR
(2105, X'839') Erreur de classe de stockage.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890') Support de stockage externe saturé.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_SYNCPOINT_LIMIT_REACHED
(2024, X'7E8') Aucun autre message pouvant être traité au sein de l'unité d'oeuvre en cours.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818') Prise en charge du point de synchronisation non disponible.

MQRC_TM_ERREUR
(2265, X'8D9') La structure du message de déclenchement n'est pas valide.

MQRC_TMC_ERREUR
(2191, X'88F') La structure de message du déclencheur de caractères n'est pas valide.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082, X'822') File d'attente de base alias inconnue.

MQRC_UNKNOWN_DEF_XMIT_Q
(2197, X'895') File d'attente de transmission par défaut inconnue.

MQRC_UNKNOWN_NOM_OBJET
(2085, X'825') Nom d'objet inconnu.

MQRC_UNKNOWN_OBJET_Q_MGR
(2086, X'826') Gestionnaire de files d'attente d'objets inconnu.

MQRC_UNKNOWN_REMOTE_Q_MGR
(2087, X'827') Gestionnaire de files d'attente éloignées inconnu.

MQRC_UNKNOWN_XMIT_Q
(2196, X'894') File d'attente de transmission inconnue.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X'932') Echec de l'inscription dans l'unité d'oeuvre globale.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X'933') Mélange d'appels d'unité d'oeuvre non pris en charge.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unité d'oeuvre non disponible pour le gestionnaire de files d'attente à utiliser.

MQRC_WIH_ERREUR
(2333, X'91D') Structure MQWIH non valide.

MQRC_WRONG_CF_NIVEAU
(2366, X'93E') La structure de l'unité de couplage est de niveau incorrect.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Version MQMD fournie non valide.

MQRC_XMIT_Q_TYPE_ERROR
(2091, X'82B') File d'attente de transmission non locale.

MQRC_XMIT_Q_USAGE_ERROR
(2092, X'82C') File d'attente de transmission avec une utilisation incorrecte.

MQRC_XQH_ERREUR
(2260, X'8D4') Structure d'en-tête de file d'attente de transmission non valide.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. Les appels MQPUT et MQPUT1 peuvent être utilisés pour placer des messages dans une file d'attente ; l'appel à utiliser dépend des circonstances:
   - Utilisez l'appel MQPUT pour placer plusieurs messages dans la *même* file d'attente.

     Un appel MQOPEN spécifiant l'option MQOO_OUTPUT est émis en premier, suivi d'une ou de plusieurs demandes MQPUT pour ajouter des messages à la file d'attente ; enfin, la file d'attente est fermée avec un appel MQCLOSE. Cela offre de meilleures performances que l'utilisation répétée de l'appel MQPUT1 .
   - Utilisez l'appel MQPUT1 pour placer *un seul* message dans une file d'attente.

     Cet appel encapsule les appels MQOPEN, MQPUT et MQCLOSE en un seul appel, ce qui réduit le nombre d'appels qui doivent être émis.
2. Si une application place une séquence de messages dans la même file d'attente sans utiliser de groupes de messages,l'ordre de ces messages est préservé sous réserve que certaines conditions soient satisfaites. Toutefois, dans la plupart des environnements, l'appel MQPUT1 ne répond pas à ces conditions et ne préserve donc pas l'ordre des messages. L'appel MQPUT doit être utilisé à la place dans ces environnements. Pour plus de détails, voir Remarques sur l'utilisation de MQPUT.
3. L'appel MQPUT1 peut être utilisé pour insérer des messages dans des listes de distribution. Pour des informations générales à ce sujet, voir les remarques sur la syntaxe des appels MQOPEN et MQPUT.

   Les listes de distribution sont prises en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que les clients WebSphere MQ connectés à ces systèmes.

   Les différences suivantes s'appliquent lors de l'utilisation de l'appel MQPUT1 :

   a. Si l'application fournit des enregistrements de réponse MQRR, ils doivent être fournis à l'aide de la structure MQOD ; ils ne peuvent pas être fournis à l'aide de la structure MQPMO.

   b. Le code anomalie MQRC_OPEN_FAILED n'est jamais renvoyé par MQPUT1 dans les enregistrements de réponse ; si l'ouverture d'une file d'attente échoue, l'enregistrement de réponse de cette file d'attente contient le code anomalie résultant de l'opération d'ouverture.

      Si une opération d'ouverture d'une file d'attente aboutit avec le code achèvement MQCC_WARNING, le code achèvement et le code anomalie de l'enregistrement de réponse de cette file d'attente sont remplacés par les codes achèvement et anomalie résultant de l'opération d'insertion.

      Comme pour les appels MQOPEN et MQPUT, le gestionnaire de files d'attente définit les enregistrements de réponse (s'ils sont fournis) uniquement lorsque le résultat de l'appel n'est pas le même pour toutes les files d'attente de la liste de distribution ; ceci est indiqué par l'exécution de l'appel avec le code anomalie MQRC_MULTIPLE_MOTIFS.
4. Si l'appel MQPUT1 est utilisé pour insérer un message dans une file d'attente de cluster, l'appel se comporte comme si MQOO_BIND_NOT_FIXED avait été spécifié sur l'appel MQOPEN.
5. Si un message est inséré avec une ou plusieurs structures d'en-tête WebSphere MQ au début des données de message d'application, le gestionnaire de files d'attente effectue certaines vérifications sur les structures d'en-tête pour vérifier qu'elles sont valides. Pour plus d'informations à ce sujet, voir les remarques sur l'utilisation de l'appel MQPUT.
6. Si plusieurs situations d'avertissement se produisent (voir le paramètre *CompCode*), le code anomalie renvoyé est le *premier* de la liste suivante qui s'applique:

   a. MQRC_MULTIPLE_MOTIFS

   b. MQRC_INCOMPLETE_MSG

   c. MQRC_INCOMPLETE_GROUP

   d. MQRC_PRIORITY_DÉPASSS_MAXIMUM ou MQRC_UNKNOWN_REPORT_OPTION

7. Pour le langage de programmation Visual Basic, les points suivants s'appliquent :
   - Si la taille du paramètre *Buffer* est inférieure à la longueur spécifiée par le paramètre *BufferLength* , l'appel échoue avec le code anomalie MQRC_BUFFER_LENGTH_ERROR.
   - Le paramètre *Buffer* est déclaré comme étant de type `String`. Si les données à placer dans la file d'attente ne sont pas de type `String`, utilisez laMQPUT1Any à la place de MQPUT1.

     L'appel MQPUT1Any possède les mêmes paramètres que l'appel MQPUT1 , sauf que le paramètre *Buffer* est déclaré comme étant de type Any, ce qui permet de placer n'importe quel type de données dans la file d'attente. Cependant, cela signifie que le système ne peut pas vérifier que le paramètre *Buffer* comporte au minimum *BufferLength* octets.
8. Lorsqu'un appel MQPUT1 est émis avec MQPMO_SYNCPOINT, le comportement par défaut change, de sorte que l'opération d'insertion est exécutée de manière asynchrone. Cela peut entraîner une modification du comportement de certaines applications qui s'appuient sur certaines zones des structures MQOD et MQMD renvoyées, mais qui contiennent désormais des valeurs non définies. Une application peut spécifier MQPMO_SYNC_RESPONSE pour s'assurer que l'opération d'insertion est effectuée de manière synchrone et que toutes les valeurs de zone appropriées sont terminées.

Appel C

```
MQPUT1 (Hconn, &ObjDesc, &MsgDesc, &PutMsgOpts,
       BufferLength, Buffer, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
     CALL 'MQPUT1' USING HCONN, OBJDESC, MSGDESC, PUTMSGOPTS,
                         BUFFERLENGTH, BUFFER, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQPUT1 (Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,
            CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
         CALL MQPUT1,(HCONN,OBJDESC,MSGDESC,PUTMSGOPTS,BUFFERLENGTH,   X
               BUFFER,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Appel Visual Basic

```
MQPUT1 Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,
       CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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-Définition des attributs d'objet

Utilisez l'appel MQSET pour modifier les attributs d'un objet représenté par un descripteur. L'objet doit être une file d'attente.

Syntaxe

MQSET (*Hconn*, *Hobj*, *SelectorCount*, *Sélecteurs*, *IntAttrCount*, *IntAttrs*, *CharAttrLength*, *CharAttrs*, *Compcode*, *Motif*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

Hobj
Type : MQHOBJ - entrée

Cet identificateur représente l'objet de file d'attente avec les attributs à définir. Le descripteur a été renvoyé par un appel MQOPEN précédent qui a spécifié l'option MQOO_SET.

SelectorCount
Type : MQLONG - entrée

Nombre de sélecteurs fournis dans le tableau *Selectors* . Il s'agit du nombre d'attributs à définir. Zéro est une valeur valide. Le nombre maximal autorisé est 256.

Sélecteurs
Type: MQLONGxSelectorNombre-entrée

Il s'agit d'un tableau de sélecteurs d'attribut *SelectorCount* ; chaque sélecteur identifie un attribut (entier ou caractère) avec une valeur à définir.

Chaque sélecteur doit être valide pour le type de file d'attente représenté par *Hobj* . Seules certaines valeurs MQIA_ * et MQCA_ * sont autorisées, comme indiqué ultérieurement.

Les sélecteurs peuvent être spécifiés dans n'importe quel ordre. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut d'entier (sélecteurs MQIA_ *) doivent être spécifiées dans *IntAttrs* dans l'ordre dans lequel ces sélecteurs apparaissent dans *Selectors*. Les valeurs d'attribut qui correspondent à des sélecteurs d'attribut de caractères (sélecteurs MQCA_ *) doivent être spécifiées dans *CharAttrs* dans l'ordre dans lequel ces sélecteurs apparaissent. Les sélecteurs MQIA_ * peuvent être imbriqués avec les sélecteurs MQCA_ * ; seul l'ordre relatif au sein de chaque type est important.

Vous pouvez spécifier le même sélecteur plusieurs fois ; si vous le faites, la dernière valeur spécifiée pour un sélecteur particulier est celle qui prend effet.

Remarque :

1. Les sélecteurs d'attribut d'entier et de caractère sont alloués dans deux plages différentes ; les sélecteurs MQIA_ * résident dans la plage MQIA_FIRST à MQIA_LAST et les sélecteurs MQCA_ * dans la plage MQCA_FIRST à MQCA_LAST.

   Pour chaque plage, les constantes MQIA_LAST_USED et MQCA_LAST_USED définissent la valeur la plus élevée acceptée par le gestionnaire de files d'attente.
2. Si tous les sélecteurs MQIA_ * apparaissent en premier, les mêmes numéros d'élément peuvent être utilisés pour adresser les éléments correspondants dans les tableaux *Selectors* et *IntAttrs* .
3. Si le paramètre *SelectorCount* est égal à zéro, *Selectors* n'est pas référencé ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en langage C ou en langage System/390 assembleur peut être null.

Les attributs qui peuvent être définis sont répertoriés dans le tableau suivant. Aucun autre attribut ne peut être défini à l'aide de cet appel. Pour les sélecteurs d'attribut MQCA_ *, la constante qui définit la longueur en octets de la chaîne requise dans *CharAttrs* est fournie entre parenthèses.

Tableau 571. Sélecteurs d'attributs MQSET pour les files d'attente

Sélecteur	Description	Remarque
DONNEES_TRIGGER_MQCA	Données de déclenchement (MQ_TRIGGER_DATA_LENGTH).	
MQIA_DIST_LISTS	Prise en charge de la liste de distribution.	1
MQIA_INHIBIT_GET	Indique si les opérations d'extraction sont autorisées.	
MQIA_INHIBIT_PUT	Indique si les opérations d'insertion sont autorisées.	
MQIA_TRIGGER_CONTROL	Contrôle du déclencheur.	
PROFONDEUR-MQIA_TRIGGER_PROFONDEUR	Longueur de déclenchement.	
MQIA_TRIGGER_MSG_PRIORITY	Priorité de message de seuil pour les déclencheurs.	
TYPE_TRIGGER_MQIA_	Type de déclencheur.	

Remarque :

1. Pris en charge uniquement sur les clients AIX, HP-UX, IBM i, Solaris, Linux, Windows, ainsi que sur les clients WebSphere MQ MQI connectés à ces systèmes.

IntAttrCount
Type : MQLONG - entrée

Il s'agit du nombre d'éléments dans le tableau *IntAttrs* et doit être au moins égal au nombre de sélecteurs MQIA_ * dans le paramètre *Selectors* . Zéro est une valeur valide s'il n'y en a pas.

IntAttrs
Type: MQLONGxIntAttrCount -entrée

Il s'agit d'un tableau de valeurs d'attribut d'entier *IntAttrCount* . Ces valeurs d'attribut doivent être dans le même ordre que les sélecteurs MQIA_ * dans le tableau *Selectors* .

Si le paramètre *IntAttrCount* ou *SelectorCount* a pour valeur zéro, il n'est pas fait référence à *IntAttrs* ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en C ou en assembleur System/390 peut être null.

CharAttrLongueur
Type : MQLONG - entrée

Il s'agit de la longueur en octets du paramètre *CharAttrs* , qui doit être au moins la somme des longueurs des attributs de caractères spécifiés dans le tableau *Selectors* . Zéro est une valeur valide s'il n'existe pas de sélecteurs MQCA_ * dans *Selectors*.

CharAttrs
Type: MQCHARxCharAttrLength -entrée

Il s'agit de la mémoire tampon contenant les valeurs d'attribut de caractère, concaténées. La longueur de la mémoire tampon est indiquée par le paramètre *CharAttrLength* .

Les attributs de caractères doivent être spécifiés dans le même ordre que les sélecteurs MQCA_ * dans le tableau *Selectors* . La longueur de chaque attribut de caractère est fixe (voir *Selectors*). Si la valeur à définir pour un attribut contient moins de caractères non blancs que la longueur définie de l'attribut, remplissez la valeur dans *CharAttrs* à droite avec des blancs pour que la valeur de l'attribut corresponde à la longueur définie de l'attribut.

Si le paramètre *CharAttrLength* ou *SelectorCount* a pour valeur zéro, il n'est pas fait référence à *CharAttrs* ; dans ce cas, l'adresse de paramètre transmise par les programmes écrits en C ou en assembleur System/390 peut être null.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_API_EXIT_ERROR
(2374, X'946') Echec de l'exit API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CF_STRUC_FAILED
(2373, X'945') Echec de la structure de l'unité de couplage.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Structure de l'unité de couplage en cours d'utilisation.

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347, X'92B') En-tête de liste de la structure d'unité de couplage en cours d'utilisation.

MQRC_CHAR_ATTR_LENGTH_ERROR
(2006, X'7D6') Longueur des attributs de caractère non valide.

MQRC_CHAR_ATTRS_ERREUR
(2007, X'7D7') Chaîne d'attributs de caractères non valide.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Demande d'attente rejetée par CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Non autorisé pour la connexion.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_DB2_NOT_AVAILABLE
(2342, X'926') Sous-système DB2 non disponible.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_HOBJ_ERROR
(2019, X'7E3') Descripteur d'objet non valide.

MQRC_INHIBIT_VALEUR_ERREUR
(2020, X'7E4') Valeur de l'attribut de file d'attente d'interdiction d'extraction ou d'interdiction d'insertion non valide.

MQRC_INT_ATTR_COUNT_ERROR
(2021, X'7E5') Nombre d'attributs de type entier non valide.

MQRC_INT_ATTRS_ARRAY_ERREUR
(2023, X'7E7') Tableau d'attributs d'entier non valide.

MQRC_NOT_OPEN_FOR_SET
(2040, X'7F8') File d'attente non ouverte pour l'ensemble.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Définition d'objet modifiée depuis son ouverture.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objet endommagé.

MQRC_PAGESET_ERROR
(2193, X'891') Erreur d'accès au fichier de l'ensemble de pages.

MQRC_Q_DELETED
(2052, X'804') File d'attente supprimée.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Nom de gestionnaire de files d'attente incorrect ou inconnu.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Gestionnaire de files d'attente non disponible pour la connexion.

MQRC_Q_MGR_STOPPING
(2162, X'872') Gestionnaire de files d'attente en cours d'arrêt.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SELECTOR_COUNT_ERREUR
(2065, X'811') Nombre de sélecteurs non valide.

MQRC_SELECTOR_ERREUR
(2067, X'813') Sélecteur d'attribut non valide.

MQRC_SELECTOR_LIMIT_EXCEEDED
(2066, X'812') Nombre de sélecteurs trop grand.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Appel supprimé par le programme d'exit.

MQRC_TRIGGER_CONTROL_ERROR
(2075, X'81B') Valeur de l'attribut trigger-control non valide.

MQRC_TRIGGER_DEPTH_ERREUR
(2076, X'81C') Valeur de l'attribut trigger-depth non valide.

MQRC_TRIGGER_MSG_PRIORITY_ERR
(2077, X'81D') Valeur de l'attribut trigger-message-priority non valide.

MQRC_TRIGGER_TYPE_ERREUR
(2078, X'81E') Valeur de l'attribut de type de déclencheur non valide.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. A l'aide de cet appel, l'application peut spécifier un tableau d'attributs entiers ou une collection de chaînes d'attributs de caractères, ou les deux. Si aucune erreur ne se produit, les attributs spécifiés sont tous définis simultanément. Si une erreur se produit (par exemple, si un sélecteur n'est pas valide ou qu'une tentative est effectuée pour définir un attribut sur une valeur non valide), l'appel échoue et aucun attribut n'est défini.
2. Les valeurs des attributs peuvent être déterminées à l'aide de l'appel MQINQ ; voir «MQINQ-Attributs d'objet Inquire», à la page 691 pour plus de détails.

   Remarque : Les valeurs des attributs dont les valeurs peuvent être obtenues à l'aide de l'appel MQINQ ne peuvent pas toutes être modifiées à l'aide de l'appel MQSET. Par exemple, aucun attribut process-object ou queue-manager ne peut être défini avec cet appel.
3. Les modifications d'attribut sont conservées lors des redémarrages du gestionnaire de files d'attente (autres que les modifications apportées aux files d'attente dynamiques temporaires qui ne survivent pas aux redémarrages du gestionnaire de files d'attente).
4. Vous ne pouvez pas modifier les attributs d'une file d'attente modèle à l'aide de l'appel MQSET. Toutefois, si vous ouvrez une file d'attente modèle à l'aide de l'appel MQOPEN avec l'option MQOO_SET, vous pouvez utiliser l'appel MQSET pour définir les attributs de la file d'attente locale dynamique créée par l'appel MQOPEN.
5. Si l'objet défini est une file d'attente de cluster, il doit y avoir une instance locale de la file d'attente de cluster pour que l'ouverture aboutisse.

Pour plus d'informations sur les attributs d'objet, voir:

- «Attributs des files d'attente», à la page 824
- «Attributs des listes de noms», à la page 857
- «Attributs des définitions de processus», à la page 859
- «Attributs du gestionnaire de files d'attente», à la page 787

Appel C

```
MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQSET' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,
                   INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,
                   CHARATTRS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
           IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQSET,(HCONN,HOBJ,SELECTORCOUNT,SELECTORS,INTATTRCOUNT,  X
             INTATTRS,CHARATTRLENGTH,CHARATTRS,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Appel Visual Basic

```
MQSET Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, CompCode, Reason
```

Déclarez les paramètres comme suit :

```
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-Définition de la propriété de message

Utilisez l'appel MQSET pour définir ou modifier une propriété d'un descripteur de message.

Syntaxe

MQSETMP (*Hconn*, *Hmsg*, *SetPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *Compcode*, *Reason*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente.

La valeur doit correspondre au descripteur de connexion utilisé pour créer le descripteur de message spécifié dans le paramètre *Hmsg* . Si le descripteur de message a été créé à l'aide de MQHC_UNASSOCIATED_HCONN, une connexion valide doit être établie sur l'unité d'exécution définissant une propriété du descripteur de message, sinon l'appel échoue avec le code anomalie MQRC_CONNECTION_BROKEN.

Msg
Type: MQHMSG-entrée

Il s'agit du descripteur de message à modifier. La valeur a été renvoyée par un appel MQCRTMH précédent.

SetPropOpts
Type: MQSMPO-entrée

Contrôle la façon dont les propriétés de message sont définies.

Cette structure permet aux applications de spécifier des options qui contrôlent la manière dont les propriétés de message sont définies. La structure est un paramètre d'entrée dans l'appel MQSETMP. Pour plus d'informations, voir MQSMPO .

Nom
Type: MQCHARV-entrée

Il s'agit du nom de la propriété à définir.

Pour plus d'informations sur l'utilisation des noms de propriété, voir Noms de propriété et Restrictions relatives aux noms de propriété .

PropDesc
Type: MQPD-entrée/sortie

Cette structure est utilisée pour définir les attributs d'une propriété, notamment:

- que se passe-t-il si la propriété n'est pas prise en charge
- quel est le contexte de message auquel appartient la propriété
- les messages dans lequel la propriété est copiée au fur et à mesure qu'elle est transmise

Pour plus d'informations sur cette structure, voir MQPD .

type
Type : MQLONG - entrée

Type de données de la propriété en cours de définition. Il peut s'agir de l'un des éléments suivants :

MQTYPE_BOOLÉEN
Une valeur booléenne. *ValueLength* doit être 4.

MQTYPE_BYTE_STRING
Chaîne d'octets. *ValueLength* doit être supérieur ou égal à zéro.

MQTYPE_INT8
Entier signé de 8 bits. *ValueLength* doit être 1.

MQTYPE_INT16
Entier signé 16 bits. *ValueLength* doit être 2.

MQTYPE_INT32
Entier signé 32 bits.*ValueLength* doit être 4.

MQTYPE_INT64
Entier signé 64 bits.*ValueLength* doit être 8.

MQTYPE_FLOAT32
Nombre à virgule flottante 32 bits. *ValueLength* doit être 4.

Remarque: ce type n'est pas pris en charge avec les applications utilisant IBM COBOL for z/OS.

MQTYPE_FLOAT64
Nombre en virgule flottante 64 bits. *ValueLength* doit être 8.

Remarque: ce type n'est pas pris en charge avec les applications utilisant IBM COBOL for z/OS.

MQTYPE_CHAINE
Chaîne de caractères. *ValueLength* doit être supérieur ou égal à zéro ou la valeur spéciale MQVL_NULL_TERMINATED.

MQTYPE_NULL
La propriété existe mais a une valeur null.*ValueLength* doit être égale à zéro.

ValueLength
Type : MQLONG - entrée

Longueur en octets de la valeur de propriété dans le paramètre *Valeur* . La valeur zéro est valide uniquement pour les valeurs null ou pour les chaînes ou les chaînes d'octets. La valeur zéro indique que la propriété existe mais que la valeur ne contient ni caractères ni octets.

La valeur doit être supérieure ou égale à zéro ou à la valeur spéciale suivante si MQTYPE_STRING est défini pour le paramètre *Type* :

MQVL_NULL_TERMINATED
La valeur est délimitée par la première valeur nulle rencontrée dans la chaîne. La valeur null n'est pas incluse dans la chaîne. Cette valeur n'est pas valide si MQTYPE_STRING n'est pas également défini.

Remarque: Le caractère null utilisé pour terminer une chaîne si MQVL_NULL_TERMINATED est défini est une valeur null du jeu de caractères de la valeur.

valeur
Type: MQBYTExValueLength-input

Valeur de la propriété à définir. La mémoire tampon doit être alignée sur une limite appropriée à la nature des données de la valeur.

Dans le langage de programmation C, le paramètre est déclaré comme un pointeur vers vide ; l'adresse de tout type de données peut être spécifiée comme paramètre.

Si *ValueLength* est égal à zéro, *Valeur* n'est pas référencé. Dans ce cas, l'adresse de paramètre transmise par les programmes écrits en assembleur C ou System/390 peut être null.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_WARNING:

MQRC_RFH_FORMAT_ERREUR
(2421, X'0975') Un dossier MQRFH2 contenant des propriétés n'a pas pu être analysé.

Si *CompCode* est MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Carte non disponible.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Impossible de charger le module de service d'adaptateur.

MQRC_ASID_MISMATCH
(2157, X'86D') Les ID espace adresse principaux et de base sont différents.

MQRC_BUFFER_ERROR
(2004, X'07D4') Paramètre de valeur non valide.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Paramètre de longueur de valeur non valide.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Appel MQI entré avant la fin de l'appel précédent.

ERREUR MQRC_HMSG_ERROR
(2460, X'099C') Le pointeur de descripteur de message n'est pas valide.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Descripteur de message déjà utilisé.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Options non valides ou non cohérentes.

MQRC_PD_ERREUR
(2482, X'09B2') Structure de descripteur de propriété non valide.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Nom de propriété non valide.

MQRC_PROPERTY_TYPE_ERROR
(2473, X'09A9') Type de données de propriété non valide.

MQRC_PROP_NUMBER_FORMAT_ERROR
(2472, X'09A8') Erreur de format numérique détectée dans les données de valeur.

MQRC_SMPO_ERREUR
(2463, X'099F') La structure des options de propriété de message n'est pas valide.

MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identificateur de jeu de caractères codés de nom de propriété non valide.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Appel C

```
MQSETMP (Hconn, Hmsg, &SetPropOpts, &Name, &PropDesc, Type,
ValueLength, &Value, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
     CALL 'MQSETMP' USING HCONN, HMSG, SETMSGOPTS, NAME, PROPDESC, TYPE,
                          VALUELENGTH, VALUE, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQSETMP (Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength,
              Value, CompCode, Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
        CALL MQSETMP,(HCONN,HMSG,SETMSGHOPTS,NAME,PROPDESC,TYPE,VALUELENGTH,
              VALUE,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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-Extraction des informations de statut

Utilisez l'appel MQSTAT pour extraire les informations de statut. Le type des informations de statut renvoyées est déterminé par la valeur Type indiquée dans l'appel.

Syntaxe

MQSTAT (*Hconn*, *Type*, *Stat* , *Code achèvement*, *Raison*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn* :

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

type
Type : MQLONG - entrée

Type d'informations de statut demandées. Les valeurs valides sont les suivantes:

MQSTAT_TYPE_ASYNC_ERROR
Renvoie des informations sur les opérations d'insertion asynchrone précédentes.

MQSTAT_TYPE_RECONNEXION

Renvoie des informations sur la reconnexion. Si la connexion se reconnecte ou ne parvient pas à se reconnecter, les informations décrivent l'échec qui a provoqué la reconnexion de la connexion.

Cette valeur n'est valide que pour les connexions client. Pour les autres types de connexion, l'appel échoue avec le code anomalie **MQRC_ENVIRONMENT_ERROR**

MQSTAT_TYPE_RECONNECTION_ERROR

Renvoie des informations sur un incident précédent lié à la reconnexion. Si la reconnexion a échoué, les informations décrivent l'échec qui a provoqué l'échec de la reconnexion.

Cette valeur n'est valide que pour les connexions client. Pour les autres types de connexion, l'appel échoue avec le code anomalie **MQRC_ENVIRONMENT_ERROR** .

Stat

Type: MQSTS-entrée/sortie

Structure des informations de statut. Voir «MQSTS-Structure de génération de rapports de statut», à la page 573 pour plus de détails.

CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison

Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_API_EXIT_ERROR
(2374, X' 946') Echec de l'exit API

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Impossible de charger l'exit API.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Appel MQI entré avant la fin de l'appel précédent.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Perte de la connexion au gestionnaire de files d'attente.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Connexion en cours d'arrêt.

MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_Q_MGR_STOPPING
(2162,X'872'-Arrêt du gestionnaire de files d'attente

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_STAT_TYPE_ERREUR
(2430, X'97E'Erreur avec le type MQSTAT

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_STS_ERREUR
(2426, X'97A') Erreur avec la structure MQSTS

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. Un appel à MQSTAT spécifiant un type de MQSTAT_TYPE_ASYNC_ERROR renvoie des informations sur les opérations précédentes asynchrones MQPUT et MQPUT1 . La structure MQSTS retransmise lors du retour de l'appel MQSTAT contient les premières informations d'avertissement ou d'erreur asynchrone enregistrées pour cette connexion. Si d'autres erreurs ou avertissements suivent le premier, ils ne modifient normalement pas ces valeurs. Toutefois, si une erreur se produit avec le code achèvement MQCC_WARNING, un échec ultérieur avec le code achèvement MQCC_FAILED est renvoyé à la place.
2. Si aucune erreur ne s'est produite depuis l'établissement de la connexion ou depuis le dernier appel à MQSTAT , un CompCode de MQCC_OK et une raison de MQRC_NONE sont renvoyés dans la structure MQSTS .
3. Le nombre d'appels asynchrones traités sous le descripteur de connexion est renvoyé par l'intermédiaire de trois zones de compteur: PutSuccessCount, PutWarningCount et PutFailureCount. Ces compteurs sont incrémentés par le gestionnaire de files d'attente chaque fois qu'une opération asynchrone est traitée avec succès, a un avertissement ou échoue (notez qu'à des fins de comptabilité, une insertion dans une liste de distribution compte une fois par file d'attente de destination plutôt qu'une fois par liste de distribution). Un compteur n'est pas incrémenté au-delà de la valeur positive maximale AMQ_LONG_MAX.
4. Un appel réussi à MQSTAT entraîne la réinitialisation des informations d'erreur ou des comptages précédents.
5. Le comportement de MQSTAT dépend de la valeur du paramètre MQSTAT Type que vous indiquez.
6. **MQSTAT_TYPE_ASYNC_ERROR**
    a. Un appel à MQSTAT spécifiant un type de MQSTAT_TYPE_ASYNC_ERROR renvoie des informations sur les opérations précédentes asynchrones MQPUT et MQPUT1 . La structure MQSTS retransmise lors du retour de l'appel MQSTAT contient les premières informations d'avertissement ou d'erreur asynchrone enregistrées pour cette connexion. Si d'autres erreurs ou avertissements suivent le premier, ils ne modifient normalement pas ces valeurs. Toutefois, si une erreur se produit avec le code achèvement MQCC_WARNING, un échec ultérieur avec le code achèvement MQCC_FAILED est renvoyé à la place.
    b. Si aucune erreur ne s'est produite depuis l'établissement de la connexion ou depuis le dernier appel à MQSTAT , un CompCode de MQCC_OK et une raison de MQRC_NONE sont renvoyés dans la structure MQSTS .
    c. Le nombre d'appels asynchrones traités sous le descripteur de connexion est renvoyé par l'intermédiaire de trois zones de compteur: PutSuccessCount, PutWarningCount et PutFailureCount. Ces compteurs sont incrémentés par le gestionnaire de files d'attente chaque fois qu'une opération asynchrone est traitée avec succès, a un avertissement ou échoue (notez qu'à des fins de comptabilité, une insertion dans une liste de distribution compte une fois par file d'attente de destination plutôt qu'une fois par liste de distribution). Un compteur n'est pas incrémenté au-delà de la valeur positive maximale AMQ_LONG_MAX.
    d. Un appel réussi à MQSTAT entraîne la réinitialisation des informations d'erreur ou des comptages précédents.

MQSTAT_TYPE_RECONNEXION

Supposons que vous appelez MQSTAT avec Type défini sur MQSTAT_TYPE_RECONNECTION dans un gestionnaire d'événements lors de la reconnexion. Tenez compte de ces exemples.

Le client tente de se reconnecter ou n'a pas réussi à se reconnecter.
CompCode dans la structure MQSTS est MQCC_FAILED et Reason peut être MQRC_CONNECTION_BROKEN ou MQRC_Q_MGR_QUIESCING . ObjectType est MQOT_Q_MGR, ObjectName est le nom du gestionnaire de files d'attente et ObjectQMgrName est vide.

La reconnexion du client a abouti ou n'a jamais été déconnectée.
CompCode dans la structure MQSTS est MQCC_OK et Reason est MQRC_NONE

Les appels ultérieurs à MQSTAT renvoient les mêmes résultats.

MQSTAT_TYPE_RECONNECTION_ERROR

Supposons que vous appelez MQSTAT avec Type défini sur MQSTAT_TYPE_RECONNECTION_ERROR en réponse à la réception de MQRC_RECONNECT_FAILED dans un appel MQI. Tenez compte de ces exemples.

Une erreur d'autorisation s'est produite lors de la réouverture d'une file d'attente lors de la reconnexion à un autre gestionnaire de files d'attente.
CompCode dans la structure MQSTS est MQCC_FAILED et Reason est la raison pour laquelle la reconnexion a échoué, par exemple MQRC_NOT_AUTHORIZED . ObjectType est le type d'objet à l'origine du problème, par exemple MQOT_QUEUE, ObjectName est le nom de la file d'attente et ObjectQMgrName le nom du gestionnaire de files d'attente propriétaire de la file d'attente.

Une erreur de connexion socket s'est produite lors de la reconnexion.
CompCode dans la structure MQSTS est MQCC_FAILED et Reason est la raison pour laquelle la reconnexion a échoué, par exemple MQRC_HOST_NOT_AVAILABLE . ObjectType est MQOT_Q_MGR, ObjectName est le nom du gestionnaire de files d'attente et ObjectQMgrName est vide.

Les appels ultérieurs à MQSTAT renvoient les mêmes résultats.

Appel C

```
MQSTAT (Hconn, StatType, &Stat, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN Hconn;        /* Connection Handle */
MQLONG StatType;    /* Status type */
MQSTS Stat;            /* Status information structure */
MQLONG CompCode;    /* Completion code */
MQLONG Reason;       /* Reason code qualifying CompCode */
```

Appel COBOL

```
CALL 'MQSTAT' USING HCONN, STATTYPE, STAT, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**    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.
```

Appel PL/I

```
call MQSTAT (Hconn, StatType, Stat, Compcode, Reason);
```

Déclarez les paramètres comme suit :

```
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 Appel de l'assembleur

```
CALL MQSTAT,(HCONN,STATTYPE,STAT,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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-Enregistrement d'abonnement

Utilisez l'appel MQSUB pour enregistrer l'abonnement des applications à une rubrique particulière.

Syntaxe

MQSUB (*Hconn*, *SubDesc*, *Hobj*, *Hsub*, *Code achèvement* , *Motif*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn* :

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

SubDesc
Type: MQSD-entrée/sortie

Il s'agit d'une structure qui identifie l'objet en cours d'utilisation qui est enregistré par l'application. Pour plus d'informations, voir «MQSD - Descripteur d'abonnement», à la page 546.

Hobj
Type: MQHOBJ-entrée / sortie

Ce descripteur représente l'accès qui a été établi pour obtenir les messages envoyés à cet abonnement. Ces messages peuvent être stockés dans une file d'attente spécifique ou le gestionnaire de files d'attente peut gérer leur stockage sans utiliser de file d'attente spécifique.

Pour utiliser une file d'attente spécifique, vous devez l'associer à l'abonnement lors de sa création. Vous pouvez effectuer cette opération de deux manières :

- En utilisant la commande DEFINE SUB MQSC et en indiquant cette commande avec le nom d'un objet file d'attente.
- En fournissant ce descripteur lors de l'appel de MQSUB avec MQSO_CREATE

  Si ce descripteur est fourni en tant que paramètre d'entrée dans l'appel, il doit s'agir d'un descripteur d'objet valide renvoyé par un appel MQOPEN précédent d'une file d'attente à l'aide d'au moins l'une des options suivantes:

– MQOO_ENTRÉE_ *
– MQOO_BROWSE
– MQOO_OUTPUT (si la file d'attente est une file d'attente éloignée)

Si tel n'est pas le cas, l'appel échoue avec MQRC_HOBJ_ERROR. Il ne peut pas s'agir d'un descripteur d'objet dans une file d'attente alias qui se résout en un objet de rubrique. Si tel est le cas, l'appel échoue avec MQRC_HOBJ_ERROR.

Si le gestionnaire de files d'attente doit gérer le stockage des messages envoyés à cet abonnement, il doit être défini lors de la création de l'abonnement, à l'aide de l'option MQSO_MANAGED. Le gestionnaire de files d'attente renvoie ensuite cet identificateur en tant que paramètre de sortie sur l'appel. Le descripteur renvoyé est appelé descripteur géré. Si MQHO_NONE est spécifié mais que MQSO_MANAGED n'est pas spécifié, l'appel échoue avec MQRC_HOBJ_ERROR.

Lorsqu'un descripteur géré vous est renvoyé par le gestionnaire de files d'attente, vous pouvez l'utiliser sur un appel MQGET ou MQCB avec ou sans options de navigation, sur un appel MQINQ ou sur MQCLOSE. Vous ne pouvez pas l'utiliser sur MQPUT, MQSUB, MQSET ; la tentative échoue avec MQRC_NOT_OPEN_FOR_OUTPUT, MQRC_HOBJ_ERROR ou MQRC_NOT_OPEN_FOR_SET.

Si cet abonnement est repris à l'aide de l'option MQSO_RESUME dans la structure MQSD, le descripteur peut être renvoyé à l'application dans ce paramètre en définissant MQSO_MANAGED sur MQHO_NONE. Vous pouvez le faire, que l'abonnement utilise un descripteur géré ou non, et il peut être utile de fournir des abonnements créés à l'aide de DEFINE SUB avec le descripteur de la file d'attente d'abonnement définie sur cette commande. Dans le cas où un abonnement créé par l'administrateur est repris, la file d'attente s'ouvre avec MQOO_INPUT_AS_Q_DEF et MQOO_BROWSE. Si vous devez spécifier d'autres options, l'application doit ouvrir la file d'attente d'abonnement de manière explicite et fournir le descripteur d'objet lors de l'appel. En cas de problème lors de l'ouverture de la file d'attente, l'appel échoue avec MQRC_INVALID_DESTINATION. Si *Hobj* est fourni, il doit être équivalent à *Hobj* dans l'appel MQSUB d'origine. Cela signifie que si un descripteur d'objet renvoyé par un appel MQOPEN est fourni, il doit être placé dans la même file d'attente que celle utilisée précédemment. S'il ne s'agit pas de la même file d'attente, l'appel échoue avec MQRC_HOBJ_ERROR.

Si cet abonnement est modifié à l'aide de l'option MQSO_ALTER dans la structure MQSD, un *Hobj* différent peut être fourni. Toutes les publications qui ont été distribuées à la file d'attente et qui ont été précédemment identifiées à l'aide de ce paramètre restent dans cette file d'attente et il incombe à l'application d'extraire ces messages si le paramètre *Hobj* représente désormais une file d'attente différente.

Le tableau récapitule l'utilisation de ce paramètre avec différentes options d'abonnement:

Options	***Hobj***	**Description**
MQSO_CREATE + MQSO_MANAGED	Ignoré en entrée	Crée un abonnement avec le stockage des messages gérés par le gestionnaire de files d'attente
MQSO_CREER	Un descripteur d'objet valide	Crée un abonnement fournissant une file d'attente spécifique comme destination des messages.
MQSO_RESUME	MQHO_AUCUN	Reprend un abonnement précédemment créé, qu'il ait été géré ou non, et demande au gestionnaire de files d'attente de renvoyer le descripteur d'objet à utiliser par l'application.

Options	*Hobj*	Description
MQSO_RESUME	Descripteur d'objet de correspondance valide	Reprend un abonnement créé précédemment qui utilise une file d'attente spécifique comme destination des messages et utilise un descripteur d'objet avec des options d'ouverture spécifiques.
MQSO_ALTER + MQSO_MANAGED	MQHO_AUCUN	Modifie un abonnement existant qui utilisait auparavant une file d'attente spécifique. Il s'agit donc désormais d'un abonnement géré. La classe de destination (gérée ou non) ne peut pas être modifiée.
MQSO_ALTER	Un descripteur d'objet valide	Modifie un abonnement existant, qu'il ait été géré ou non, de sorte qu'il utilise désormais une file d'attente spécifique. Lorsque l'option MQSO_MANAGED n'est pas utilisée, la file d'attente fournie peut être modifiée, mais la classe de destination (gérée ou non) ne peut pas être modifiée.

Qu'il ait été fourni ou renvoyé, *Hobj* doit être spécifié lors des appels MQGET ou MQCB suivants qui souhaitent recevoir les messages de publication envoyés à cet abonnement.

Le descripteur *Hobj* n'est plus valide lorsque l'appel MQCLOSE est émis dessus ou lorsque l'unité de traitement qui définit la portée du descripteur s'arrête (jusqu'à ce que l'application se déconnecte). La portée du descripteur d'objet renvoyé est identique à celle du descripteur de connexion spécifié dans l'appel. Pour plus d'informations sur la portée de descripteur, voir Hconn (MQHCONN)-output . Une exception MQCLOSE de l'identificateur *Hobj* n'affecte pas l'identificateur *Hsub* .

HSub

Type: MQHOBJ-sortie

Ce descripteur représente l'abonnement qui a été effectué. Il peut être utilisé pour deux autres opérations:

- Il peut être utilisé lors d'un appel MQSUBRQ ultérieur pour demander l'envoi de publications lorsque l'option MQSO_PUBLICATIONS_ON_REQUEST a été utilisée lors de la création de l'abonnement.
- Il peut être utilisé lors d'un appel MQCLOSE ultérieur pour supprimer l'abonnement qui a été effectué. Le descripteur *Hsub* cesse d'être valide lorsque l'appel MQCLOSE est émis ou lorsque l'unité de traitement qui définit la portée du descripteur s'arrête. La portée du descripteur d'objet renvoyé est identique à celle du descripteur de connexion spécifié dans l'appel. Une exception MQCLOSE de l'identificateur *Hsub* n'affecte pas l'identificateur *Hobj* .

Ce descripteur ne peut pas être transmis à un appel MQGET ou MQCB. Vous devez utiliser le paramètre *Hobj* . Vous ne pouvez pas utiliser ce descripteur sur un appel WebSphere MQ autre que MQCLOSE ou MQSUBRQ. La transmission de ce descripteur à tout autre appel WebSphere MQ génère une erreur MQRC_HOBJ_ERROR.

CompCode

Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi

MQCC_WARNING
Avertissement (achèvement partiel)

MQCC_FAILED
Echec de l'appel

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK, le code anomalie est le suivant:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED, le code anomalie est l'un des suivants:

Erreur de résolution MQRC_CLUSTER_RESOLUTION_ERROR
(2189, X'88D') La résolution du nom de cluster a échoué.

MQRC_DURABILITY_NOT_ALLOWED
2436 (X'0984') Un appel MQSUB utilisant l'option MQSO_DURABLE a échoué.

MQRC_FUNCTION_NOT_SUPPORTED
2298 (X'08FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

MQRC_HOBJ_ERROR
2019 (X'07E3') Descripteur d'objet Hobj non valide.

MQRC_IDENTITY_MISMATCH
2434 (X'0982') Le nom de l'abonnement correspond à l'abonnement existant.

MQRC_NOT_AUTHORIZED
2035 (X'07F3') L'utilisateur n'est pas autorisé à effectuer l'opération.

MQRC_OBJECT_STRING_ERROR
2441 (X'0989') Zone Objectstring non valide.

MQRC_OPTIONS_ERROR
2046 (X'07FE') Le paramètre ou la zone Options contient des options non valides ou une combinaison d'options non valides.

MQRC_Q_MGR_QUIESCING
2161 (X'0871') Mise au repos du gestionnaire de files d'attente.

MQRC_RECONNECT_Q_MGR_REQD
2555 (X'09FB'X) L'option MQCNO_RECONNECT_Q_MGR est requise.

MQRC_RETAINED_MSG_Q_ERREUR
2525 (X'09DD') Les publications conservées qui existent pour la chaîne de rubrique souscrite ne peuvent pas être extraites.

MQRC_RETAINED_NOT_LIVRÉ
2526 (X'09DE') Les publications conservées qui existent pour la chaîne de rubrique souscrite ne peuvent pas être distribuées à la file d'attente de destination de l'abonnement et ne peuvent pas être distribuées à la file d'attente de rebut.

MQRC_SD_ERREUR
2424 (X'0978') Descripteur d'abonnement (MQSD) non valide.

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') La chaîne de sélection ne suit pas la syntaxe du sélecteur WebSphere MQ et aucun fournisseur de sélection de message étendu n'est disponible.

MQRC_SELECTION_STRING_ERROR
2519 (X'09D7') La chaîne de sélection doit être spécifiée comme décrit dans la documentation de la structure MQCHARV.

MQRC_SELECTOR_SYNTAX_ERREUR
2459 (X'099B') Un appel MQOPEN, MQPUT1ou MQSUB a été émis, mais une chaîne de sélection contenant une erreur de syntaxe a été spécifiée.

MQRC_SUB_USER_DATA_ERROR
2431 (X'097F') Zone de données SubUsernon valide.

MQRC_SUB_NAME_ERREUR
2440 (X'0988') Zone SubName non valide.

MQRC_SUB_ALREADY_EXISTS
2432 (X'0980') L'abonnement existe déjà.

MQRC_SUB_USER_DATA_ERROR
2431 (X'097F') Zone de données SubUsernon valide.

MQRC_TOPIC_STRING_ERROR
2425 (X'0979') La chaîne de rubrique n'est pas valide.

MQRC_UNKNOWN_NOM_OBJET
2085 (X'0825') L'objet identifié est introuvable.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

1. L'abonnement est effectué à une rubrique, nommée à l'aide du nom abrégé d'un objet de rubrique prédéfini, du nom complet de la chaîne de rubrique ou de la concaténation de deux parties. Voir la description de *ObjectName* et *ObjectString* dans «MQSD - Descripteur d'abonnement», à la page 546.
2. Le gestionnaire de files d'attente effectue des contrôles de sécurité lorsqu'un appel MQSUB est émis pour vérifier que l'ID utilisateur sous lequel l'application s'exécute dispose du niveau de droits approprié avant que l'accès ne soit autorisé. L'objet de rubrique approprié se trouve dans la hiérarchie de rubriques et une vérification des droits d'accès est effectuée sur cet objet de rubrique pour s'assurer que les droits d'abonnement sont définis. Si l'option MQSO_MANAGED n'est pas utilisée, une vérification des droits d'accès est effectuée sur la file d'attente de destination pour s'assurer que les droits d'accès à la sortie sont définis. Si l'option MQSO_MANAGED est utilisée, aucune vérification des droits d'accès n'est effectuée sur la file d'attente gérée pour la sortie ou l'accès en interrogation.
3. Si vous ne fournissez pas d'entrée Hobj, l'appel MQSUB alloue deux descripteurs, un descripteur d'objet (Hobj) et un descripteur d'abonnement (Hsub).
4. Le Hobj renvoyé dans l'appel MQSUB lorsque l'option MQSO_MANAGED est utilisée, peut être interrogé afin de trouver des attributs tels que le seuil d'annulation et le nom de remise en file d'attente d'annulation excessive. Vous pouvez également demander le nom de la file d'attente gérée, mais vous ne devez pas tenter d'ouvrir directement cette file d'attente.
5. Les abonnements peuvent être regroupés, ce qui permet de distribuer une seule publication au groupe d'abonnements même si plusieurs groupes correspondent à la publication. Les abonnements sont regroupés à l'aide de l'option MQSO_GROUP_SUB et, pour pouvoir regrouper les abonnements, ils doivent être
   - utilisation de la même file d'attente nommée (qui n'utilise pas l'option MQSO_MANAGED) sur le même gestionnaire de files d'attente-représenté par le paramètre Hobj de l'appel MQSUB
   - partager le même ID SubCorrel
   - être du même SubLevel

   Ces attributs définissent l'ensemble des abonnements considérés comme faisant partie du groupe et sont également les attributs qui ne peuvent pas être modifiés si un abonnement est groupé. L'altération de SubLevel entraîne MQRC_SUBLEVEL_NOT_ALTERABLE, et l'altération de l'un des autres (qui peut être modifié si un abonnement n'est pas groupé) entraîne MQRC_GROUPING_NOT_ALTERABLE.
6. Les zones du MQSD sont renseignées en cas de retour d'un appel MQSUB qui utilise l'option MQSO_RESUME. Le MQSD renvoyé peut être transmis directement dans un appel MQSUB qui utilise

l'option MQSO_ALTER avec les modifications que vous devez apporter à l'abonnement appliqué au MQSD. Certaines zones comportent des considérations spéciales, comme indiqué dans le tableau.

Sortie MQSD de MQSUB	
Nom de zone dans MQSD	**Des considérations spéciales**
Options d'accès ou de création	Certaines options peuvent être réinitialisées en cas de retour à partir de l'appel MQSUB. Si vous réutilisez ensuite le MQSD dans un appel MQSUB, l'option requise doit être explicitement définie.
Options de durabilité, Options de destination, Options d'enregistrement et Options de caractère générique	Ces options sont définies comme il convient
Options de publication	Ces options sont définies comme il convient, à l'exception de MQSO_NEW_PUBLICATIONS_ONLY, qui s'applique uniquement à MQSO_CREATE.
Autres options	Ces options sont inchangées en cas de retour d'un appel MQSUB. Ils contrôlent la façon dont l'appel d'API est émis et ne sont pas stockés avec l'abonnement. Ils doivent être définis comme requis lors de tout appel MQSUB ultérieur réutilisant le MQSD.
ObjectName	Cette zone d'entrée uniquement n'est pas modifiée en cas de retour d'un appel MQSUB.
ObjectString	Cette zone d'entrée uniquement n'est pas modifiée en cas de retour d'un appel MQSUB. Le nom de rubrique complet utilisé est renvoyé dans la zone *ResObjectString* , si une mémoire tampon est fournie.
ID AlternateUseret ID AlternateSecurity	Ces zones d'entrée uniquement sont inchangées en cas de retour d'un appel MQSUB. Ils contrôlent la façon dont l'appel d'API est émis et ne sont pas stockés avec l'abonnement. Ils doivent être définis comme requis lors de tout appel MQSUB ultérieur réutilisant le MQSD.
SubExpiry	En cas de retour d'un appel MQSUB à l'aide de l'option MQSO_RESUME, cette zone est définie sur l'expiration d'origine de l'abonnement et non sur l'heure d'expiration restante. Si vous réutilisez ensuite le MQSD dans un appel MQSUB à l'aide de l'option MQSO_ALTER, vous réinitialisez l'expiration de l'abonnement pour recommencer le décomptage.
SubName	Cette zone est une zone d'entrée dans un appel MQSUB et n'est pas modifiée dans la sortie.

Sortie MQSD de MQSUB *(suite)*	
Nom de zone dans MQSD	**Des considérations spéciales**
SubUserData et SelectionString	Ces zones de longueur variable sont renvoyées dans la sortie d'un appel MQSUB à l'aide de l'option MQSO_RESUME, si une mémoire tampon est fournie, ainsi qu'une longueur de mémoire tampon positive dans *VSBufSize*. Si aucune mémoire tampon n'est fournie, seule la longueur est renvoyée dans la zone *VSLength* de MQCHARV. Si la mémoire tampon fournie est inférieure à l'espace requis pour renvoyer la zone, seuls *VSBufSize* octets sont renvoyés dans la mémoire tampon fournie. Si vous réutilisez ensuite le MQSD dans un appel MQSUB à l'aide de l'option MQSO_ALTER et qu'une mémoire tampon n'est pas fournie, mais qu'une valeur *VSLength* différente de zéro est fournie, si cette longueur correspond à la longueur existante de la zone, aucune modification n'est apportée à la zone.
ID SubCorrelet jeton PubAccounting	Si vous n'utilisez pas MQSO_SET_CORREL_ID, *SubCorrelId* est généré par le gestionnaire de files d'attente. Si vous n'utilisez pas MQSO_SET_IDENTITY_CONTEXT, le *PubAccountingToken* est généré par le gestionnaire de files d'attente. Ces zones sont renvoyées dans le MQSD à partir d'un appel MQSUB à l'aide de l'option MQSO_RESUME. Si elles sont générées par le gestionnaire de files d'attente, la valeur générée est renvoyée dans un appel MQSUB à l'aide de l'option MQSO_CREATE ou MQSO_ALTER.
PubPriority, SubLevel & PubApplIdentityData	Ces zones sont renvoyées dans le MQSD.
Chaîne ResObject	Cette zone de sortie uniquement est renvoyée dans le MQSD si une mémoire tampon est fournie.

Appel C

```
MQSUB (Hconn, &SubDesc, &Hobj, &Hsub, &CompCode, &Reason)
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQSUB' USING HCONN, SUBDESC, HOBJ, HSUB, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Appel PL/I

```
call MQSUB (Hconn, SubDesc, Hobj, Hsub, CompCode, Reason)
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```

CALL MQSUB,(HCONN,SUBDESC,HOBJ,HSUB,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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-Demande d'abonnement

Utilisez l'appel MQSUBRQ pour demander la publication conservée lorsque l'abonné a été enregistré auprès de MQSO_PUBLICATIONS_ON_REQUEST.

Syntaxe

MQSUBRQ (*Hconn*, *Hsub*, *Action*, *SubRqOpts*, *Code achèvement*, *Motif*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente. La valeur de *Hconn* a été renvoyée par un appel MQCONN ou MQCONNX précédent.

Sur les applications z/OS for CICS et sur IBM i pour les applications s'exécutant en mode compatibilité, l'appel MQCONN peut être omis et la valeur suivante peut être spécifiée pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

HSub
Type : MQHOBJ - entrée

Ce descripteur représente l'abonnement pour lequel une mise à jour doit être demandée. La valeur de *Hsub* a été renvoyée à partir d'un appel MQSUB précédent.

Action
Type : MQLONG - entrée

Ce paramètre contrôle l'action particulière demandée sur l'abonnement. La valeur suivante doit être spécifiée:

PUBLICATION MQSR_ACTION_PUBLICATION

Cette action demande qu'une publication de mise à jour soit envoyée pour la rubrique spécifiée. Il ne peut être utilisé que si l'abonné a spécifié l'option MQSO_PUBLICATIONS_ON_REQUEST sur l'appel MQSUB lorsqu'il a effectué l'abonnement. Si le gestionnaire de files d'attente possède une publication conservée pour la rubrique, celle-ci est envoyée à l'abonné. Si ce n'est pas le cas, l'appel échoue. Si une application reçoit une publication qui a été conservée, elle est indiquée par la propriété de message MQIsRetained de cette publication.

Etant donné que la rubrique de l'abonnement existant représenté par le paramètre Hsub peut contenir des caractères génériques, l'abonné peut recevoir plusieurs publications conservées.

SubRqOpts
Type: MQSRO-entrée / sortie

Ces options contrôlent l'action de MQSUBRQ. Pour plus de détails, voir «MQSRO-Options de demande d'abonnement», à la page 570 .

Si aucune option n'est requise, les programmes écrits en langage C ou en langage S/390 assembleur peuvent spécifier une adresse de paramètre null au lieu de spécifier l'adresse d'une structure MQSRO.

CompCode
Type : MQLONG - sortie

Code achèvement ; il s'agit d'un des codes suivants :

MQCC_OK
Achèvement réussi

MQCC_WARNING
Avertissement (achèvement partiel)

MQCC_FAILED
Echec de l'appel

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_FUNCTION_NOT_SUPPORTED
2298 (X'08FA') La fonction demandée n'est pas disponible dans l'environnement en cours.

MQRC_NO_RETAINED_MSG
2437 (X'0985') Aucune publication conservée n'est actuellement stockée pour cette rubrique.

MQRC_OPTIONS_ERROR
2046 (X'07FE') Le paramètre ou la zone Options contient des options non valides ou une combinaison d'options non valides.

MQRC_Q_MGR_QUIESCING
2161 (X'0871') Mise au repos du gestionnaire de files d'attente.

MQRC_SRO_ERREUR
2438 (X'0986') Sur l'appel MQSUBRQ, les options de demande d'abonnement MQSRO ne sont pas valides.

MQRC_RETAINED_MSG_Q_ERREUR
2525 (X'09DD') Les publications conservées qui existent pour la chaîne de rubrique souscrite ne peuvent pas être extraites.

MQRC_RETAINED_NOT_LIVRÉ
2526 (X'09DE') Les publications conservées qui existent pour la chaîne de rubrique souscrite ne peuvent pas être distribuées à la file d'attente de destination de l'abonnement et ne peuvent pas être distribuées à la file d'attente de rebut.

Pour plus de détails sur ces codes, voir Codes anomalie.

Notes d'utilisation

Les remarques de syntaxe suivantes s'appliquent à l'utilisation du code d'action MQSR_ACTION_PUBLICATION:

1. Si cette instruction aboutit, les publications conservées correspondant à l'abonnement spécifié ont été envoyées à l'abonnement et peuvent être reçues à l'aide de MQGET ou de MQCB à l'aide du Hobj renvoyé dans l'instruction MQSUB d'origine qui a créé l'abonnement.
2. Si la rubrique à laquelle est abonné l'instruction MQSUB d'origine qui a créé l'abonnement contient un caractère générique, plusieurs publications conservées peuvent être envoyées. Le nombre de publications envoyées à la suite de cet appel est enregistré dans la zone NumPubs de la structure SubRqOpts.
3. Si cette instruction se termine avec le code anomalie MQRC_NO_RETAINED_MSG, cela signifie qu'il n'y a pas de publications actuellement conservées pour la rubrique spécifiée. #
4. Si cette instruction se termine avec le code anomalie MQRC_RETAINED_MSG_Q_ERROR ou MQRC_RETAINED_NOT_LIVRÉES, des publications sont actuellement conservées pour la rubrique spécifiée, mais une erreur s'est produite indiquant qu'elles n'ont pas pu être distribuées.
5. L'application doit disposer d'un abonnement en cours à la rubrique pour pouvoir effectuer cet appel. Si l'abonnement a été effectué dans une instance précédente de l'application et qu'un descripteur valide de l'abonnement n'est pas disponible, l'application doit d'abord appeler MQSUB avec l'option MQSO_RESUME afin d'obtenir un descripteur pour l'utiliser dans cet appel.
6. Les publications sont envoyées à la destination enregistrée pour être utilisée avec l'abonnement en cours de cette application. Si les publications doivent être envoyées ailleurs, l'abonnement doit d'abord être modifié à l'aide de l'appel MQSUB avec l'option MQSO_ALTER.

Appel C

```
MQSUB (Hconn, Hsub, Action, &SubRqOpts, &CompCode, &Reason)
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel COBOL

```
CALL 'MQSUBRQ' USING HCONN, HSUB, ACTION, SUBRQOPTS, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
** 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.
```

Appel PL/I

```
call MQSUBRQ (Hconn, Hsub, Action, SubRqOpts, CompCode, Reason)
```

Déclarez les paramètres comme suit :

```
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 */
```

Appel d'assembleur de haut niveau

```
CALL MQSUBRQ,(HCONN, HSUB, ACTION, SUBRQOPTS,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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
```

Attributs des objets

Cette collection de rubriques répertorie uniquement les objets WebSphere MQ qui peuvent faire l'objet d'un appel de fonction MQINQ et fournit des détails sur les attributs pouvant être recherchés et les sélecteurs à utiliser.

Attributs du gestionnaire de files d'attente

Certains attributs de gestionnaire de files d'attente sont fixes pour des implémentations particulières ; d'autres peuvent être modifiés à l'aide de la commande MQSC ALTER QMGR.

Les attributs peuvent également être affichés à l'aide de la commande DISPLAY QMGR. La plupart des attributs du gestionnaire de files d'attente peuvent être renseignés en ouvrant un objet MQOT_Q_MGR spécial et en utilisant l'appel MQINQ avec le descripteur renvoyé.

Le tableau suivant récapitule les attributs spécifiques au gestionnaire de files d'attente. Les attributs sont décrits par ordre alphabétique.

Remarque : Les noms des attributs affichés dans cette section sont des noms descriptifs utilisés avec l'appel MQINQ ; les noms sont identiques à ceux des commandes PCF. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, des noms abrégés alternatifs sont utilisés ; voir Commandes de script (MQSC) pour plus d'informations.

Tableau 572. Attributs du gestionnaire de files d'attente.

Liste des attributs de gestionnaire de files d'attente avec liens et brève description

Attribut	Description
AccountingConnOverride	Remplacer les paramètres de comptabilité.
AccountingInterval	Fréquence d'écriture des enregistrements comptables intermédiaires.
ActivityConnOverride	Remplacer les paramètres d'activité.
ActivityTrace	Contrôle la collecte de la trace d'activité de l'application WebSphere MQ MQI.
AdoptNewMCACheck	Eléments vérifiés pour déterminer s'il convient d'adopter un nouvel agent MCA.
AdoptNewMCAType	Indique s'il faut redémarrer automatiquement une instance orpheline d'un agent MCA d'un type de canal particulier.
AlterationDate	Date de la dernière modification de la définition
AlterationTime	Heure de la dernière modification de la définition
AuthorityEvent	Contrôle si des événements d'autorisation (non autorisés) sont générés
BridgeEvent	Attribut de contrôle pour les événements de pont.
ChannelAutoDef	Contrôle si la définition de canal automatique est autorisée
ChannelAutoDefEvent	Contrôle si des événements de définition automatique de canal sont générés
ChannelAutoDefExit	Nom de l'exit utilisateur pour la définition de canal automatique
ChannelEvent	Attribut de contrôle pour les événements de canal.
ChannelInitiatorControl	Attribut de contrôle de l'initiateur de canal
ChannelMonitoring	Données de surveillance en ligne pour les canaux
ChannelStatistics	Contrôle la collecte des données statistiques pour les canaux.
ChinitAdapters	Nombre de sous-tâches d'adaptateur pour le traitement des appels WebSphere MQ .
ChinitDispatchers	Nombre de répartiteurs à utiliser pour l'initiateur de canal.
	Réservé à IBM .
ChinitTraceAutoStart	Indique si la trace de l'initiateur de canal doit démarrer automatiquement.
ChinitTraceTableSize	Taille de l'espace de données de trace de l'initiateur de canal.
ClusterSenderMonitoringDefault	Valeurs par défaut des données de surveillance en ligne pour les canaux émetteurs de cluster
StatistiquesClusterSender	Contrôle la collecte des informations de surveillance des statistiques pour les canaux émetteurs de cluster.
ClusterWorkloadData	Données utilisateur pour l'exit de charge de travail de cluster
ClusterWorkloadExit	Nom de l'exit utilisateur pour la gestion de la charge de travail du cluster
ClusterWorkloadLength	Longueur maximale des données de message transmises à l'exit de charge de travail du cluster
CLWLMRUChannels	Nombre de canaux les plus récemment utilisés pour l'équilibrage de charge de cluster
CLWLUseQ	La charge de travail du cluster utilise la file d'attente éloignée.
CodedCharSetId	Identificateur de jeu de caractères codés
CommandEvent	Attribut de contrôle pour les événements de commande.
AttributCommandInputQName	Nom de la file d'attente d'entrée des commandes
CommandLevel	Niveau de commande
AttributCommandServerControl	Attribut de contrôle du serveur de commandes.
Attribut Evénement de configuration	Attribut de contrôle pour les événements de configuration.
DeadLetterQName	Nom de la file d'attente de rebut
DEFCLXQ	Type de file d'attente de transmission de cluster par défaut
DefXmitQName	Nom de file d'attente de transmission par défaut
DistLists	Prise en charge de la liste de diffusion
DNSGroup	Nom du groupe pour le programme d'écoute TCP lors de l'utilisation de la prise en charge de Workload Manager Dynamic Domain Name Services.

Tableau 572. Attributs du gestionnaire de files d'attente.

Liste des attributs de gestionnaire de files d'attente avec liens et brève description

(suite)

Attribut	Description
DNSWLM	Indique si le programme d'écoute TCP s'enregistre auprès de Workload Manager pour Dynamic Domain Name Services.
ExpiryInterval	Intervalle entre les analyses des messages arrivés à expiration
IGQPutAuthority	Droit d'insertion de la mise en file d'attente intragroupe
IGQUserId	Identificateur utilisateur de la mise en file d'attente intra-groupe
InhibitEvent	Contrôle si des événements d'inhibition (Inhibition Get et Inhibition Put) sont générés
IPAddressVersion	Version de l'adresse Internet Protocol
IntraGroupQueuing	Prise en charge de la mise en file d'attente intra-groupe
ListenerTimer	Intervalle entre les tentatives de redémarrage du programme d'écoute après un échec APPC ou TCP/IP.
LocalEvent	Contrôle si des événements d'erreur locaux sont générés
LoggerEvent	Contrôle si les événements du consignateur sont générés
LUGroupName	Nom d'unité logique générique pour le programme d'écoute d'unité logique 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente.
nom ul	Nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes.
LU62ARMSuffix	Suffixe de SYS1.PARMLIB APPCPMxx, qui désigne LUADD pour cet initiateur de canal.
LU62Channels	Nombre maximal de canaux en cours ou de clients connectés qui utilisent l'unité logique 6.2.
MaxActiveChannels	Nombre maximal de canaux pouvant être actifs à tout moment.
MaxChannels	Nombre maximal de canaux en cours.
MaxHandles	Nombre maximal de descripteurs
Longueur maximale des messages	Longueur maximale des messages (en octets)
AttributMaxPriority	Priorité maximum
MaxPropertiesLength	Longueur maximale des données de propriété en octets
MaxUncommittedMsgs	Nombre maximal de messages non validés dans une unité de travail
MQIAccounting	Contrôle la collecte des informations de comptabilité pour les données MQI.
MQIStatistics	Contrôle la collecte des informations de surveillance des statistiques pour le gestionnaire de files d'attente.
MsgMarkBrowseInterval	Intervalle après lequel le gestionnaire de files d'attente peut supprimer la marque des messages consultés.
OutboundPortMin	Avec *OutboundPortMin*, définit une plage de numéros de port à utiliser lors de la liaison des canaux sortants.
OutboundPortMin	Avec *OutboundPortMax*, définit une plage de numéros de port à utiliser lors de la liaison des canaux sortants.
PerformanceEvent	Contrôle si les événements liés aux performances sont générés
Plateforme	Plateforme sur laquelle le gestionnaire de files d'attente est en cours d'exécution
PubSubNPInputMsg	Indique s'il faut supprimer (ou conserver) un message d'entrée non distribué
PubSubNPResponse	Contrôle le comportement des commandes non livrées
PubSubMaxMsgRetryCount	Nombre de nouvelles tentatives lors du traitement (sous le point de synchronisation) d'un message de commande ayant échoué
PubSubSyncPoint	Indique si seuls les messages persistants (ou tous les messages) doivent être traités pour un point de synchronisation.
PubSubMode	Indique si l'interface de publication / abonnement en file d'attente est en cours d'exécution
QMgrDesc	Description du gestionnaire de files d'attente
QMgrIdentifier	Identificateur unique généré en interne du gestionnaire de files d'attente
QMgrName	Nom gest. de files
QSGName	Nom du groupe de partage de files d'attente
QueueAccounting	Contrôle la collecte des informations comptables pour les files d'attente.
QueueMonitoring	Données de surveillance en ligne pour les files d'attente

Tableau 572. Attributs du gestionnaire de files d'attente.

Liste des attributs de gestionnaire de files d'attente avec liens et brève description

(suite)

Attribut	Description
QueueStatistics	Contrôle la collecte des données statistiques pour les files d'attente.
ReceiveTimeout	Durée pendant laquelle le canal TCP/IP attend les données avant de revenir à l'état inactif.
ReceiveTimeoutMin	Qualificateur pour *ReceiveTimeout*.
ReceiveTimeoutType	Durée minimale pendant laquelle le canal TCP/IP attend les données avant de revenir à l'état inactif.
RemoteEvent	Contrôle si des événements d'erreur distants sont générés
RepositoryName	Nom du cluster pour lequel ce gestionnaire de files d'attente fournit des services de référentiel
RepositoryNamelist	Nom de l'objet liste de noms contenant les noms des clusters pour lesquels ce gestionnaire de files d'attente fournit des services de référentiel
ScyCase	Cas des profils de sécurité
SharedQMgr	Nom du gestionnaire de files d'attente partagées
«SPLCAP», à la page 820	WebSphere MQ Advanced Protection de la sécurité des messages pour un gestionnaire de files d'attente activé ou désactivé.
SSLCRLNamelist 1	Nom de l'objet liste de noms contenant les noms des objets d'informations d'authentification.
SSLCryptoHardware 1	Chaîne de configuration du matériel de cryptographie.
SSLEvent	Attribut de contrôle pour les événements SSL.
SSLFIPSRequired	Utilisez uniquement des algorithmes certifiés FIPS pour la cryptographie.
SSLKeyRepository 1	Emplacement du référentiel de clés SSL.
SSLKeyResetNombre	Nombre de réinitialisations de clé SSL.
SSLTpose 1	Nombre de sous-tâches de serveur pour le traitement des appels SSL.
StatisticsInterval	Fréquence d'écriture des données de surveillance des statistiques.
StartStopEvent	Contrôle si des événements de démarrage et d'arrêt sont générés
SyncPoint	Prise en charge points sync.
TCPChannels	Nombre maximal de canaux en cours ou de clients connectés qui utilisent TCP/IP.
TCPKeepAlive	Indique si TCP KEEPALIVE doit être utilisé pour vérifier les autres fins de connexion.
TCPName	Nom du système TCP/IP que vous utilisez.
TCPStackType	Comment l'initiateur de canal peut utiliser les adresses TCP/IP.
AttributTraceRouteEnregistrement	Contrôle l'enregistrement des informations de routage de trace.
TriggerInterval	Intervalle entre les messages de déclenchement
Version	Version
XrCapability	Indique si les commandes de télémétrie sont prises en charge.

Remarques :

1. Cet attribut ne peut pas être interrogé à l'aide de l'appel MQINQ et n'est pas décrit dans cette section. Pour plus d'informations sur cet attribut, voir Modification du gestionnaire de files d'attente .

Tâches associées

Comment indiquer que seuls les CipherSpecs certifiés FIPS sont utilisés lors de l'exécution sur MQI Client

Référence associée

FIPS (Federal Information Processing Standards) pour UNIX, Linux et Windows

Remplacement de AccountingConn(MQLONG)

Cela permet aux applications de remplacer le paramètre des valeurs ACCTMQI et ACCTQDATA dans l'attribut Qmgr.

La valeur est l'une des suivantes :

MQMON_DISABLED
Les applications ne peuvent pas remplacer les paramètres des attributs ACCTMQI et ACCTQ Qmgr à l'aide de la zone Options de la structure MQCNO sur l'appel MQCONNX. Il s'agit de la valeur par défaut.

MQMON_ENABLED
Les applications peuvent remplacer les attributs Qmgr ACCTQ et ACCTMQI à l'aide de la zone Options de la structure MQCNO.

Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente après la modification de l'attribut.

Cet attribut est pris en charge uniquement sur les systèmes IBM i, Unix et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ACCOUNTING_CONN_OVERRIDE avec l'appel MQINQ.

AccountingInterval (MQLONG)

Indique la durée d'écriture des enregistrements comptables intermédiaires (en secondes).

La valeur est un entier compris entre 0 et 604800, avec une valeur par défaut de 1800 (30 minutes). Indiquez 0 pour désactiver les enregistrements intermédiaires.

Cet attribut est pris en charge uniquement sur les systèmes IBM i, Windows, UNIXet Linux .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ACCOUNTING_INTERVAL avec l'appel MQINQ.

Substitution ActivityConn(MQLONG)

Cela permet aux applications de remplacer la valeur ACTVTRC dans l'attribut de gestionnaire de files d'attente.

La valeur est l'une des suivantes :

MQMON_DISABLED
Les applications ne peuvent pas remplacer le paramètre de l'attribut de gestionnaire de files d'attente ACTVTRC à l'aide de la zone Options de la structure MQCNO sur l'appel MQCONNX. Il s'agit de la valeur par défaut.

MQMON_ENABLED
Les applications peuvent remplacer l'attribut de gestionnaire de files d'attente ACTVTRC à l'aide de la zone Options de la structure MQCNO.

Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente après la modification de l'attribut.

Cet attribut est pris en charge uniquement sur les systèmes IBM i, Unix et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ACTIVITY_CONN_OVERRIDE avec l'appel MQINQ .

ActivityTrace (MQLONG)

Cette option contrôle la collecte de la trace de l'activité de l'application WebSphere MQ MQI.

La valeur est l'une des suivantes :

MQMON_ON
Collectez la trace de l'activité de l'application WebSphere MQ MQI.

MQMON_OFF
Ne collectez pas de trace d'activité d'application WebSphere MQ MQI. Il s'agit de la valeur par défaut.

Si vous définissez l'attribut de gestionnaire de files d'attente ACTVCON0 sur ENABLED, cette valeur peut être remplacée pour des connexions individuelles à l'aide de la zone Options de la structure MQCNO.

Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente après la modification de l'attribut.

Cet attribut est pris en charge uniquement sur les systèmes IBM i, Unix et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ACTIVITY_TRACE avec l'appel MQINQ .

AdoptNewMCACheck (MQLONG)

Définit les éléments à vérifier pour déterminer s'il convient d'adopter un agent MCA lorsqu'un nouveau canal entrant portant le même nom qu'un agent MCA déjà actif est détecté.

La valeur est l'une des suivantes :

MQADOPT_CHECK_Q_MGR_NAME
Vérifiez le nom du gestionnaire de files d'attente.

MQADOPT_CHECK_NET_ADDR
Vérifiez l'adresse réseau.

MQADOPT_CHECK_ALL
Vérifiez le nom du gestionnaire de files d'attente et l'adresse réseau. Si possible, effectuez cette vérification pour empêcher l'arrêt de vos canaux, par inadvertance ou de manière malveillante. Il s'agit de la valeur par défaut.

MQADOPT_CHECK_NONE
Ne cochez aucun élément.

Les modifications apportées à cet attribut prennent effet la prochaine fois qu'un canal tente d'adopter un canal.

Cet attribut est pris en charge uniquement sur z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ADOPTNEWMCA_CHECK avec l'appel MQINQ.

AdoptNewMCAType (MQLONG)

Indique s'il faut redémarrer automatiquement une instance orpheline d'un agent MCA d'un type de canal particulier lorsqu'une nouvelle demande de canal entrant correspondant à l'attribut MCACheck AdoptNewest détectée

Ses valeurs sont l'une des suivantes :

MQADOPT_TYPE_NO
L'adoption d'instances de canal orphelines n'est pas requise. Il s'agit de la valeur par défaut.

MQADOPT_TYPE_ALL
Adoptez tous les types de canal.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ADOPTNEWMCA_TYPE avec l'appel MQINQ.

AlterationDate (MQCHAR12)

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ALTERATION_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_DATE_LENGTH.

AlterationTime (MQCHAR8)

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ALTERATION_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_TIME_LENGTH.

AuthorityEvent (MQLONG)

Indique si les événements d'autorisation (Non autorisé) sont générés. Ses valeurs sont l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_AUTHORITY_EVENT avec l'appel MQINQ.

BridgeEvent (MQLONG)

Indique si des événements de pont IMS sont générés.

La valeur est l'une des suivantes :

MQEVR_ENABLED
Générez des événements de pont IMS , comme suit:

MQRC_BRIDGE_STARTED
MQRC_BRIDGE_STOPPED

MQEVR_DISABLED
Ne générez pas d'événements de pont IMS ; il s'agit de la valeur par défaut.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_BRIDGE_EVENT avec l'appel MQINQ.

Def ChannelAuto(MQLONG)

Cet attribut contrôle la définition automatique des canaux de type MQCHT_RECEIVER et MQCHT_SVRCONN. La définition automatique des canaux MQCHT_CLUSSDR est toujours activée. La valeur est l'une des suivantes :

MQCHAD_DISABLED
Définition automatique de canal désactivée.

MQCHAD_ENABLED
Définition automatique de canal activée.

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CHANNEL_AUTO_DEF avec l'appel MQINQ.

ChannelAutoDefEvent (MQLONG)

Indique si les événements de définition automatique de canal sont générés. Elle s'applique aux canaux de type MQCHT_RECEIVER, MQCHT_SVRCONN et MQCHT_CLUSSDR. La valeur est l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CHANNEL_AUTO_DEF_EVENT avec l'appel MQINQ.

ChannelAutoDefExit (MQCHARn)

Il s'agit du nom de l'exit utilisateur pour la définition de canal automatique. Si ce nom n'est pas vide et que *ChannelAutoDef* a la valeur MQCHAD_ENABLED, l'exit est appelé chaque fois que le gestionnaire de files d'attente est sur le point de créer une définition de canal. S'applique aux canaux de type MQCHT_RECEIVER, MQCHT_SVRCONN et MQCHT_CLUSSDR. L'exit peut alors effectuer l'une des opérations suivantes:

- Créez la définition de canal sans modification.
- Modifiez les attributs de la définition de canal créée.
- Supprimer complètement la création du canal.

Remarque : La longueur et la valeur de cet attribut sont spécifiques à l'environnement. Voir l'introduction de la structure MQCD dans «MQCD-Définition de canal», à la page 1045 pour plus de détails sur la valeur de cet attribut dans divers environnements.

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris, Windowset z/OS. Sous z/OS, elle s'applique uniquement aux canaux émetteur et récepteur de cluster.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CHANNEL_AUTO_DEF_EXIT avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_EXIT_NAME_LENGTH.

ChannelEvent (MQLONG)

Indique si des événements de canal sont générés.

Ses valeurs sont l'une des suivantes :

MQEVR_EXCEPTION
Générez uniquement les événements de canal suivants:

- MQRC_CHANNEL_ACTIVATED
- MQRC_CHANNEL_CONV_ERROR
- MQRC_CHANNEL_NOT_ACTIVATED
- MQRC_CHANNEL_STOPPED avec les ReasonQualifierssuivants:

  MQRQ_CHANNEL_STOPPED_ERROR
  MQRQ_CHANNEL_STOPPED_RETRY
  MQRQ_CHANNEL_STOPPED_DISABLED

  MQRC_CHANNEL_STOPPED_BY_USER

MQEVR_ENABLED
Générez tous les événements de canal. En d'autres termes, en plus de celles générées par EXCEPTION, générez les événements de canal suivants:

- MQRC_CHANNEL_STARTED
- MQRC_CHANNEL_STOPPED avec l'élément ReasonQualifiersuivant:

  MQRQ_CHANNEL_STOPPED_OK

MQEVR_DISABLED
Ne générez pas d'événements de canal ; il s'agit de la valeur par défaut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CHANNEL_EVENT avec l'appel MQINQ.

Contrôle ChannelInitiator(MQLONG)

Indique si l'initiateur de canal doit être démarré lors du démarrage du gestionnaire de files d'attente.

Ses valeurs sont l'une des suivantes :

MQSVC_CONTROL_MANUAL
L'initiateur de canal ne doit pas être démarré automatiquement.

MQSVC_CONTROL_Q_MGR
L'initiateur de canal doit être démarré automatiquement au démarrage du gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CHINIT_CONTROL avec l'appel MQINQ.

ChannelMonitoring (MQLONG)

Indique les données de surveillance en ligne pour les canaux.

La valeur est l'une des suivantes :

MQMON_AUCUN
Désactivez la collecte de données pour la surveillance des canaux pour tous les canaux, quelle que soit la valeur de l'attribut de canal MONCHL. Il s'agit de la valeur par défaut.

MQMON_OFF
Désactivez la collecte des données de surveillance pour les canaux qui spécifient QMGR dans l'attribut de canal MONCHL.

MQMON_FAIBLE
Activez la collecte de données de surveillance avec un faible taux de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal MONCHL.

MQMON_MEDIUM
Activez la collecte de données de surveillance avec un rapport modéré de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal MONCHL.

MQMON_ELEVE
Activez la collecte de données de surveillance avec un taux élevé de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal MONCHL.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MONITORING_CHANNEL avec l'appel MQINQ.

ChannelStatistics (MQLONG)

Cette option contrôle la collecte des données statistiques pour les canaux.

La valeur est l'une des suivantes :

MQMON_AUCUN
Désactivez la collecte de données pour les statistiques de canal pour tous les canaux, quelle que soit la valeur de l'attribut de canal STATCHL. Il s'agit de la valeur par défaut.

MQMON_OFF
Désactivez la collecte des données statistiques pour les canaux qui spécifient QMGR dans l'attribut de canal STATCHL.

MQMON_FAIBLE
Activez la collecte de données statistiques avec un faible taux de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal STATCHL.

MQMON_MEDIUM
Activez la collecte de données statistiques avec un taux modéré de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal STATCHL.

MQMON_ELEVE
Activez la collecte de données statistiques avec un taux élevé de collecte de données pour les canaux spécifiant QMGR dans l'attribut de canal STATCHL.

Pour la plupart des systèmes, il est recommandé d'utiliser MEDIUM. Toutefois, pour un canal qui traite un volume élevé de messages par seconde, vous pouvez réduire le niveau d'échantillonnage en sélectionnant

LOW. En outre, pour un canal qui ne traite que quelques messages et pour lequel les informations les plus récentes sont importantes, vous pouvez sélectionner HIGH.

Cet attribut est pris en charge uniquement sur les systèmes IBM i, UNIX et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_STATISTICS_CHANNEL avec l'appel MQINQ.

ChinitAdapters (MQLONG)

Il s'agit du nombre de sous-tâches d'adaptateur à utiliser pour traiter les appels WebSphere MQ . La valeur doit être comprise entre 0 et 9999, avec une valeur par défaut de 8.

Le rapport entre les adaptateurs et les répartiteurs (attribut ChinitDispatchers) doit être d'environ 8 à 5. Toutefois, si vous n'avez que quelques canaux, vous n'avez pas besoin de diminuer la valeur de ce paramètre par rapport à la valeur par défaut. Vous pouvez utiliser les valeurs suivantes: pour un système de test, 8 (valeur par défaut) ; pour un système de production, 20. Idéalement, vous devez disposer de 20 adaptateurs, ce qui offre un parallélisme plus important des appels WebSphere MQ . Cette caractéristique est importante pour les messages persistants. Moins d'adaptateurs peuvent être utilisés pour les messages non persistants.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CHINIT_ADAPTERS avec l'appel MQINQ.

ChinitDispatchers (MQLONG)

Nombre de répartiteurs à utiliser pour l'initiateur de canal. La valeur doit être comprise entre 0 et 9999, avec une valeur par défaut de 5.

Comme instruction, autorisez un répartiteur pour 50 canaux en cours. Toutefois, si vous n'avez que quelques canaux, vous n'avez pas besoin de diminuer la valeur de cet attribut par rapport à la valeur par défaut. Si vous utilisez TCP/IP, le plus grand nombre de répartiteurs utilisés pour les canaux TCP/IP est 100, même si vous indiquez une valeur plus élevée ici. Vous pouvez utiliser les paramètres suivants: systèmes de test, 5 (par défaut) ; systèmes de production, 20 (vous avez besoin de 20 répartiteurs pour gérer jusqu'à 1000 canaux actifs).

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CHINIT_DISPATCHERS avec l'appel MQINQ.

ChinitTraceAutoStart (MQLONG)

Indique si la trace de l'initiateur de canal doit être démarrée automatiquement.

La valeur est l'une des suivantes :

MQTRAXSTR_YES
Démarrer automatiquement la trace de l'initialisateur de canal. Il s'agit de la valeur par défaut.

MQTRAXSTR_NO
Ne pas démarrer automatiquement la trace de l'initialisateur de canal.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CHINIT_TRACE_AUTO_START avec l'appel MQINQ.

ChinitTraceTableSize (MQLONG)

Il s'agit de la taille de l'espace de données de trace de l'initiateur de canal (en Mo).

La valeur doit être comprise entre 0 et 2048, avec une valeur par défaut de 2.

Remarque : Chaque fois que vous utilisez des espaces de données z/OS volumineux, vérifiez que vous disposez de suffisamment de mémoire secondaire sur votre système pour prendre en charge toute

activité de pagination z/OS associée. Vous devrez peut-être également augmenter la taille de vos fichiers SYS1.DUMP.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CHINIT_TRACE_TABLE_SIZE avec l'appel MQINQ.

ClusterSenderMonitoringDefault (MQLONG)

Indique la valeur à remplacer par l'attribut ChannelMonitoring des canaux émetteurs de cluster définis automatiquement.

La valeur est l'une des suivantes :

MQMON_Q_DIR
La collecte des données de surveillance en ligne est héritée de la valeur de l'attribut *ChannelMonitoring* du gestionnaire de files d'attente. Il s'agit de la valeur par défaut.

MQMON_OFF
La surveillance du canal est désactivée

MQMON_FAIBLE
A moins que *ChannelMonitoring* ne soit MQMON_NONE, la surveillance est activée avec un faible taux de collecte de données avec un impact minimal sur les performances du système. Il est peu probable que les données collectées soient les plus récentes.

MQMON_MEDIUM
A moins que *ChannelMonitoring* ne soit MQMON_NONE, la surveillance est activée avec un taux modéré de collecte de données avec un effet limité sur les performances du système.

MQMON_ELEVE
A moins que *ChannelMonitoring* ne soit défini sur MQMON_NONE, la surveillance est activée avec un taux élevé de collecte de données, ce qui a probablement un impact sur les performances du système. Les données collectées sont les plus récentes disponibles.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MONITORING_AUTO_CLUSSDR avec l'appel MQINQ.

ClusterSender-Statistiques (MQLONG)

Etant donné que les canaux émetteurs de cluster peuvent être définis automatiquement à partir de la définition de CLUSRCVR dans le référentiel, vous ne pouvez pas modifier le paramètre de l'attribut STATCHL pour ces canaux émetteurs de cluster définis automatiquement à l'aide du canal ALTER. Pour ces canaux, la décision de collecter ou non des données de surveillance en ligne est basée sur la définition de cet attribut de gestionnaire de files d'attente.

La valeur est l'une des suivantes :

MQMON_Q_DIR
La collecte de données statistiques pour les canaux émetteurs de cluster définis automatiquement est basée sur la valeur de l'attribut de gestionnaire de files d'attente STATCHL. Il s'agit de la valeur par défaut.

MQMON_OFF
Désactivez la collecte des données statistiques pour les canaux émetteurs de cluster définis automatiquement.

MQMON_FAIBLE
Activez la collecte de données statistiques pour les canaux émetteurs de cluster définis automatiquement avec un faible taux de collecte de données.

MQMON_MEDIUM
Activez la collecte de données statistiques pour les canaux émetteurs de cluster définis automatiquement avec un taux modéré de collecte de données.

MQMON_ELEVE
Activez la collecte de données statistiques pour les canaux émetteurs de cluster définis automatiquement avec un taux élevé de collecte de données.

Pour la plupart des systèmes, nous recommandons MEDIUM. Toutefois, pour un canal émetteur de cluster défini automatiquement qui traite un volume élevé de messages par seconde, vous pouvez réduire le niveau d'échantillonnage en sélectionnant LOW. En outre, pour un canal qui ne traite que quelques messages et pour lequel les informations les plus récentes sont importantes, vous pouvez sélectionner HIGH.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_STATISTICS_AUTO_CLUSSDR avec l'appel MQINQ.

ClusterWorkload-Données (MQCHAR32)

Il s'agit d'une chaîne de caractères de 32 octets définie par l'utilisateur qui est transmise à l'exit de charge de travail du cluster lorsqu'il est appelé. S'il n'y a pas de données à transmettre à l'exit, la chaîne est vide.

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris, Windows et z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CLUSTER_WORKLOAD_DATA avec l'appel MQINQ.

Exit ClusterWorkload(MQCHARn)

Il s'agit du nom de l'exit utilisateur pour la gestion de la charge de travail du cluster. Si ce nom n'est pas vide, l'exit est appelé chaque fois qu'un message est inséré dans une file d'attente de cluster ou déplacé d'une file d'attente d'émetteur de cluster vers une autre. L'exit peut alors soit accepter l'instance de file d'attente sélectionnée par le gestionnaire de files d'attente comme destination du message, soit sélectionner une autre instance de file d'attente.

Remarque : La longueur et la valeur de cet attribut sont spécifiques à l'environnement.

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris, Windows et z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CLUSTER_WORKLOAD_EXIT avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_EXIT_NAME_LENGTH.

ClusterWorkloadLongueur (MQLONG)

Longueur maximale des données de message transmises à l'exit de charge de travail du cluster. La longueur réelle des données transmises à l'exit est la valeur minimale suivante:

- Longueur du message.
- Attribut *MaxMsgLength* du gestionnaire de files d'attente.
- Attribut *ClusterWorkloadLength* .

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris, Windows et z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CLUSTER_WORKLOAD_LENGTH avec l'appel MQINQ.

CLWLMRUChannels (MQLONG)

Indique le nombre maximal de canaux de cluster les plus récemment utilisés, à utiliser par l'algorithme de choix de charge de travail de cluster.

Il s'agit d'une valeur comprise entre 1 et 999999999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CLWL_MRU_CHANNELS avec l'appel MQINQ.

CLWLUseQ (MQLONG)

Indique si les files d'attente éloignées doivent être utilisées pour la charge de travail du cluster.

La valeur est l'une des suivantes :

MQCLWL_USEQ_ANY
Utilisez les files d'attente locales et distantes.

MQCLWL_USEQ_LOCAL
N'utilisez pas de files d'attente éloignées. Il s'agit de la valeur par défaut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CLWL_USEQ avec l'appel MQINQ.

CodedCharSetId (MQLONG)

Définit le jeu de caractères utilisé par le gestionnaire de files d'attente pour toutes les zones de chaîne de caractères définies dans l'interface MQI, telles que les noms des objets, ainsi que la date et l'heure de création de la file d'attente. Le jeu de caractères doit comporter des caractères mono-octet pour les caractères admis dans les noms d'objet. Elle ne s'applique pas aux données d'application contenues dans le message. La valeur dépend de l'environnement:

- Sous z/OS, la valeur est définie à partir des paramètres système lors du démarrage du gestionnaire de files d'attente ; la valeur par défaut est 500.
- Sous Windows, la valeur est le CODEPAGE principal de l'utilisateur qui crée le gestionnaire de files d'attente.
- Sous IBM i, la valeur est celle qui est définie dans l'environnement lorsque le gestionnaire de files d'attente est créé pour la première fois.
- Sur les systèmes UNIX , la valeur est la valeur par défaut CODESET pour l'environnement local de l'utilisateur qui crée le gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CODED_CHAR_SET_ID avec l'appel MQINQ.

CommandEvent (MQLONG)

Indique si des événements de commande sont générés, comme suit:

MQEVR_DISABLED
Ne générez pas d'événements de commande. Il s'agit de l'option par défaut.

MQEVR_ENABLED
Générez des événements de commande.

MQEVR_NO_DISPLAY
Des événements de commande sont générés pour toutes les commandes ayant abouti autres que MQINQ.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_COMMAND_EVENT avec l'appel MQINQ.

CommandInputQName (MQCHAR48)

Il s'agit du nom de la file d'attente d'entrée de commandes définie sur le gestionnaire de files d'attente local. Il s'agit d'une file d'attente à laquelle les utilisateurs peuvent envoyer des commandes, s'ils y sont autorisés. Le nom de la file d'attente dépend de l'environnement:

- Sous z/OS, le nom de la file d'attente est SYSTEM.COMMAND.INPUT; les commandes MQSC et PCF peuvent lui être envoyées. Voir Les commandes MQSC pour plus de détails sur les commandes MQSC et Définitions des formats de commande programmables pour plus de détails sur les commandes PCF.
- Dans tous les autres environnements, le nom de la file d'attente est SYSTEM.ADMIN.COMMAND.QUEUE, et seules les commandes PCF peuvent lui être envoyées. Toutefois, une commande MQSC peut être envoyée à cette file d'attente si la commande MQSC est incluse dans une commande PCF de type MQCMD_ESCAPE. Pour plus d'informations sur la commande Escape, voir Echap .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_COMMAND_INPUT_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

CommandLevel (MQLONG)

Indique le niveau des commandes de contrôle du système prises en charge par le gestionnaire de files d'attente. Il peut s'agir de l'une des valeurs suivantes:

MQCMDL_LEVEL_1
Niveau 1 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- MQSeries for AIX Version 2 Edition 2
- MQSeries pour
  - Version 1 édition 1.1
  - Version 1 édition 1.2
  - Version 1 édition 1.3
- MQSeries pour OS/400
  - Version 2 édition 3
  - Version 3 édition 1
  - Version 3 édition 6
- MQSeries for Windows version 2 édition 0

MQCMDL_LEVEL_101
MQSeries for Windows version 2 édition 0.1.

MQCMDL_LEVEL_110
MQSeries for Windows version 2 édition 1.

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 version 2 édition 0.

MQCMDL_LEVEL_210
MQSeries pour OS/390 Version 2 Release 1.0.

MQCMDL_LEVEL_220
Niveau 220 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- MQSeries for AT & T GIS UNIX version 2 édition 2
- MQSeries pour SINIX et DC/OSx version 2 édition 2
- MQSeries for SunOS version 2 édition 2
- MQSeries for Tandem NonStop Kernel version 2 édition 2

MQCMDL_LEVEL_221
Niveau 221 des commandes de contrôle du système.

Cette valeur est renvoyée par MQSeries pour AIX Version 2 Release 2.1

MQCMDL_LEVEL_320
Niveau 320 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- MQSeries pour OS/400

  - Version 3 édition 2
  - Version 3 édition 7

MQCMDL_LEVEL_420
Niveau 420 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- MQSeries pour IBM i
  - Version 4 édition 2.0
  - Version 4 édition 2.1

MQCMDL_LEVEL_500
Niveau 500 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- IBM WebSphere MQ for AIX version 5 édition 0
- MQSeries for HP-UX version 5 édition 0
- MQSeries for Solaris version 5 édition 0
- MQSeries for Windows NT Version 5 Edition 0

MQCMDL_LEVEL_510
Niveau 510 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- IBM WebSphere MQ for AIX version 5 édition 1
- MQSeries for AS/400 version 5 édition 1
- MQSeries for HP-UX version 5 édition 1
- IBM WebSphere MQ for HP Integrity NonStop Server version 5 édition 3
- MQSeries for Compaq Tru64 UNIX version 5 édition 1
- MQSeries for Solaris version 5 édition 1
- MQSeries for Windows NT version 5 édition 1

MQCMDL_LEVEL_520
Niveau 520 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- MQSeries for AIX ersion 5 édition 2
- MQSeries for AS/400 Version 5 Edition 2
- MQSeries for HP-UX version 5 édition 2
- MQSeries for Linux version 5 édition 2
- MQSeries for OS/390 Version 5 Edition 2
- MQSeries for Sun Solaris version 5 édition 2
- MQSeries for Windows NT version 5 édition 2

MQCMDL_LEVEL_530
Niveau 530 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- IBM WebSphere MQ for AIX version 5 édition 3
- IBM WebSphere MQ for HP-UX version 5 édition 3
- IBM WebSphere MQ for i / Series version 5 édition 3
- IBM WebSphere MQ for Linux for Intel version 5 édition 3
- IBM WebSphere MQ for Linux for zSeries version 5 édition 3

- IBM WebSphere MQ for Solaris version 5 édition 3
- IBM WebSphere MQ for Windows version 5 édition 3
- IBM WebSphere MQ for z/OS version 5 édition 3

MQCMDL_LEVEL_600
Niveau 600 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- IBM WebSphere MQ for AIX Version 6.0
- IBM WebSphere MQ for HP-UX Version 6.0
- IBM WebSphere MQ pour 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
Niveau 700 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- 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
Niveau 701 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- 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
Niveau 710 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- 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
Niveau 750 des commandes de contrôle du système.

Cette valeur est renvoyée par les versions suivantes de IBM WebSphere MQ:

- 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

L'ensemble de commandes de contrôle du système qui correspond à une valeur particulière de l'attribut *CommandLevel* varie en fonction de la valeur de l'attribut *Platform* . Ces deux commandes doivent être utilisées pour déterminer les commandes de contrôle du système qui sont prises en charge.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_COMMAND_LEVEL avec l'appel MQINQ .

Contrôle CommandServer(MQLONG)
Indique si le serveur de commandes doit être démarré lors du démarrage du gestionnaire de files d'attente.

La valeur peut être :

MQSVC_CONTROL_MANUAL
Le serveur de commandes ne doit pas être démarré automatiquement.

MQSVC_CONTROL_Q_MGR
Le serveur de commandes doit être démarré automatiquement lors du démarrage du gestionnaire de files d'attente.

Cet attribut n'est pas pris en charge sur z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CMD_SERVER_CONTROL avec l'appel MQINQ.

ConfigurationEvent (MQLONG)

Contrôle si les événements de configuration sont générés.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CONFIGURATION_EVENT avec l'appel MQINQ.

La valeur peut être :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

DeadLetterQName (MQCHAR48)

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local en tant que file d'attente de rebut (messages non livrés). Les messages sont envoyés à cette file d'attente s'ils ne peuvent pas être acheminés vers leur destination correcte.

Par exemple, les messages sont placés dans cette file d'attente lorsque:

- Un message arrive sur un gestionnaire de files d'attente, destiné à une file d'attente qui n'est pas encore définie sur ce gestionnaire de files d'attente
- Un message arrive dans un gestionnaire de files d'attente, mais la file d'attente à laquelle il est destiné ne peut pas le recevoir pour la raison suivante:

- La file d'attente est saturée
- Les demandes d'insertion sont interdites
- Le noeud émetteur n'a pas le droit d'insérer des messages dans la file d'attente

Les applications peuvent également placer des messages dans la file d'attente des messages non livrés.

Les messages de rapport sont traités de la même manière que les messages ordinaires ; si le message de rapport ne peut pas être distribué à sa file d'attente de destination (généralement la file d'attente spécifiée par la zone *ReplyToQ* dans le descripteur de message du message d'origine), le message de rapport est placé dans la file d'attente de rebut (message non distribué).

Remarque : Les messages dont l'heure d'expiration est dépassée (voir MQMD-Zone d'expiration) ne sont **pas** transférés dans cette file d'attente lorsqu'ils sont supprimés. Toutefois, un message de rapport d'expiration (MQRO_EXPIRATION) est toujours généré et envoyé à la file d'attente *ReplyToQ* , si l'application émettrice le demande.

Les messages ne sont pas insérés dans la file d'attente des messages non livrés lorsque l'application qui a émis la demande d'insertion a été informée de manière synchrone du problème à l'aide du code anomalie renvoyé par l'appel MQPUT ou MQPUT1 (par exemple, un message inséré dans une file d'attente locale pour laquelle les demandes d'insertion sont interdites).

Les messages de la file d'attente de messages non livrés ont parfois des données de message d'application préfixées avec une structure MQDLH. Cette structure contient des informations supplémentaires qui indiquent la raison pour laquelle le message a été placé dans la file d'attente des messages non livrés. Pour plus d'informations sur cette structure, voir «En-tête MQDLH-Dead-letter», à la page 327 .

Cette file d'attente doit être une file d'attente locale, avec l'attribut *Usage* MQUS_NORMAL.

Si un gestionnaire de files d'attente ne prend pas en charge une file d'attente de rebut (message non distribué) ou si aucune file d'attente n'a été définie, le nom est à blanc. Tous les gestionnaires de files d'attente WebSphere MQ prennent en charge une file d'attente de messages non livrés, mais elle n'est pas définie par défaut.

Si la file d'attente de rebut (messages non livrés) n'est pas définie, saturée ou inutilisable pour une autre raison, un message qui lui aurait été transféré par un agent de canal de transmission est conservé dans la file d'attente de transmission.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_DEAD_LETTER_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

DefClusterXmitQueueType (MQLONG)

Attribut `DefClusterXmitQueue` contrôle la file d'attente de transmission qui est sélectionnée par défaut par les canaux émetteurs de cluster pour extraire les messages et les envoyer aux canaux récepteurs de cluster.

Les valeurs de l'attribut `DefClusterXmitQueueType` sont `MQCLXQ_SCTQ` ou `MQCLXQ_CHANNEL`.

`MQCLXQ_SCTQ`

Tous les canaux émetteurs de cluster envoient des messages à partir de `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. L'ID `correlID` de messages placés dans la file d'attente de transmission identifie le canal émetteur de cluster auquel le message est destiné.

`SCTQ` est défini lorsqu'un gestionnaire de files d'attente est défini. Ce comportement est implicite dans les versions de IBM WebSphere MQ, antérieures à Version 7.5. Dans les versions précédentes, l'attribut de gestionnaire de files d'attente `DefClusterXmitQueueType` n'existait pas.

`MQCLXQ_CHANNEL`

Chaque canal émetteur de cluster envoie des messages à partir d'une file d'attente de transmission différente. Chaque file d'attente de transmission est créée en tant que file d'attente dynamique permanente à partir du modèle de file d'attente `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`.

L'attribut n'est pas pris en charge sur z/OS.

Si l'attribut de gestionnaire de files d'attente DefClusterXmitQueueTypeest défini sur CHANNEL, la configuration par défaut est modifiée pour les canaux émetteurs de cluster associés à des files d'attente de transmission de cluster individuelles. Les files d'attente de transmission sont des files d'attente dynamiques permanentes créées à partir de la file d'attente modèle SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE. Chacune d'elles est associée à un canal émetteur de cluster. Etant donné qu'un canal émetteur de cluster sert une file d'attente de transmission de cluster, cette dernière contient les messages relatifs à un seul gestionnaire de files d'attente d'un seul cluster. Vous pouvez configurer les clusters afin que chaque gestionnaire de files d'attente d'un cluster contienne une seule file d'attente de cluster. Dans le cas, le trafic de messages entre un gestionnaire de files d'attente et chaque file d'attente de cluster est transféré séparément des files d'attente de messages à d'autres files d'attente.

Pour interroger la valeur, appelez MQINQou envoyez une commande Inquire Queue Manager (MQCMD_INQUIRE_Q_MGR) PCF, en définissant le sélecteur MQIA_DEF_CLUSTER_XMIT_Q_TYPE . Pour modifier la valeur, envoyez une commande PCFMQCMD_CHANGE_Q_MGR(Modifier un gestionnaire de files d'attente), en définissant le sélecteur MQIA_DEF_CLUSTER_XMIT_Q_TYPE .

Référence associée

Modifier un gestionnaire de files d'attente

Consulter les gestionnaires de files d'attente

«MQINQ-Attributs d'objet Inquire», à la page 691

L'appel MQINQ renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'un objet.

DefXmitQName (MQCHAR48)

Il s'agit du nom de la file d'attente de transmission utilisée pour la transmission des messages aux gestionnaires de files d'attente éloignées, s'il n'existe aucune autre indication de la file d'attente de transmission à utiliser.

S'il n'existe pas de file d'attente de transmission par défaut, le nom est entièrement vide. La valeur initiale de cet attribut est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_DEF_XMIT_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

DistLists (MQLONG)

Indique si le gestionnaire de files d'attente local prend en charge les listes de distribution dans les appels MQPUT et MQPUT1 . Ses valeurs sont l'une des suivantes :

MQDL_SUPPORTED
Listes de distribution prises en charge.

MQDL_NOT_SUPPORTED
Les listes de distribution ne sont pas prises en charge.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DIST_LISTS avec l'appel MQINQ.

DNSGroup (MQCHAR18)

Il s'agit du nom du groupe du programme d'écoute TCP qui gère les transmissions entrantes du groupe de partage de files d'attente à joindre lors de l'utilisation de la prise en charge des services de nom de domaine dynamique de Workload Manager. Sa longueur maximale est de 18 caractères. Si vous laissez ce nom vide, le nom du groupe de partage de files d'attente est utilisé.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_DNS_GROUP avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_DNS_GROUP_NAME_LENGTH.

DNSWLM (MQLONG)

Indique si le programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente s'enregistre auprès de Workload Manager for Dynamic Domain Name Services

La valeur est l'une des suivantes :

MQDNSWLM_OUI
Le programme d'écoute s'enregistre auprès de Workload Manager.

MQDNSWLM_NO
Le programme d'écoute ne s'enregistre pas auprès du gestionnaire de charge de travail. Il s'agit de la valeur par défaut.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DNS_WLM avec l'appel MQINQ.

ExpiryInterval (MQLONG)

Indique la fréquence à laquelle le gestionnaire de files d'attente analyse les files d'attente à la recherche de messages arrivés à expiration. Il s'agit soit d'un intervalle de temps en secondes compris entre 1 et 99 999 999, soit de la valeur spéciale suivante:

MQEXPI_OFF
Le gestionnaire de files d'attente n'analyse pas les files d'attente à la recherche de messages arrivés à expiration.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_EXPIRY_INTERVAL avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sur z/OS.

IGQPutAuthority (MQLONG)

Cet attribut s'applique uniquement si le gestionnaire de files d'attente local est membre d'un groupe de partage de files d'attente. Il indique le type de vérification des droits d'accès effectuée lorsque l'agent de mise en file d'attente intra-groupe local (agent IGQ) supprime un message de la file d'attente de transmission partagée et place le message dans une file d'attente locale. La valeur est l'une des suivantes :

MQIGQPA_PAR DEFAUT
L'identificateur utilisateur vérifié pour l'autorisation est la valeur de la zone *UserIdentifier* dans le MQMD *distinct* associé au message lorsque le message se trouve dans la file d'attente de transmission partagée. Il s'agit de l'ID utilisateur du programme qui a placé le message dans la file d'attente de transmission partagée et qui est généralement identique à l'ID utilisateur sous lequel le gestionnaire de files d'attente éloignées s'exécute.

Si le profil RESLEVEL indique que plusieurs identificateurs d'utilisateur doivent être vérifiés, l'identificateur d'utilisateur de l'agent IGQ local (*IGQUserId*) est également vérifié.

MQIGQPA_CONTEXT
L'identificateur utilisateur vérifié pour l'autorisation est la valeur de la zone *UserIdentifier* dans le MQMD *distinct* associé au message lorsque le message se trouve dans la file d'attente de transmission partagée. Il s'agit de l'ID utilisateur du programme qui a placé le message dans la file d'attente de transmission partagée et qui est généralement identique à l'ID utilisateur sous lequel le gestionnaire de files d'attente éloignées s'exécute.

Si le profil RESLEVEL indique que plusieurs identificateurs d'utilisateur doivent être vérifiés, l'identificateur d'utilisateur de l'agent IGQ local (*IGQUserId*) et la valeur de la zone *UserIdentifier* dans le MQMD *intégré* sont également vérifiés. Ce dernier identifiant d'utilisateur est habituellement l'identifiant d'utilisateur de l'application à l'origine du message.

MQIGQPA_ONLY_IGQ
L'ID utilisateur vérifié pour l'autorisation est l'ID utilisateur de l'agent IGQ local (*IGQUserId*).

Si le profil RESLEVEL indique que plusieurs ID utilisateur doivent être vérifiés, cet ID utilisateur est utilisé pour toutes les vérifications.

MQIGQPA_ALTERNATE_OR_IGQ
L'ID utilisateur vérifié pour l'autorisation est l'ID utilisateur de l'agent IGQ local (*IGQUserId*).

Si le profil RESLEVEL indique que plusieurs ID utilisateur doivent être vérifiés, la valeur de la zone *UserIdentifier* dans le MQMD *imbriqué* est également vérifiée. Il s'agit généralement de l'ID utilisateur de l'application à l'origine du message.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_IGQ_PUT_AUTHORITY avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sur z/OS.

IGQUserId (MQLONG)

Cet attribut est applicable uniquement si le gestionnaire de files d'attente locales appartient à un groupe de partage de files d'attente. Il indique l'ID utilisateur associé à l'agent de mise en file d'attente intra-groupe local (agent de mise en file d'attente intra-groupe). Il s'agit de l'un des ID utilisateur dont l'autorisation peut être vérifiée lorsque l'agent de mise en file d'attente intra-groupe insère des messages dans des files d'attente locales. Les identificateurs d'utilisateur réels vérifiés dépendent de la définition de l'attribut *IGQPutAuthority* et des options de sécurité externes.

Si *IGQUserId* est vide, aucun identificateur d'utilisateur n'est associé à l'agent IGQ et la vérification d'autorisation correspondante n'est pas effectuée (bien que d'autres identificateurs d'utilisateur puissent encore être vérifiés pour l'autorisation).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_IGQ_USER_ID avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_USER_ID_LENGTH.

Cet attribut est pris en charge uniquement sur z/OS.

InhibitEvent (MQLONG)

Cette option contrôle si des événements d'inhibition (Inhibition Get et Inhibition Put) sont générés. La valeur est l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_INHIBIT_EVENT avec l'appel MQINQ.

Sous z/OS, vous ne pouvez pas utiliser l'appel MQINQ pour déterminer la valeur de cet attribut.

Mise en file d'attente IntraGroup(MQLONG)

Cet attribut s'applique uniquement si le gestionnaire de files d'attente local est membre d'un groupe de partage de files d'attente. Indique si la mise en file d'attente intra-groupe est activée pour le groupe de partage de files d'attente. La valeur est l'une des suivantes :

MQIGQ_DISABLED
Tous les messages destinés aux autres gestionnaires de files d'attente du groupe de partage de files d'attente sont transmis à l'aide de canaux conventionnels.

MQIGQ_ENABLED
Les messages destinés aux autres gestionnaires de files d'attente du groupe de partage de files d'attente sont transmis à l'aide de la file d'attente de transmission partagée si la condition suivante est remplie:

- La longueur des données de message et de l'en-tête de transmission ne dépasse pas 63 Ko (64 512 octets).

  Il est recommandé d'allouer un peu plus d'espace que la taille de MQXQH pour l'en-tête de transmission ; la constante MQ_MSG_HEADER_LENGTH est fournie à cet effet.

Si cette condition n'est pas remplie, le message est transmis à l'aide de canaux classiques.

Remarque : Lorsque la mise en file d'attente intra-groupe est activée, l'ordre des messages transmis à l'aide de la file d'attente de transmission partagée n'est pas conservé par rapport à ceux transmis à l'aide des canaux classiques.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_INTRA_GROUP_QUEUING avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sur z/OS.

IPAddressVersion (MQLONG)

Indique quelle version d'adresse IP, IPv4 ou IPv6, est utilisée.

Cet attribut est pertinent uniquement pour les systèmes qui exécutent à la fois IPv4 et IPv6 et affecte uniquement les canaux définis comme ayant un *TransportType* de MQXPY_TCP lorsque l'une des conditions suivantes est remplie:

- Le *ConnectionName* du canal est un nom d'hôte qui se résout à la fois en une adresse IPv4 et IPv6 et son paramètre *LocalAddress* n'est pas spécifié.
- Les noms d'hôte *ConnectionName* et *LocalAddress* du canal sont tous deux des noms d'hôte qui se résolvent en adresses IPv4 et IPv6 .

La valeur peut être :

MQIPADDR_IPV4
IPv4 est utilisé.

MQIPADDR_IPV6
IPv6 est utilisé.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_IP_ADDRESS_VERSION avec l'appel MQINQ.

ListenerTimer (MQLONG)

Il s'agit de l'intervalle (en secondes) entre les tentatives de redémarrage du programme d'écoute par WebSphere MQ en cas d'échec APPC ou TCP/IP. La valeur doit être comprise entre 5 et 9999, avec une valeur par défaut de 60.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_LISTENER_TIMER avec l'appel MQINQ.

LocalEvent (MQLONG)

Cette option contrôle si des événements d'erreur locaux sont générés. La valeur est l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_LOCAL_EVENT avec l'appel MQINQ.

Sous z/OS, vous ne pouvez pas utiliser l'appel MQINQ pour déterminer la valeur de cet attribut.

LoggerEvent (MQLONG)

Cette option contrôle si les événements du journal de reprise sont générés. La valeur est l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_LOGGER_EVENT avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris et Windows.

LUGroupName (MQCHAR8)

Il s'agit du nom de LU générique du programme d'écoute de LU 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente. Si vous laissez ce nom vide, vous ne pouvez pas utiliser ce programme d'écoute.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_LU_GROUP_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_LU_NAME_LENGTH.

Nom d'unité logique (MQCHAR8)

Il s'agit du nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes. Définissez cette valeur sur la même unité logique que celle utilisée par le programme d'écoute pour les transmissions entrantes. Si vous laissez ce nom vide, l'unité logique par défaut APPC/MVS est utilisée ; il s'agit d'une variable. Par conséquent, définissez toujours le nom d'unité logique si vous utilisez LU6.2.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_LU_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_LU_NAME_LENGTH.

LU62ARMSuffix (MQCHAR2)

Il s'agit du suffixe de SYS1.PARMLIB APPCPMxx, qui désigne le LUADD pour cet initiateur de canal. La commande z/OS SET APPC=xx est émise lorsque ARM redémarre l'initiateur de canal. Si vous laissez ce nom à blanc, aucun SET APPC=xx n'est émis.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_LU62_ARM_SUFFIX avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_ARM_SUFFIX_LENGTH.

LU62Channels (MQLONG)

Il s'agit du nombre maximal de canaux pouvant être en cours, ou de clients pouvant être connectés, qui utilisent le protocole de transmission LU 6.2 .

La valeur doit être comprise entre 0 et 9999, avec une valeur par défaut de 200. Si vous définissez cette valeur sur zéro, le protocole de transmission LU 6.2 n'est pas utilisé.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_LU62_CHANNELS avec l'appel MQINQ.

MaxActiveactifs (MQLONG)

Cet attribut correspond au nombre maximal de canaux pouvant être *actifs* à tout moment.

La valeur par défaut est celle spécifiée pour l'attribut `MaxChannels`. Pour z/OS, la valeur doit être comprise entre 1 et 9 999. Pour toutes les autres plateformes, la valeur doit être comprise entre 1 et 65 535.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ACTIVE_CHANNELS avec l'appel **MQINQ** .

Concepts associés

Etats des canaux

MaxChannels (MQLONG)

Cet attribut correspond au nombre maximal de canaux pouvant être *en cours* (y compris les canaux de connexion serveur avec des clients connectés).

Pour z/OS, la valeur doit être comprise entre 1 et 9 999, avec une valeur par défaut de 200. Pour toutes les autres plateformes, la valeur doit être comprise entre 1 et 65 535, avec une valeur par défaut de 100. Un système qui est occupé à servir des connexions à partir du réseau peut avoir besoin d'un nombre supérieur à la valeur par défaut. Déterminez la valeur correcte pour votre environnement, idéalement en observant le comportement de votre système lors des tests.

Pour les plateformes autres que z/OS, la valeur de MaxChannels est définie dans le fichier qm.ini des gestionnaires de files d'attente respectifs.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MAX_CHANNELS avec l'appel **MQINQ** .

Concepts associés

Etats des canaux

MaxHandles (MQLONG)

Il s'agit du nombre maximal de descripteurs ouverts pouvant être utilisés simultanément par une tâche. Chaque appel MQOPEN réussi pour une file d'attente unique (ou pour un objet qui n'est pas une file d'attente) utilise un descripteur. Ce descripteur devient disponible pour être réutilisé lorsque l'objet est fermé. Toutefois, lorsqu'une liste de distribution est ouverte, chaque file d'attente de la liste de distribution se voit attribuer un descripteur distinct, de sorte que l'appel MQOPEN utilise autant de descripteurs que de files d'attente de la liste de distribution. Cela doit être pris en compte lors de la décision d'une valeur appropriée pour *MaxHandles*.

L'appel MQPUT1 effectue un appel MQOPEN dans le cadre de son traitement ; par conséquent, MQPUT1 utilise autant de descripteurs que le ferait MQOPEN, mais les descripteurs ne sont utilisés que pendant la durée de l'appel MQPUT1 lui-même.

Sous z/OS, *tâche* signifie une tâche CICS , une tâche MVS ou une région IMS dépendante.

La valeur est comprise entre 1 et 999 999 999. La valeur par défaut est déterminée par l'environnement:

- Sous z/OS, la valeur par défaut est 100.
- Dans tous les autres environnements, la valeur par défaut est 256.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MAX_POIGNÉES avec l'appel MQINQ.

Longueur MaxMsg(MQLONG)

Il s'agit de la longueur du message *physique* le plus long que le gestionnaire de files d'attente peut traiter. Toutefois, comme l'attribut de gestionnaire de files d'attente *MaxMsgLength* peut être défini indépendamment de l'attribut de file d'attente *MaxMsgLength* , le message physique le plus long pouvant être placé dans une file d'attente correspond à la valeur la moins élevée de ces deux valeurs.

Si le gestionnaire de files d'attente prend en charge la segmentation, une application peut insérer un message *logique* plus long que le moindre des deux attributs *MaxMsgLength* , mais uniquement si l'application spécifie l'indicateur MQMF_SEGMENTATION_ALLOWED dans MQMD. Si cet indicateur est spécifié, la limite supérieure de la longueur d'un message logique est de 999 999 999 999 octets, mais les contraintes de ressources imposées par le système d'exploitation ou par l'environnement dans lequel l'application est exécutée entraînent une limite inférieure.

La limite inférieure de l'attribut *MaxMsgLength* est de 32 ko (32 768 octets). La limite supérieure est de 100 Mo (104 857 600 octets).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MAX_MSG_LENGTH avec l'appel MQINQ.

MaxPriority (MQLONG)

Il s'agit de la priorité de message maximale prise en charge par le gestionnaire de files d'attente. Les priorités sont comprises entre zéro (niveau le plus bas) et *MaxPriority* (niveau le plus élevé).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MAX_PRIORITY avec l'appel MQINQ.

MaxPropertiesLongueur (MQLONG)

Permet de contrôler la taille des propriétés pouvant être associées à un message. Cela inclut à la fois le nom de la propriété en octets et la taille de la valeur de la propriété également en octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MAX_PROPERTIES_LENGTH avec l'appel MQINQ.

Messages MaxUncommitted(MQLONG)

Il s'agit du nombre maximal de messages non validés pouvant exister dans une unité de travail. Le nombre de messages non validés correspond à la somme des éléments suivants depuis le début de l'unité de travail en cours:

- Messages insérés par l'application avec l'option MQPMO_SYNCPOINT
- Messages extraits par l'application avec l'option MQGMO_SYNCPOINT
- Messages de déclenchement et messages de rapport COA générés par le gestionnaire de files d'attente pour les messages insérés avec l'option MQPMO_SYNCPOINT
- Messages de rapport COD générés par le gestionnaire de files d'attente pour les messages extraits à l'aide de l'option MQGMO_SYNCPOINT

Les éléments suivants ne sont *pas* comptés comme des messages non validés:

- Messages insérés ou extraits par l'application en dehors d'une unité de travail
- Messages de déclenchement ou messages de rapport COA/COD générés par le gestionnaire de files d'attente suite à l'insertion ou à l'extraction de messages en dehors d'une unité d'oeuvre
- Messages de rapport d'expiration générés par le gestionnaire de files d'attente (même si l'appel à l'origine du message de rapport d'expiration a spécifié MQGMO_SYNCPOINT)
- Messages d'événement générés par le gestionnaire de files d'attente (même si l'appel à l'origine du message d'événement a spécifié MQPMO_SYNCPOINT ou MQGMO_SYNCPOINT)

Remarque :

1. Les messages de rapport d'exception sont générés par l'agent MCA (Message Channel Agent) ou par l'application et sont traités de la même manière que les messages ordinaires insérés ou extraits par l'application.
2. Lorsqu'un message ou un segment est inséré avec l'option MQPMO_SYNCPOINT, le nombre de messages non validés est incrémenté de un, quel que soit le nombre de messages physiques résultant de l'insertion. (Plusieurs messages physiques peuvent être émis si le gestionnaire de files d'attente doit subdiviser le message ou le segment.)
3. Lorsqu'une liste de distribution est placée avec l'option MQPMO_SYNCPOINT, le nombre de messages non validés est incrémenté d'un *pour chaque message physique généré*. Elle peut être aussi petite que 1 ou aussi grande que le nombre de destinations dans la liste de distribution.

La limite inférieure de cet attribut est 1 ; la limite supérieure est 999 999 999. La valeur par défaut est 10 000.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MAX_UNCOMMITTED_MSGS avec l'appel MQINQ.

MQIAccounting (MQLONG)

Permet de contrôler la collecte des informations de comptabilité pour les données MQI.

La valeur est l'une des suivantes :

MQMON_ON
Collectez les données de comptabilité d'API.

MQMON_OFF
Ne collectez pas de données de comptabilité d'API. Il s'agit de la valeur par défaut.

Si vous définissez l'attribut de gestionnaire de files d'attente ACCTCONO sur ENABLED, cette valeur peut être remplacée pour des connexions individuelles à l'aide de la zone Options de la structure MQCNO. Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente qui se produisent après la modification de l'attribut.

Cet attribut est pris en charge uniquement sur les systèmes IBM i, UNIX et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ACCOUNTING_MQI avec l'appel MQINQ.

MQIStatistics (MQLONG)

Cette option contrôle la collecte des informations de surveillance des statistiques pour le gestionnaire de files d'attente.

La valeur est l'une des suivantes :

MQMON_ON
Collecte des statistiques MQI.

MQMON_OFF
Ne collectez pas de statistiques MQI. Il s'agit de la valeur par défaut.

Cet attribut est pris en charge uniquement sur les systèmes IBM i, UNIX and Linux et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_STATISTICS_MQI avec l'appel MQINQ.

MsgMarkBrowseInterval (MQLONG)

Intervalle de temps en millisecondes après lequel le gestionnaire de files d'attente peut supprimer automatiquement la marque des messages de navigation.

Il s'agit d'un intervalle de temps (en millisecondes) après lequel le gestionnaire de files d'attente peut supprimer automatiquement la marque des messages de navigation.

Cet attribut décrit l'intervalle de temps pendant lequel les messages qui ont été marqués comme parcourus par un appel à MQGET, à l'aide de l'option d'obtention de message MQGMO_MARK_BROWSE_CO_OP, doivent rester marqués comme parcourus.

Le gestionnaire de files d'attente peut automatiquement annuler le marquage des messages consultés qui ont été marqués comme ayant été consultés pour l'ensemble de descripteurs coopérant lorsqu'ils ont été marqués pour une durée supérieure à cet intervalle approximatif.

Cela n'affecte pas l'état d'un message marqué comme Parcourir, qui a été obtenu par un appel à MQGET, à l'aide de l'option d'obtention de message MQGMO_MARK_BROWSE_HANDLE.

La valeur maximale est 999 999 999 et la valeur par défaut est 5000. La valeur spéciale -1 pour *MsgMarkBrowseInterval* représente un intervalle de temps illimité.

Avertissement : Cette valeur ne doit pas être inférieure à la valeur par défaut de 5000.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MSG_MARK_BROWSE_INTERVAL avec l'appel MQINQ.

OutboundPortMax (MQLONG)

Il s'agit du numéro de port le plus élevé dans la plage, définie par OutboundPortMin et OutboundPortMax, des numéros de port à utiliser pour lier les canaux sortants.

La valeur est un entier compris entre 0 et 65535 et doit être supérieure ou égale à la valeur minimale de OutboundPort. La valeur par défaut est 0.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_OUTBOUND_PORT_MAX avec l'appel MQINQ.

OutboundPortMin (MQLONG)

Il s'agit du numéro de port le plus bas de la plage, définie par OutboundPortMin et OutboundPortMax, des numéros de port à utiliser pour lier les canaux sortants.

La valeur est un entier compris entre 0 et 65535 et doit être inférieure ou égale à la valeur maximale de OutboundPort. La valeur par défaut est 0.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_OUTBOUND_PORT_MIN avec l'appel MQINQ.

PerformanceEvent (MQLONG)

Permet de contrôler si des événements liés aux performances sont générés. Ses valeurs sont l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_PERFORMANCE_EVENT avec l'appel MQINQ.

Plateforme (MQLONG)

Indique le système d'exploitation sur lequel le gestionnaire de files d'attente s'exécute:

MQPL_AIX
AIX (même valeur que MQPL_UNIX).

MQPL_MVS
z/OS (même valeur que MQPL_ZOS).

MQPL_NSK
HP Integrity NonStop Server.

MQPL_OS390
z/OS (même valeur que MQPL_ZOS).

MQPL_OS400
IBM i.

MQPL_UNIX
Systèmes UNIX .

MQPL_WINDOWS_NT
Systèmes Windows .

MQPL_ZOS
z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_PLATFORM avec l'appel MQINQ.

PubSubNPInputMsg (MQLONG)

Indique s'il faut supprimer ou conserver un message d'entrée non distribué.

La valeur est l'une des suivantes :

Carte MQUNDELIVERED_DISCARD
Les message en entrée non persistants peuvent être supprimés s'ils ne peuvent pas être traités.

Il s'agit de la valeur par défaut.

MQUNDELIVERED_KEEP
Les message en entrée non persistants ne sont pas supprimés s'ils ne peuvent pas être traités. Dans ce cas, l'interface de publication / abonnement mise en file d'attente continuera à relancer le processus à des intervalles appropriés et ne continuera pas à traiter les messages suivants.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_PUBSUB_NP_MSG avec l'appel MQINQ.

PubSubPubSub (MQLONG)
Contrôle le comportement des messages de réponse non distribués.

La valeur est l'une des suivantes :

Mqundelivered_normal
Les réponses non persistantes qui ne peuvent pas être placées dans la file d'attente de réponses sont placées dans la file d'attente des messages non livrés. Si elles ne peuvent pas être placées dans la file d'attente des messages non livrés, elles sont supprimées.

MQUNDELIVERED_SAFE
Les réponses non persistantes ne pouvant être placées dans la file de réponses le sont dans la file de rebut. Si la réponse ne peut pas être définie et ne peut pas être placée dans la file d'attente des messages non livrés, l'interface de publication / abonnement mise en file d'attente annule l'opération en cours, puis effectue une nouvelle tentative à des intervalles appropriés et ne poursuit pas le traitement des messages suivants.

Carte MQUNDELIVERED_DISCARD
Les réponses non persistantes ne sont pas placées dans la file d'attente de réponses et sont supprimées.

Il s'agit de la valeur par défaut pour les nouveaux gestionnaires de files d'attente.

MQUNDELIVERED_KEEP
Les réponses non persistantes ne sont pas placées dans la file d'attente des messages non livrés ni supprimées. A la place, l'interface de publication / abonnement mise en file d'attente renverra l'opération en cours, puis la relancera à des intervalles appropriés.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_PUBSUB_NP_RESP avec l'appel MQINQ.

Valeur par défaut pour les gestionnaires de files d'attente migrés.
Si le gestionnaire de files d'attente a été migré depuis WebSphere MQ V6.0, la valeur initiale de cet attribut dépend des valeurs de DiscardNonPersistentResponse et de la réponse DLQNonPersistentavant la migration, comme indiqué dans le tableau suivant.

		Réponse DLQNonPersistent		
		Oui	**Non**	**Non définie**
DiscardNonPersistentResponse	**Oui**	Mqundelivered_normal	Carte MQUNDELIVERED_DISCARD	Mqundelivered_normal
	Non	MQUNDELIVERED_SAFE	MQUNDELIVERED_KEEP	MQUNDELIVERED_SAFE
	Non définie	Si SyncPointPersistent = No, MQUNDELIVERED_SAFE sinon MQUNDELIVERED_NORMAL	Si SyncPointPersistent = No, MQUNDELIVERED_KEEP sinon MQUNDELIVERED_DISCARD	Si SyncPointPersistent = No, MQUNDELIVERED_SAFE sinon MQUNDELIVERED_NORMAL

PubSubMaxMsgRetryCount (MQLONG)
Nombre de nouvelles tentatives lors du traitement d'un message de commande ayant échoué sous le point de synchronisation.

La valeur est l'une des suivantes :

0 - 999 999 999
La valeur par défaut est 5.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_PUBSUB_MAXMSG_RETRY_COUNT avec l'appel MQINQ.

PubSubSyncPoint (MQLONG)
Indique si seuls les messages persistants ou tous les messages sont traités sous le point de synchronisation.

La valeur est l'une des suivantes :

MQSYNCPOINT_IFPER
Cela permet à l'interface de publication / abonnement en file d'attente de recevoir des messages non persistants en dehors du point de synchronisation. S'il reçoit une publication hors du point de synchronisation, il la transmet aux abonnés qu'il connaît hors du point de synchronisation.

Il s'agit de la valeur par défaut.

MQSYNCPOINT_OUI
Cela permet à l'interface de publication / abonnement en file d'attente de recevoir tous les messages sous le point de synchronisation.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_PUBSUB_SYNC_PT avec l'appel MQINQ.

Mode PubSub(MQLONG)
Indique si le moteur de publication / abonnement et l'interface de publication / abonnement en file d'attente sont en cours d'exécution, ce qui permet aux applications de publier / abonnement à l'aide de l'interface de programmation d'application et des files d'attente surveillées par l'interface de publication / abonnement en file d'attente.

La valeur est l'une des suivantes :

MQPSM_COMPAT
Le moteur publication/abonnement est en cours d'exécution. Il est donc possible de publier / s'abonner à l'aide de l'interface de programmation d'application. L'interface de publication / abonnement en file d'attente n'est pas en cours d'exécution. Par conséquent, aucun message inséré dans les files d'attente surveillées par l'interface de publication / abonnement en file d'attente n'est traité. Ce paramètre est utilisé à des fins de compatibilité avec WebSphere Message Broker V6 ou versions antérieures utilisant ce gestionnaire de files d'attente, car il doit lire les mêmes files d'attente que celles à partir desquelles l'interface de publication / abonnement en file d'attente lit normalement.

MQPSM_DISABLED
Le moteur pub./abon. et l'interface pub./abon. en file d'attente sont inactifs. Il n'est donc pas possible de publier / s'abonner à l'aide de l'interface de programmation d'application. Les messages de publication / abonnement placés dans les files d'attente surveillées par l'interface de publication / abonnement en file d'attente ne sont pas traités.

MQPSM_ENABLED
Le moteur publication/abonnement et l'interface publication/abonnement sont en cours d'exécution. Il est donc possible de publier / abonner à l'aide de l'interface de programmation d'application et des files d'attente surveillées par l'interface de publication / abonnement en file d'attente. Il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_PUBSUB_MODE avec l'appel MQINQ.

QMgrDesc (MQCHAR64)

Utilisez cette zone pour un commentaire décrivant le gestionnaire de files d'attente. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si

nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, cette zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

Remarque : Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente *CodedCharSetId*), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

- Sous z/OS, la valeur par défaut est le nom du produit et le numéro de version.
- Dans tous les autres environnements, la valeur par défaut est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_Q_MGR_DESC avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_MGR_DESC_LENGTH.

QMgrIdentifier (MQCHAR48)

Il s'agit d'un nom unique généré en interne pour le gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_Q_MGR_IDENTIFIER avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_MGR_IDENTIFIER_LENGTH.

Cet attribut est pris en charge dans les environnements suivants: AIX, HP-UX, z/OS, IBM i, Solaris, Linux, Windows, plus WebSphere MQ clients connectés à ces systèmes.

QMgrName (MQCHAR48)

Il s'agit du nom du gestionnaire de files d'attente local, c'est-à-dire du nom du gestionnaire de files d'attente auquel l'application est connectée.

Les 12 premiers caractères du nom sont utilisés pour construire un identificateur de message unique (voir la zone MQMD- MsgId). Les gestionnaires de files d'attente qui peuvent intercommuniquer doivent donc avoir des noms qui diffèrent dans les 12 premiers caractères, afin que les identificateurs de message soient uniques dans le réseau de gestionnaires de files d'attente.

Sous z/OS, le nom est identique au nom du sous-système, qui est limité à 4 caractères non blancs.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_Q_MGR_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_MGR_NAME_LENGTH.

QSGName (MQCHAR4)

Il s'agit du nom du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local. Si le gestionnaire de files d'attente local n'appartient pas à un groupe de partage de files d'attente, le nom est vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_QSG_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_QSG_NAME_LENGTH.

Cet attribut est pris en charge uniquement sur z/OS.

QueueAccounting (MQLONG)

Permet de contrôler la collecte des informations de comptabilité pour les files d'attente.

La valeur est l'une des suivantes :

MQMON_AUCUN
Ne collectez pas de données de comptabilité pour les files d'attente, quelle que soit la valeur de l'attribut de comptabilité de file d'attente ACCTQ. Il s'agit de la valeur par défaut.

MQMON_OFF
Ne collectez pas de données comptables pour les files d'attente qui spécifient QMGR dans l'attribut de file d'attente ACCTQ.

MQMON_ON
Collectez les données de comptabilité pour les files d'attente qui spécifient QMGR dans l'attribut de file d'attente ACCTQ.

Les modifications apportées à cette valeur ne sont effectives que pour les connexions au gestionnaire de files d'attente qui se produisent après la modification de l'attribut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ACCOUNTING_Q avec l'appel MQINQ.

QueueMonitoring (MQLONG)

Indique le paramètre par défaut pour la surveillance en ligne des files d'attente.

Si l'attribut de file d'attente *QueueMonitoring* est défini sur MQMON_Q_MGR, cet attribut spécifie la valeur utilisée par le canal. La valeur peut être :

MQMON_OFF
La collecte des données de surveillance en ligne est désactivée. Il s'agit de la valeur par défaut initiale du gestionnaire de files d'attente.

MQMON_AUCUN
La collecte des données de surveillance en ligne est désactivée pour les files d'attente, quelle que soit la valeur de leur attribut *QueueMonitoring* .

MQMON_FAIBLE
La collecte de données de surveillance en ligne est activée, avec un faible taux de collecte de données.

MQMON_MEDIUM
La collecte de données de surveillance en ligne est activée, avec un taux modéré de collecte de données.

MQMON_ELEVE
La collecte de données de surveillance en ligne est activée, avec un taux élevé de collecte de données.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MONITORING_Q avec l'appel MQINQ.

QueueStatistics (MQLONG)

Cette option contrôle la collecte des données statistiques pour les files d'attente.

Ses valeurs sont l'une des suivantes :

MQMON_AUCUN
Ne collectez pas de statistiques de file d'attente pour les files d'attente, quel que soit le paramètre de l'attribut de file d'attente *QueueStatistics* . Il s'agit de la valeur par défaut.

MQMON_OFF
Ne collectez pas de données statistiques pour les files d'attente qui spécifient le gestionnaire de files d'attente dans l'attribut de file d'attente *QueueStatistics* .

MQMON_ON
Collectez des données statistiques pour les files d'attente qui spécifient le gestionnaire de files d'attente dans l'attribut de file d'attente *QueueStatistics* .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_STATISTICS_Q avec l'appel MQINQ.

ReceiveTimeout (MQLONG)

Indique la durée pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de son partenaire avant de revenir à l'état inactif. Elle s'applique uniquement aux canaux de message et non aux canaux MQI.

La signification exacte de ReceiveTimeout est modifiée par la valeur spécifiée dans le type ReceiveTimeout. Le type ReceiveTimeoutpeut être défini sur l'une des valeurs suivantes:

- MQRCVTIME_EQUAL-cette valeur correspond au nombre de secondes d'attente du canal. Indiquez une valeur comprise entre 0 et 999999.
- MQRCVTIME_ADD-cette valeur correspond au nombre de secondes à ajouter à l'élément HBINT négocié et détermine la durée d'attente d'un canal. Indiquez une valeur comprise entre 1 et 999999.
- MQRCVTIME_MULTIPLIER-cette valeur est un multiplicateur à appliquer au HBINT négocié. Indiquez une valeur de 0 ou une valeur comprise entre 2 et 99.

La valeur par défaut est 0.

Définissez ReceiveTimeoutsur MQRCVTIME_MULTIPLIER ou MQRCVTIME_EQUAL, et sur ReceiveTimeout sur 0, pour empêcher un canal d'expirer son attente de réception des données de son partenaire.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_RECEIVE_TIMEOUT avec l'appel MQINQ.

ReceiveTimeoutMin (MQLONG)

Il s'agit de la durée minimale, en secondes, pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif.

Elle s'applique uniquement aux canaux de transmission de messages et non aux canaux MQI. La valeur doit être comprise entre 0 et 999999, la valeur par défaut étant 0.

Si vous utilisez le type ReceiveTimeoutpour indiquer que le temps d'attente du canal TCP/IP doit être calculé par rapport à la valeur négociée de HBINT et que la valeur résultante est inférieure à la valeur de ce paramètre, cette valeur est utilisée à la place.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_RECEIVE_TIMEOUT_MIN avec l'appel MQINQ.

ReceiveTimeoutType (MQLONG)

Il s'agit du qualificateur appliqué à ReceiveTimeout pour définir la durée pendant laquelle un canal TCP/IP attend la réception de données, y compris les signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. Elle s'applique uniquement aux canaux de transmission de messages et non aux canaux MQI.

La valeur est l'une des suivantes :

MQRCVTIME_MULTIPLIER
ReceiveTimeout est un multiplicateur à appliquer à la valeur HBINT négociée pour déterminer la durée d'attente d'un canal. Il s'agit de la valeur par défaut.

MQRCVTIME_ADD
ReceiveTimeout est une valeur, en secondes, à ajouter à la valeur HBINT négociée pour déterminer la durée d'attente d'un canal.

MQRCVTIME_EQUAL
ReceiveTimeout est une valeur, en secondes, que le canal attend.

Pour arrêter un dépassement du délai d'attente d'un canal pour recevoir des données de son partenaire, définissez le type ReceiveTimeoutsur MQRCVTIME_MULTIPLIER ou MQRCVTIME_EQUAL et ReceiveTimeout sur 0.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_RECEIVE_TIMEOUT_TYPE avec l'appel MQINQ.

RemoteEvent (MQLONG)

Permet de contrôler si des événements d'erreur distants sont générés. Ses valeurs sont l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_REMOTE_EVENT avec l'appel MQINQ.

RepositoryName (MQCHAR48)

Il s'agit du nom d'un cluster pour lequel ce gestionnaire de files d'attente fournit un service de gestionnaire de référentiels. Si le gestionnaire de files d'attente fournit ce service pour plusieurs clusters, *RepositoryNamelist* spécifie le nom d'un objet liste de noms qui identifie les clusters et *RepositoryName* est vide. Au moins l'un des éléments *RepositoryName* et *RepositoryNamelist* doit être vide.

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris, Windowset z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_REPOSITORY_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_MGR_NAME_LENGTH.

RepositoryNamelist (MQCHAR48)

Il s'agit du nom d'un objet liste de noms qui contient les noms des clusters pour lesquels ce gestionnaire de files d'attente fournit un service de gestionnaire de référentiels. Si le gestionnaire de files d'attente fournit ce service pour un seul cluster, l'objet liste de noms ne contient qu'un seul nom. Vous pouvez également utiliser *RepositoryName* pour spécifier le nom du cluster, auquel cas *RepositoryNamelist* est vide. Au moins l'un des éléments *RepositoryName* et *RepositoryNamelist* doit être vide.

Cet attribut est pris en charge uniquement sous AIX, HP-UX, IBM i, Linux, Solaris, Windowset z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_REPOSITORY_NAMELIST avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_NAMELIST_NAME_LENGTH.

ScyCase(MQCHAR8)

Indique si le gestionnaire de files d'attente prend en charge les noms de profil de sécurité en casse mixte ou en majuscules uniquement.

La valeur est l'une des suivantes :

MQSCYC_SUPÉRIEURE
Les noms de profil de sécurité doivent être en majuscules.

MQSCYC_MIXTE
Les noms de profil de sécurité peuvent être en majuscules ou en casse mixte.

Les modifications apportées à cet attribut prennent effet lorsqu'une commande Refresh Security est exécutée avec *SecurityType(MQSECTYPE_CLASSES)* spécifié.

Cet attribut est pris en charge uniquement sur z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_SECURITY_CASE avec l'appel MQINQ.

Nom SharedQMgr(MQLONG)

Indique si *ObjectQmgrName* doit être utilisé ou traité comme gestionnaire de files d'attente local sur un appel MQOPEN, pour une file d'attente partagée, lorsque *ObjectQmgrName* est celui d'un autre gestionnaire de files d'attente du groupe de partage de files d'attente.

La valeur peut être :

MQSQQM_USE
ObjectQmgrName est utilisé et la file d'attente de transmission appropriée est ouverte.

MQSQQM_IGNORE
Si la file d'attente cible est partagée et que *ObjectQmgrName* est celle d'un gestionnaire de files d'attente du même groupe de partage de files d'attente, l'ouverture est effectuée en local.

Cet attribut est valide uniquement sur z/OS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_SHARED_Q_Q_MGR_NAME avec l'appel MQINQ.

SPLCAP
Indique si les fonctions de sécurité de WebSphere MQ Advanced Message Security sont disponibles pour un gestionnaire de files d'attente.

MQCAP_PRIS en charge
Il s'agit de la valeur par défaut si le composant AMS WebSphere MQ est installé pour l'installation sous laquelle s'exécute le gestionnaire de files d'attente.

MQCAP_NON_PRIS en charge

SSLEvent (MQLONG)
Indique si des événements SSL sont générés.

Ses valeurs sont l'une des suivantes :

MQEVR_ENABLED
Générez des événements SSL, comme suit:

MQRC_CHANNEL_SSL_ERROR

MQEVR_DISABLED
Ne générez pas d'événements SSL ; il s'agit de la valeur par défaut.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_SSL_EVENT avec l'appel MQINQ.

SSLFIPSRequired (MQLONG)

Cela vous permet de spécifier que seuls les algorithmes certifiés FIPS doivent être utilisés si la cryptographie est exécutée dans WebSphere MQ, plutôt que dans du matériel de cryptographie. Si du matériel de cryptographie est configuré, les modules de cryptographie utilisés sont ceux fournis par le produit matériel ; ces modules peuvent ou non être certifiés FIPS à un niveau particulier en fonction du produit matériel utilisé.

La valeur est l'une des suivantes:

MQSSL_FIPS_NO
Utilisez tout CipherSpec pris en charge sur la plateforme utilisée. Cette valeur est la valeur par défaut.

MQSSL_FIPS_YES
Utilisez uniquement les algorithmes de cryptographie certifiés FIPS dans les CipherSpecs autorisés sur toutes les connexions SSL depuis et vers ce gestionnaire de files d'attente.

Ce paramètre est valide uniquement sur les plateformes UNIX, Linux, Windowset z/OS .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_SSL_FIPS_REQUIRED avec l'appel MQINQ.

Tâches associées
Comment indiquer que seuls les CipherSpecs certifiés FIPS sont utilisés lors de l'exécution sur MQI Client
Référence associée
FIPS (Federal Information Processing Standards) pour UNIX, Linux et Windows

Nombre SSLKeyReset(MQLONG)
Indique à quel moment les agents MCA qui initient la communication réinitialisent la clé secrète utilisée pour le chiffrement sur le canal.

La valeur représente le nombre total d'octets non chiffrés qui sont envoyés et reçus sur le canal avant la renégociation de la clé secrète. Le nombre d'octets inclut les informations de contrôle envoyées par l'agent MCA.

La valeur est un nombre compris entre 0 et 999 999 999, avec une valeur par défaut de 0. Si vous spécifiez un nombre de réinitialisations de clés secrètes SSL/TLS compris entre 1 et 32 Ko, les canaux SSL/TLS utiliseront un nombre de réinitialisations de clés secrètes de 32 Ko. Cela permet d'éviter le coût de traitement des réinitialisations de clé excessives qui se produiraient pour les petites valeurs de réinitialisation de clé secrète SSL/TLS.

La clé secrète est renégociée lorsque le nombre total d'octets non chiffrés envoyés et reçus par l'agent MCA du canal initiateur dépasse la valeur spécifiée, ou si les signaux de présence du canal sont activés avant l'envoi ou la réception des données à la suite d'un signal de présence du canal, selon ce qui se produit en premier.

Le nombre d'octets envoyés et reçus pour la renégociation inclut les informations de contrôle envoyées et reçues par l'agent MCA de canal et est réinitialisé chaque fois qu'une renégociation se produit.

Utilisez la valeur 0 pour indiquer que les clés secrètes ne sont jamais renégociées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_SSL_RESET_COUNT avec l'appel MQINQ.

Evénement StartStop(MQLONG)

Cette option contrôle si les événements de démarrage et d'arrêt sont générés. La valeur est l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_START_STOP_EVENT avec l'appel MQINQ.

StatisticsInterval (MQLONG)

Indique la fréquence (en secondes) à laquelle les données de surveillance des statistiques doivent être écrites dans la file d'attente de surveillance.

La valeur est un entier compris entre 0 et 604800, avec une valeur par défaut de 1800 (30 minutes).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_STATISTICS_INTERVAL avec l'appel MQINQ.

SyncPoint (MQLONG)

Indique si le gestionnaire de files d'attente local prend en charge les unités de travail et la synchronisation avec les appels MQGET, MQPUT et MQPUT1 .

MQSP_XX_ENCODE_CASE_ONE disponible
Unités de travail et point de synchronisation disponibles.

MQSP_NON DISPONIBLE
Unités de travail et point de synchronisation non disponibles.

- Sous z/OS , cette valeur n'est jamais renvoyée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_SYNCPOINT avec l'appel MQINQ.

TCPChannels (MQLONG)

Il s'agit du nombre maximal de canaux pouvant être en cours, ou de clients pouvant être connectés, qui utilisent le protocole de transmission TCP/IP.

La valeur doit être comprise entre 0 et 9999, avec une valeur par défaut de 200. Si vous indiquez 0, TCP/IP n'est pas utilisé.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TCP_CHANNELS avec l'appel MQINQ.

TCPKeepAlive (MQLONG)

Indique si TCP KEEPALIVE doit être utilisé pour vérifier que l'autre extrémité de la connexion est toujours disponible. S'il n'est pas disponible, le canal est fermé.

La valeur est l'une des suivantes :

MQTCPKEEP_OUI
Utilisez TCP KEEPALIVE comme indiqué dans le fichier de configuration de profil TCP. Si vous spécifiez l'attribut de canal KeepAliveInterval (KAINT), la valeur à laquelle il est défini est utilisée.

MQTCPKEEP_NO
N'utilisez pas TCP KEEPALIVE. Il s'agit de la valeur par défaut.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TCP_KEEP_ALIVE avec l'appel MQINQ.

TCPName (MQCHAR8)

Il s'agit du nom du seul système TCP/IP ou du système TCP/IP par défaut que vous utilisez, en fonction de la valeur de TCPStackType. La valeur par défaut est TCPIP.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_TCP_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_TCP_NAME_LENGTH.

TCPStackType (MQLONG)

Indique si l'initiateur de canal peut utiliser uniquement l'espace adresse TCP/IP spécifié dans TCPName, ou s'il peut éventuellement se connecter à une adresse TCP/IP sélectionnée

La valeur est l'une des suivantes :

MQTCPSTACK_SINGLE
L'initiateur de canal peut utiliser uniquement les espaces adresse TCP/IP nommés dans TCPName. Il s'agit de la valeur par défaut.

MQTCPSTACK_MULTIPLE
L'initiateur de canal peut utiliser n'importe quel espace adresse TCP/IP disponible. La valeur par défaut est celle indiquée dans TCPName si aucune autre valeur n'est indiquée pour un canal ou un programme d'écoute.

Cet attribut est pris en charge sur z/OS uniquement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TCP_STACK_TYPE avec l'appel MQINQ.

Enregistrement de TraceRoute(MQLONG)

Permet de contrôler l'enregistrement des informations de routage de trace.

La valeur est l'une des suivantes :

MQRECORDING_DISABLED
Aucun ajout aux messages de trace-route n'est autorisé.

MQRECORDING_Q
Insertion de messages de trace-route dans une file d'attente nommée fixe.

MSG_ENREGISTREMENT_MQG
Placez les messages de trace-route dans une file d'attente déterminée à l'aide du message lui-même. Il s'agit de la valeur par défaut

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TRACE_ROUTE_RECORDING avec l'appel MQINQ.

TriggerInterval (MQLONG)

Il s'agit d'un intervalle de temps (en millisecondes) utilisé pour limiter le nombre de messages de déclenchement. Cela n'est pertinent que si *TriggerType* est MQTT_FIRST. Dans ce cas, les messages de déclenchement sont généralement générés uniquement lorsqu'un message approprié arrive dans la file d'attente et que la file d'attente était précédemment vide. Dans certaines circonstances, toutefois, un message de déclenchement supplémentaire peut être généré avec le déclenchement MQTT_FIRST même si la file d'attente n'était pas vide. Ces messages de déclenchement supplémentaires ne sont pas générés plus souvent que toutes les *TriggerInterval* millisecondes.

Pour plus d'informations sur le déclenchement, voir Déclenchement de canaux .

La valeur n'est pas inférieure à 0 ni supérieure à 999 999 999. La valeur par défaut est 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TRIGGER_INTERVAL avec l'appel MQINQ.

TriggerInterval (MQLONG)

Il s'agit d'un intervalle de temps (en millisecondes) utilisé pour limiter le nombre de messages de déclenchement. Cela n'est pertinent que si *TriggerType* est MQTT_FIRST. Dans ce cas, les messages de déclenchement sont généralement générés uniquement lorsqu'un message approprié arrive dans la file d'attente et que la file d'attente était précédemment vide. Dans certaines circonstances, toutefois, un message de déclenchement supplémentaire peut être généré avec le déclenchement MQTT_FIRST même si la file d'attente n'était pas vide. Ces messages de déclenchement supplémentaires ne sont pas générés plus souvent que toutes les *TriggerInterval* millisecondes.

Pour plus d'informations sur le déclenchement, voir Déclenchement de canaux .

La valeur n'est pas inférieure à 0 ni supérieure à 999 999 999. La valeur par défaut est 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TRIGGER_INTERVAL avec l'appel MQINQ.

Version (MQCFST)

Il s'agit de la version du code WebSphere MQ en tant que VVRRMMFF, où:

VV-Version

RR-Libération

MM-Niveau de maintenance

FF-Niveau de correctif

XrCapability(MQLONG)

Cette commande contrôle si les commandes WebSphere MQ Telemetry sont prises en charge par le gestionnaire de files d'attente.

La valeur est l'une des suivantes :

MQCAP_PRIS en charge
Le composant WebSphere MQ Telemetry est installé et les commandes de télémétrie sont prises en charge.

MQCAP_NON_PRIS en charge
Le composant webSphere MQ Telemetry n'est pas installé.

Cet attribut est pris en charge uniquement sur les systèmes IBM i, Unix et Windows.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_XR_CAPABILITY avec l'appel MQINQ .

Attributs des files d'attente

Il existe cinq types de définition de file d'attente. Certains attributs de file d'attente s'appliquent à tous les types de file d'attente ; d'autres s'appliquent uniquement à certains types de file d'attente.

Types de file d'attente

Le gestionnaire de files d'attente prend en charge les types de définition de file d'attente suivants:

File d'attente locale
Vous pouvez stocker des messages dans une file d'attente locale. Sous z/OS , vous pouvez en faire une file d'attente partagée ou privée.

Une file d'attente est considérée par un programme comme étant *locale* si elle est détenue par le gestionnaire de files d'attente auquel le programme est connecté. Vous pouvez extraire et insérer des messages dans les files d'attente locales.

L'objet définition de file d'attente stocke les informations de définition de la file d'attente, ainsi que les messages physiques insérés dans la file d'attente.

file d'attente du gestionnaire de files d'attente local
La file d'attente existe sur le gestionnaire de files d'attente local. La file d'attente est appelée file d'attente privée sur z/OS.

File d'attente partagée (z/OS uniquement)
La file d'attente existe dans un référentiel partagé accessible à tous les gestionnaires de files d'attente appartenant au groupe de partage de files d'attente propriétaire du référentiel partagé.

Les applications connectées à un gestionnaire de files d'attente dans le groupe de partage de files d'attente peuvent placer des messages dans des files d'attente de ce type et en supprimer. Ces files d'attente sont en fait les mêmes que les files d'attente locales. La valeur de l'attribut de file d'attente *QType* est MQQT_LOCAL.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente de ce type et en supprimer. La valeur de l'attribut de file d'attente *QType* est MQQT_LOCAL.

File d'attente de cluster

Vous pouvez stocker des messages dans une file d'attente de cluster sur le gestionnaire de files d'attente où ils sont définis. Une file d'attente de cluster est une file d'attente hébergée par un gestionnaire de files d'attente de cluster et accessible aux autres gestionnaires de files d'attente dans le cluster. La valeur de l'attribut de file d'attente *QType* est MQQT_CLUSTER.

Une file d'attente de cluster est annoncée aux autres gestionnaires de files d'attente de cluster. Ces autres gestionnaires de files d'attente de cluster peuvent insérer des messages dans une file d'attente de cluster sans qu'une définition de file d'attente éloignée correspondante soit nécessaire. Une file d'attente de cluster peut être annoncée dans plusieurs clusters à l'aide d'une liste de noms de cluster.

Lorsqu'une file d'attente est annoncée, les gestionnaires de files d'attente de cluster peuvent y insérer des messages. Pour insérer un message, le gestionnaire de files d'attente doit déterminer, à partir des référentiels complets, l'emplacement où la file d'attente est hébergée. Il ajoute alors des informations de routage au message et insère ce dernier dans une file d'attente de transmission du cluster.

A l'exception de z/OS, un gestionnaire de files d'attente peut stocker des messages pour d'autres gestionnaires de files d'attente d'un cluster dans plusieurs files d'attente de transmission. Vous pouvez configurer un gestionnaire de files d'attente de deux manières pour stocker les messages dans plusieurs files d'attente de transmission de cluster. Si vous définissez la valeur CHANNEL pour l'attribut DEFCLXQ, une file d'attente de transmission du cluster est créée automatiquement depuis SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE pour chaque canal émetteur de cluster. Si

vous définissez l'option de file d'attente de transmission CLCHNAME sur un ou plusieurs canaux émetteurs de cluster, le gestionnaire de files d'attente peut stocker les messages pour les canaux correspondants dans cette file d'attente de transmission.

Une file d'attente de cluster peut être une file d'attente partagée par les membres d'un groupe de partage de files d'attente dans IBM WebSphere MQ for z/OS.

File d'attente éloignée
Une file d'attente éloignée n'est pas une file d'attente physique ; il s'agit de la définition locale d'une file d'attente qui existe sur un gestionnaire de files d'attente éloignées. La définition locale de la file d'attente éloignée contient des informations indiquant au gestionnaire de files d'attente local comment acheminer les messages vers le gestionnaire de files d'attente éloignées.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente de ce type ; les messages sont placés dans la file d'attente de transmission locale utilisée pour acheminer les messages vers le gestionnaire de files d'attente éloignées. Les applications ne peuvent pas supprimer les messages des files d'attente éloignées. La valeur de l'attribut de file d'attente *QType* est MQQT_REMOTE.

Vous pouvez également utiliser une définition de file d'attente éloignée pour:

- Alias de file d'attente de réponses

  Dans ce cas, le nom de la définition est le nom d'une file d'attente de réponse. Pour plus d'informations, voir Alias de file d'attente de réponse et clusters.

- Alias du gestionnaire de files d'attente

  Dans ce cas, le nom de la définition est un alias pour un gestionnaire de files d'attente et non le nom d'une file d'attente. Pour plus d'informations, voir Alias des gestionnaires de files d'attente et clusters.

File d"attente d'alias
Il ne s'agit pas d'une file d'attente physique ; il s'agit d'un autre nom pour une file d'attente locale, une file d'attente partagée, une file d'attente de cluster ou une file d'attente éloignée. Le nom de la file d'attente dans laquelle l'alias est résolu fait partie de la définition de la file d'attente alias.

Les applications connectées au gestionnaire de files d'attente local peuvent placer des messages dans des files d'attente de ce type ; les messages sont placés dans la file d'attente dans laquelle l'alias est résolu. Les applications peuvent supprimer des messages des files d'attente de ce type si l'alias est résolu en file d'attente locale, en file d'attente partagée ou en file d'attente de cluster comportant une instance locale. La valeur de l'attribut de file d'attente *QType* est MQQT_ALIAS.

File d'attente modèle
Il ne s'agit pas d'une file d'attente physique ; il s'agit d'un ensemble d'attributs de file d'attente à partir desquels une file d'attente locale peut être créée.

Les messages ne peuvent pas être stockés dans des files d'attente de ce type.

Attributs File d'attente

Certains attributs de file d'attente s'appliquent à tous les types de file d'attente ; d'autres s'appliquent uniquement à certains types de file d'attente. Les types de file d'attente auxquels un attribut s'applique sont affichés dans Tableau 573, à la page 826 et dans les tableaux suivants.

Le Tableau 573, à la page 826 récapitule les attributs spécifiques aux files d'attente. Les attributs sont décrits par ordre alphabétique.

Remarque : Les noms des attributs affichés dans cette section sont des noms descriptifs utilisés avec les appels MQINQ et MQSET ; les noms sont identiques à ceux des commandes PCF. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, des noms abrégés alternatifs sont utilisés ; pour plus de détails, voir Commandes MQSC (Script) .

Tableau 573. Attributs des files d'attente. Les colonnes s'appliquent comme suit:

- La colonne des files d'attente locales s'applique également aux files d'attente partagées.
- La colonne des files d'attente modèles indique les attributs hérités par la file d'attente locale créée à partir de la file d'attente modèle.
- La colonne des files d'attente de cluster indique les attributs qui peuvent être renseignés lorsque la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie. Si d'autres attributs sont requis, l'appel renvoie le code achèvement MQCC_WARNING et le code anomalie MQRC_SELECTOR_NOT_FOR_TYPE (2068).

  Si la file d'attente de cluster est ouverte pour l'interrogation plus une ou plusieurs entrées, parcourez ou définissez, la colonne des files d'attente locales s'applique à la place.

  Si la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie, en plus de la spécification du nom du gestionnaire de files d'attente de base, la colonne des files d'attente locales s'applique à la place.

Attribut	Description	Locale	Modèle	Alias	Eloignée	Cluster
AlterationDate	Date de la dernière modification de la définition	✓		✓	✓	
AlterationTime	Heure de la dernière modification de la définition	✓		✓	✓	
BackoutRequeueQName	Nom de file d'attente de remise en file d'attente d'annulation excessive	✓	✓			
BackoutThreshold	Seuil d'annulation	✓	✓			
BaseQName	Nom de la file d'attente dans laquelle l'alias est résolu			✓		
CFStrucName	Nom de la structure d'unité de couplage	✓	✓			
CLCHNAME	Noms des canaux émetteurs de cluster	✓	✓			
ClusterName	Nom du cluster auquel appartient la file d'attente	✓		✓	✓	✓
ClusterNamelist	Nom de l'objet liste de noms contenant les noms des clusters auxquels appartient la file d'attente	✓		✓	✓	
CLWLQueuePriority	Priorité de file d'attente de charge de travail	✓		✓	✓	✓
CLWLQueueRank	Rang de file d'attente de charge de travail	✓		✓	✓	✓
CLWLUseQ	Utiliser la file d'attente éloignée	✓				
CreationDate	Date de création de la file d'attente	✓				
CreationTime	Heure de création de la file d'attente	✓				
CurrentQDepth	Longueur actuelle de la file d'attente	✓				
DefaultPutResponse	Réponse d'insertion par défaut	✓	✓	✓	✓	
DefBind	Liaison par défaut	✓		✓	✓	✓
DefinitionType attribute	Type de définition de file d'attente	✓	✓			
DefInputOpenOption	Option d'ouverture en entrée par défaut	✓	✓			
DefPersistence	Persistance par défaut	✓	✓	✓	✓	✓
DefPriority	Priorité par défaut	✓	✓	✓	✓	✓
DefReadAhead	Lecture anticipée par défaut	✓	✓	✓		
DistLists	Prise en charge de la liste de diffusion	✓	✓			

Tableau 573. Attributs des files d'attente. Les colonnes s'appliquent comme suit:

- La colonne des files d'attente locales s'applique également aux files d'attente partagées.
- La colonne des files d'attente modèles indique les attributs hérités par la file d'attente locale créée à partir de la file d'attente modèle.
- La colonne des files d'attente de cluster indique les attributs qui peuvent être renseignés lorsque la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie. Si d'autres attributs sont requis, l'appel renvoie le code achèvement MQCC_WARNING et le code anomalie MQRC_SELECTOR_NOT_FOR_TYPE (2068).

  Si la file d'attente de cluster est ouverte pour l'interrogation plus une ou plusieurs entrées, parcourez ou définissez, la colonne des files d'attente locales s'applique à la place.

  Si la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie, en plus de la spécification du nom du gestionnaire de files d'attente de base, la colonne des files d'attente locales s'applique à la place.

(suite)

Attribut	Description	Locale	Modèle	Alias	Eloignée	Cluster
HardenGetBackout	Indique s'il convient de conserver un nombre d'annulations précis	✓	✓			
IndexType	Type d'index	✓	✓			
InhibitGet	Indique si les opérations d'extraction de la file d'attente sont autorisées	✓	✓	✓		
InhibitPut	Indique si les opérations d'insertion pour la file d'attente sont autorisées	✓	✓	✓	✓	✓
InitiationQName	Nom de la file d'attente d'initialisation	✓	✓			
MaxMsgLength	Longueur maximale des messages (en octets)	✓	✓			
MaxQDepth	Long. max. file d'attente	✓	✓			
MsgDeliverySequence attribute	Séquence livraison messages	✓	✓			
NonPersistentMessage Class	Objectif de fiabilité pour les messages non persistants	✓	✓			
OpenInputCount	Nombre d'ouvertures pour la saisie	✓				
OpenOutputCount	Nombre d'ouvertures pour la sortie	✓				
PropertyControl	Contrôle des propriétés	✓	✓	✓		
ProcessName	Nom du processus	✓	✓			
QDepthHighEvent attribute	Indique si les événements Longueur élevée de la file d'attente sont générés	✓	✓			
QDepthHighLimit	Limite supérieure de la longueur de la file d'attente	✓	✓			
QDepthLowEvent attribute	Indique si les événements Longueur faible de la file d'attente sont générés	✓	✓			
QDepthLowLimit attribute	Limite inférieure de la longueur de la file d'attente	✓	✓			
QDepthMaxEvent	Indique si des événements de file d'attente saturée sont générés	✓	✓			
QDesc	Description de file d'attente	✓	✓	✓	✓	✓
QName	Nom de la file d'attente	✓		✓	✓	✓
QServiceInterval	Cible de l'intervalle de service de file d'attente	✓	✓			

Tableau 573. Attributs des files d'attente. Les colonnes s'appliquent comme suit:

- La colonne des files d'attente locales s'applique également aux files d'attente partagées.
- La colonne des files d'attente modèles indique les attributs hérités par la file d'attente locale créée à partir de la file d'attente modèle.
- La colonne des files d'attente de cluster indique les attributs qui peuvent être renseignés lorsque la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie. Si d'autres attributs sont requis, l'appel renvoie le code achèvement MQCC_WARNING et le code anomalie MQRC_SELECTOR_NOT_FOR_TYPE (2068).

  Si la file d'attente de cluster est ouverte pour l'interrogation plus une ou plusieurs entrées, parcourez ou définissez, la colonne des files d'attente locales s'applique à la place.

  Si la file d'attente de cluster est ouverte pour l'interrogation seule ou pour l'interrogation et la sortie, en plus de la spécification du nom du gestionnaire de files d'attente de base, la colonne des files d'attente locales s'applique à la place.

(suite)

Attribut	Description	Locale	Modèle	Alias	Eloignée	Cluster
QServiceIntervalEvent attribute	Indique si des événements d'intervalle de service élevé ou d'intervalle de service OK sont générés	✓	✓			
QSGDisp attribute	Disposition de groupe de partage de files d'attente	✓		✓	✓	
QueueAccounting	Collecte des données de comptabilité des files d'attente	✓	✓	✓	✓	✓
QueueMonitoring	Données de surveillance en ligne pour les files d'attente	✓	✓			
QueueStatistics	collecte de données statistiques de file d'attente	✓	✓	✓	✓	✓
QType	Type de file d'attente	✓		✓	✓	✓
RemoteQMgrName	Nom du gestionnaire de files d'attente éloignées				✓	
RemoteQName	Nom de la file d'attente éloignée				✓	
RetentionInterval	Intervalle de conservation	✓	✓			
Scope	Indique si une entrée de la file d'attente existe également dans un répertoire de cellule	✓		✓	✓	
Shareability	Partageabilité de la file d'attente	✓	✓			
StorageClass	Classe de stockage pour la file d'attente	✓	✓			
TriggerControl	Contrôle du déclenchement	✓	✓			
TriggerData	Données de déclenchement	✓	✓			
TriggerDepth	Longueur de déclenchement	✓	✓			
TriggerMsgPriority	Priorité des messages de seuil des déclencheurs	✓	✓			
TriggerType	Type de déclenchement	✓	✓			
Usage attribute	Utilisation de la file d'attente	✓	✓			
XmitQName	Nom de la file d'attente de transmission				✓	

Concepts associés

Files d'attente de cluster

Files d'attente locales

AlterationDate (MQCHAR12)

Date de la dernière modification de la définition.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour obtenir une longueur de 12 octets (par exemple, 1992-09-23¬¬, où ¬¬ représente deux caractères blancs).

Les valeurs de certains attributs (par exemple, *CurrentQDepth*) changent lorsque le gestionnaire de files d'attente fonctionne. Les modifications apportées à ces attributs n'affectent pas *AlterationDate*.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ALTERATION_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_DATE_LENGTH.

AlterationTime (MQCHAR8)

Heure de la dernière modification de la définition.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS à l'aide de l'horloge au format 24 heures, avec un zéro non significatif si l'heure est inférieure à 10 (par exemple, 09.10.20).

- Sous z/OS, l'heure est l'heure GMT (Greenwich Mean Time), sous réserve que l'horloge système soit réglée avec précision sur l'heure GMT.
- Dans d'autres environnements, l'heure est l'heure locale.

Les valeurs de certains attributs (par exemple, *CurrentQDepth*) changent lorsque le gestionnaire de files d'attente fonctionne. Les modifications apportées à ces attributs n'affectent pas *AlterationTime*.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ALTERATION_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_TIME_LENGTH.

BackoutRequeueQName (MQCHAR48)

Il s'agit du nom de la file d'attente de remise en file d'attente d'annulation excessive. En plus de permettre l'interrogation de sa valeur, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de cet attribut.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Les applications qui s'exécutent dans WebSphere Application Server et celles qui utilisent WebSphere MQ Application Server Facilities utilisent cet attribut pour déterminer l'emplacement des messages qui ont été annulés. Pour toutes les autres applications, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de l'attribut.

WebSphere MQ classes for JMS utilise cet attribut pour déterminer où transférer un message qui a déjà été annulé le nombre maximal de fois indiqué par l'attribut *BackoutThreshold* .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_BACKOUT_REQ_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

BackoutThreshold (MQLONG)

Il s'agit du seuil d'annulation. En plus de permettre l'interrogation de sa valeur, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de cet attribut.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Les applications s'exécutant dans WebSphere Application Server et celles qui utilisent WebSphere MQ Application Server Facilities utiliseront cet attribut pour déterminer si un message doit être annulé. Pour toutes les autres applications, le gestionnaire de files d'attente n'effectue aucune action en fonction de la valeur de l'attribut.

WebSphere MQ classes for JMS utilise cet attribut pour déterminer le nombre de fois où un message peut être annulé avant d'être transféré dans la file d'attente spécifiée par l'attribut *BackoutRequeueQName* .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_BACKOUT_THRESHOLD avec l'appel MQINQ.

BaseQName (MQCHAR48)
Il s'agit du nom d'une file d'attente définie pour le gestionnaire de files d'attente local.

Locale	Modèle	Alias	Eloignée	Cluster
		X		

(Pour plus d'informations sur les noms de file d'attente, voir la zone MQOD- ObjectName.) La file d'attente est de l'un des types suivants:

MQQT_LOCAL
File d'attente locale.

MQQT_REMOTE
Définition locale d'une file d'attente éloignée.

MQQT_CLUSTER
File d'attente de cluster.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_BASE_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

BaseType (MQCFIN)
Type d'objet auquel l'alias est résolu.

Locale	Modèle	Alias	Eloignée	Cluster
		X		

Ses valeurs sont l'une des suivantes :

MQOT_Q
Le type d'objet de base est une file d'attente

MQOT_TOPIC
Le type d'objet de base est une rubrique

CFStrucName (MQCHAR12)
Il s'agit du nom de la structure d'unité de couplage dans laquelle les messages de la file d'attente sont stockés. Le premier caractère du nom est compris entre A et Z, et les caractères restants sont compris entre A et Z, 0 et 9, ou à blanc.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Pour obtenir le nom complet de la structure dans l'unité de couplage, ajoutez en suffixe la valeur de l'attribut de gestionnaire de files d'attente *QSGName* à la valeur de l'attribut de file d'attente *CFStrucName* .

Cet attribut s'applique uniquement aux files d'attente partagées ; il est ignoré si *QSGDisp* ne possède pas la valeur MQQSGD_SHARED.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CF_STRUC_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_CF_STRUC_NAME_LENGTH.

Cet attribut est pris en charge uniquement sur z/OS.

ClusterChannelNom (MQCHAR20)

`ClusterChannelNom` est le nom générique des canaux émetteurs de cluster qui utilisent cette file d'attente comme file d'attente de transmission. L'attribut indique quels canaux émetteurs de cluster envoient des messages à un canal récepteur de cluster à partir de cette file d'attente de transmission de cluster. `ClusterChannelNom` n'est pas pris en charge sous z/OS.

Locale	Modèle	Alias	Eloignée	Cluster
✓	✓			

La configuration de gestionnaire de files d'attente par défaut permet à tous les canaux émetteurs de cluster d'envoyer des messages à partir d'une file d'attente de transmission unique, `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. Vous pouvez changer la configuration par défaut en modifiant l'attribut de gestionnaire de files d'attente `DefClusterXmitQueueType`. La valeur par défaut de l'attribut est `SCTQ`. Vous pouvez la remplacer par `CHANNEL`. Si vous associez l'attribut `DefClusterXmitQueueType` à la valeur `CHANNEL`, chaque canal émetteur de cluster utilisera par défaut une file d'attente de transmission de cluster spécifique, `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`*.

Vous pouvez aussi associer l'attribut de file d'attente de transmission `ClusterChannelName` à un canal émetteur de cluster manuellement. Les messages destinés au gestionnaire de files d'attente connecté par le canal émetteur de cluster sont stockés dans la file d'attente de transmission qui identifie le canal émetteur de cluster. Ils ne sont pas stockés dans la file d'attente de transmission du cluster. Si vous associez l'attribut `ClusterChannelName` à des blancs, le canal utilise la file d'attente de transmission du cluster par défaut lorsqu'il redémarre. La file d'attente par défaut est `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`* ou `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, en fonction de la valeur de l'attribut `DefClusterXmitQueueType` du gestionnaire de files d'attente.

En spécifiant des astérisques, "`*`", dans `ClusterChannelName`, vous pouvez associer une file d'attente de transmission à un ensemble de canaux émetteurs de cluster. Ces astérisques peuvent se trouver au début, à la fin ou à plusieurs endroits dans la chaîne de nom de canal. `ClusterChannelName` ne peut pas comporter plus de 20 caractères : `MQ_CHANNEL_NAME_LENGTH`.

ClusterName (MQCHAR48)

Il s'agit du nom du cluster auquel appartient la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	X

Si la file d'attente appartient à plusieurs clusters, *`ClusterNamelist`* spécifie le nom d'un objet liste de noms qui identifie les clusters et *`ClusterName`* est vide. Au moins l'un des éléments *`ClusterName`* et *`ClusterNamelist`* doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CLUSTER_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_CLUSTER_NAME_LENGTH.

ClusterNamelist (MQCHAR48)

Il s'agit du nom d'un objet liste de noms qui contient les noms des clusters auxquels appartient cette file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	

Si la file d'attente appartient à un seul cluster, l'objet liste de noms ne contient qu'un seul nom. Vous pouvez également utiliser *ClusterName* pour spécifier le nom du cluster, auquel cas *ClusterNamelist* est vide. Au moins l'un des éléments *ClusterName* et *ClusterNamelist* doit être vide.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CLUSTER_NAMELIST avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_NAMELIST_NAME_LENGTH.

CLWLQueuePriority (MQLONG)
Il s'agit de la priorité de la file d'attente de charge de travail du cluster, une valeur comprise entre 0 et 9 représentant la priorité de la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	X

Pour plus d'informations, voir Files d'attente de cluster.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CLWL_Q_PRIORITY avec l'appel MQINQ.

CLWLQueueRank (MQLONG)
Il s'agit du rang de la file d'attente de charge de travail du cluster, une valeur comprise entre 0 et 9 représentant le rang de la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	X

Pour plus d'informations, voir Files d'attente de cluster.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CLWL_Q_RANK avec l'appel MQINQ.

CLWLUseQ (MQLONG)
Cela définit le comportement d'un MQPUT lorsque la file d'attente cible comporte à la fois une instance locale et au moins une instance de cluster distant. Si l'insertion émane d'un canal de cluster, cet attribut ne s'applique pas.

Locale	Modèle	Alias	Eloignée	Cluster
X				

La valeur est l'une des suivantes :

MQCLWL_USEQ_ANY
Utilisez des files d'attente distantes et locales.

MQCLWL_USEQ_LOCAL
N'utilisez pas de files d'attente éloignées.

MQCLWL_USEQ_AS_Q_MGR
Hérite de la définition de MQIA_CLWL_USEQ du gestionnaire de files d'attente.

Pour plus d'informations, voir Files d'attente de cluster.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CLWL_USEQ avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_CLWL_USEQ_LENGTH.

CreationDate (MQCHAR12)
Il s'agit de la date de création de la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X				

Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour obtenir une longueur de 12 octets (par exemple, 2013-09-23¬¬, où ¬¬ représente 2 caractères blancs).

- Sous IBM i, la date de création d'une file d'attente peut être différente de celle de l'entité de système d'exploitation sous-jacente (fichier ou espace utilisateur) qui représente la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CREATION_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_CREATION_DATE_LENGTH.

CreationTime (MQCHAR8)

Il s'agit de l'heure à laquelle la file d'attente a été créée.

Locale	Modèle	Alias	Eloignée	Cluster
X				

Le format de l'heure est HH.MM.SS à l'aide de l'horloge au format 24 heures, avec un zéro non significatif si l'heure est inférieure à 10 (par exemple, 09.10.20).

- Sous z/OS, l'heure est l'heure GMT (Greenwich Mean Time), sous réserve que l'horloge système soit réglée avec précision sur l'heure GMT.
- Dans d'autres environnements, l'heure est l'heure locale.
- Sous IBM i, l'heure de création d'une file d'attente peut être différente de celle de l'entité de système d'exploitation sous-jacente (fichier ou espace utilisateur) qui représente la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_CREATION_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_CREATION_TIME_LENGTH.

CurrentQDepth (MQLONG)

Il s'agit du nombre de messages stockés actuellement dans la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X				

Elle est incrémentée lors d'un appel MQPUT et lors de l'annulation d'un appel MQGET. Il est décrémenté lors d'un appel MQGET non navigant et lors de l'annulation d'un appel MQPUT. Cela a pour effet que le nombre inclut les messages qui ont été placés dans la file d'attente dans une unité d'oeuvre, mais qui n'ont pas encore été validés, même s'ils ne peuvent pas être extraits par l'appel MQGET. De même, il exclut les messages qui ont été extraits dans une unité d'oeuvre à l'aide de l'appel MQGET, mais qui n'ont pas encore été validés.

Ce nombre inclut également les messages dont l'heure d'expiration a été dépassée mais qui n'ont pas encore été supprimés, bien que ces messages ne puissent pas être extraits. Pour plus d'informations, voir MQMD-Zone d'expiration .

Le traitement de l'unité de travail et la segmentation des messages peuvent tous deux entraîner un dépassement de la valeur de *CurrentQDepth* *MaxQDepth*. Toutefois, cela n'affecte pas la possibilité d'extraction des messages ; *tous* les messages de la file d'attente peuvent être extraits à l'aide de l'appel MQGET de manière normale.

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_CURRENT_Q_DEPTH avec l'appel MQINQ.

Réponse DefaultPut(MQLONG)
Indique le type de réponse à utiliser pour les opérations d'insertion dans la file d'attente lorsqu'une application spécifie MQPMO_RESPONSE_AS_Q_DEF.

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X	X	

Ses valeurs sont l'une des suivantes :

MQPRT_SYNC_REPONSE
L'opération d'insertion est émise de manière synchrone et renvoie une réponse.

MQPRT_ASYNC_RESPONSE
L'opération d'insertion est émise de manière asynchrone et renvoie un sous-ensemble de zones MQMD.

DefBind (MQLONG)
Il s'agit de la liaison par défaut utilisée lorsque MQOO_BIND_AS_Q_DEF est spécifié dans l'appel MQOPEN et que la file d'attente est une file d'attente de cluster.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	X

La valeur est l'une des suivantes :

MQBND_BIND_ON_OPEN
Liaison corrigée par l'appel MQOPEN.

MQBND_BIND_NOT_FIXED
Liaison non fixe.

MQBND_BIND_ON_GROUPE
Permet à une application de demander qu'un groupe de messages soit alloué à la même instance de destination. Cette valeur étant nouvelle dans IBM WebSphere MQ Version 7.1, elle ne doit pas être utilisée si l'une des applications ouvrant cette file d'attente se connecte à des gestionnaires de files d'attente IBM WebSphere MQ Version 7.0.1 ou version antérieure.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DEF_BIND avec l'appel MQINQ.

DefinitionType (MQLONG)
Indique comment la file d'attente a été définie.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

La valeur est l'une des suivantes :

MQQDT_PRÉDÉFINI
La file d'attente est une file d'attente permanente créée par l'administrateur système ; seul l'administrateur système peut la supprimer.

Les files d'attente prédéfinies sont créées à l'aide de la commande DEFINE MQSC et peuvent être supprimées uniquement à l'aide de la commande DELETE MQSC. Les files d'attente prédéfinies ne peuvent pas être créées à partir de files d'attente modèles.

Les commandes peuvent être émises par un opérateur ou par un utilisateur autorisé à envoyer un message de commande à la file d'attente d'entrée de commande (pour plus d'informations, voir Attribut QNameCommandInput).

MQQDT_PERMANENT_DYNAMIQUE
La file d'attente est une file d'attente permanente créée par une application émettant un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La

définition de file d'attente modèle avait la valeur MQQDT_PERMANENT_DYNAMIC pour l'attribut *DefinitionType* .

Ce type de file d'attente peut être supprimé à l'aide de l'appel MQCLOSE. Pour plus d'informations, voir «MQCLOSE-Fermer l'objet», à la page 633.

La valeur de l'attribut *QSGDisp* pour une file d'attente dynamique permanente est MQQSGD_Q_MGR.

MQQDT_TEMPORARY_DYNAMIQUE
La file d'attente est une file d'attente temporaire créée par une application qui émet un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La définition de file d'attente modèle avait la valeur MQQDT_TEMPORARY_DYNAMIC pour l'attribut *DefinitionType* .

Ce type de file d'attente est supprimé automatiquement par l'appel MQCLOSE lorsqu'il est fermé par l'application qui l'a créé.

La valeur de l'attribut *QSGDisp* pour une file d'attente dynamique temporaire est MQQSGD_Q_MGR.

MQQDT_PARTAGE_DYNAMIQUE
La file d'attente est une file d'attente permanente partagée qui a été créée par une application émettant un appel MQOPEN avec le nom d'une file d'attente modèle indiquée dans le descripteur d'objet MQOD. La définition de file d'attente modèle avait la valeur MQQDT_SHARED_DYNAMIC pour l'attribut *DefinitionType* .

Ce type de file d'attente peut être supprimé à l'aide de l'appel MQCLOSE. Pour plus d'informations, voir «MQCLOSE-Fermer l'objet», à la page 633.

La valeur de l'attribut *QSGDisp* pour une file d'attente dynamique partagée est MQQSGD_SHARED.

Cet attribut dans une définition de file d'attente modèle n'indique pas comment la file d'attente modèle a été définie, car les files d'attente modèles sont toujours prédéfinies. A la place, la valeur de cet attribut dans la file d'attente modèle est utilisée pour déterminer le *DefinitionType* de chacune des files d'attente dynamiques créées à partir de la définition de file d'attente modèle à l'aide de l'appel MQOPEN.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DEFINITION_TYPE avec l'appel MQINQ.

DefInputOpenOption (MQLONG)

Il s'agit de la méthode par défaut permettant d'ouvrir la file d'attente pour l'entrée.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Elle s'applique si l'option MQOO_INPUT_AS_Q_DEF est spécifiée dans l'appel MQOPEN lorsque la file d'attente est ouverte. La valeur est l'une des suivantes :

MQOO_INPUT_EXCLUSIVE
Ouvrez la file d'attente pour obtenir les messages avec un accès exclusif.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel échoue avec le code anomalie MQRC_OBJECT_IN_USE si la file d'attente est actuellement ouverte par cette application ou par une autre application pour l'entrée d'un type quelconque (MQOO_INPUT_SHARED ou MQOO_INPUT_EXCLUSIVE).

MQOO_INPUT_SHARED
Ouvrez la file d'attente pour obtenir les messages avec accès partagé.

La file d'attente est ouverte pour être utilisée avec les appels MQGET suivants. L'appel peut aboutir si la file d'attente est actuellement ouverte par cette application ou par une autre application avec MQOO_INPUT_SHARED, mais échoue avec le code anomalie MQRC_OBJECT_IN_USE si la file d'attente est actuellement ouverte avec MQOO_INPUT_EXCLUSIVE.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DEF_INPUT_OPEN_OPTION avec l'appel MQINQ.

DefPersistence (MQLONG)

Il s'agit de la persistance par défaut des messages dans la file d'attente. Elle s'applique si MQPER_PERSISTENCE_AS_Q_DEF est spécifié dans le descripteur de message lors de l'insertion du message.

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X	X	X

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la persistance par défaut est extraite de la valeur de cet attribut dans la *première* définition du chemin au moment de l'appel MQPUT ou MQPUT1 . Possibilités :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName*)

La valeur est l'une des suivantes :

MQPER_PERSISTANT
Le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente. Les messages persistants ne peuvent pas être placés sur:

- Files d'attente dynamiques temporaires
- Files d'attente partagées qui sont mappées à un objet CFSTRUCT au niveau CFLEVEL (2) ou à un niveau inférieur, ou où l'objet CFSTRUCT est défini en tant que RECOVER (NO).

Les messages persistants peuvent être placés dans des files d'attente dynamiques permanentes et dans des files d'attente prédéfinies.

MQPER_NON_PERSISTENT
Le message ne survit normalement pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente. Cela s'applique même si une copie intacte du message est trouvée sur la mémoire secondaire lors du redémarrage d'un gestionnaire de files d'attente.

Dans le cas de files d'attente partagées, les messages non persistants *survivent* aux redémarrages des gestionnaires de files d'attente dans le groupe de partage de files d'attente, mais ne survivent pas aux échecs de l'unité de couplage utilisée pour stocker les messages dans les files d'attente partagées.

Les messages persistants et non persistants peuvent exister dans la même file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DEF_PERSISTENCE avec l'appel MQINQ.

DefPriority (MQLONG)

Il s'agit de la priorité par défaut pour les messages de la file d'attente. Cela s'applique si MQPRI_PRIORITY_AS_Q_DEF est spécifié dans le descripteur de message lorsque le message est inséré dans la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X	X	X

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, la priorité par défaut du message est extraite de la valeur de cet attribut dans la *première* définition du chemin au moment de l'opération d'insertion. Possibilités :

- Une file d'attente alias
- Une file d'attente locale
- Définition locale d'une file d'attente éloignée
- Un alias de gestionnaire de files d'attente
- Une file d'attente de transmission (par exemple, la file d'attente *DefXmitQName*)

La manière dont un message est placé dans une file d'attente dépend de la valeur de l'attribut *MsgDeliverySequence* de la file d'attente:

- Si l'attribut *MsgDeliverySequence* est MQMDS_PRIORITY, la position logique à laquelle un message est placé dans la file d'attente dépend de la valeur de la zone *Priority* dans le descripteur de message.
- Si l'attribut *MsgDeliverySequence* est MQMDS_FIFO, les messages sont placés dans la file d'attente comme s'ils avaient une priorité égale à *DefPriority* de la file d'attente résolue, quelle que soit la valeur de la zone *Priority* dans le descripteur de message. Toutefois, la zone *Priority* conserve la valeur spécifiée par l'application qui a inséré le message. Pour plus d'informations, voir Attribut de séquenceMsgDelivery .

Les priorités sont comprises entre zéro (valeur la plus faible) et *MaxPriority* (valeur la plus élevée) ; voir AttributMaxPriority.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DEF_PRIORITY avec l'appel MQINQ.

DefReadAhead (MQLONG)

Indique le comportement de lecture anticipée par défaut pour les messages non persistants distribués au client.

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X		

DefReadAhead peut être défini sur l'une des valeurs suivantes:

MQREADA_NO
Les messages non persistants ne sont pas envoyés au client avant qu'une application ne les demande. Un maximum d'un message non persistant peut être perdu si le client se termine de manière anormale.

MQREADA_YES
Les messages non persistants sont envoyés au client avant qu'une application ne les demande. Les messages non persistants peuvent être perdus si le client se termine de manière anormale ou si le client ne consomme pas tous les messages qu'il envoie.

MQREADA_DISABLED
La lecture anticipée des messages non persistants n'est pas activée pour cette file d'attente. Les messages ne sont pas envoyés à l'avance au client, que la lecture anticipée soit demandée ou non par l'application client.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DEF_READ_AHEAD avec l'appel MQINQ.

DefPResp (MQLONG)

L'attribut de type de réponse d'insertion par défaut (DEFPRESP) définit la valeur utilisée par les applications lorsque le type PutResponsedans MQPMO a été défini sur MQPMO_RESPONSE_AS_Q_DEF. Cet attribut est valide pour tous les types de file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
Oui	Oui	Oui	Oui	Oui

La valeur est l'une des suivantes :

SYNC
L'opération d'insertion est émise de manière synchrone et renvoie une réponse.

ASYNC
L'opération d'insertion est émise de manière asynchrone et renvoie un sous-ensemble de zones MQMD.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DEF_PUT_RESPONSE_TYPE avec l'appel MQINQ.

DistLists (MQLONG)

Indique si les messages de liste de distribution peuvent être placés dans la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Un agent MCA définit l'attribut pour indiquer au gestionnaire de files d'attente local si le gestionnaire de files d'attente à l'autre extrémité du canal prend en charge les listes de distribution. Ce dernier gestionnaire de files d'attente (appelé gestionnaire de files d'attente *de partenariat*) est celui qui reçoit ensuite le message, une fois qu'il a été supprimé de la file d'attente de transmission locale par un agent MCA émetteur.

L'agent MCA émetteur définit l'attribut chaque fois qu'il établit une connexion à l'agent MCA récepteur sur le gestionnaire de files d'attente partenaire. Ainsi, l'agent MCA émetteur peut amener le gestionnaire de files d'attente local à placer dans la file d'attente de transmission uniquement les messages que le gestionnaire de files d'attente partenaire peut traiter correctement.

Cet attribut est principalement utilisé avec les files d'attente de transmission, mais le traitement décrit est effectué quelle que soit l'utilisation définie pour la file d'attente (voir Attribut d'utilisation).

La valeur est l'une des suivantes :

MQDL_SUPPORTED
Les messages de liste de distribution peuvent être stockés dans la file d'attente et transmis au gestionnaire de files d'attente partenaire sous cette forme. Cela réduit la quantité de traitement requise pour envoyer le message à plusieurs destinations.

MQDL_NOT_SUPPORTED
Les messages de liste de distribution ne peuvent pas être stockés dans la file d'attente car le gestionnaire de files d'attente partenaire ne prend pas en charge les listes de distribution. Si une application insère un message de liste de distribution et que ce message doit être placé dans cette file d'attente, le gestionnaire de files d'attente fractionne le message de liste de distribution et place les messages individuels dans la file d'attente à la place. Cela augmente la quantité de traitement requise pour envoyer le message à plusieurs destinations, mais garantit que les messages sont traités correctement par le gestionnaire de files d'attente partenaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_DIST_LISTS avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

Cet attribut n'est pas pris en charge sur z/OS.

HardenGetAnnulation (MQLONG)

Pour chaque message, le nombre de fois où le message est extrait par un appel MQGET au sein d'une unité de travail est comptabilisé et cette unité de travail est ensuite annulée.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Ce nombre est disponible dans la zone *BackoutCount* du descripteur de message une fois l'appel MQGET terminé.

Le nombre d'annulations de message survit aux redémarrages du gestionnaire de files d'attente. Toutefois, pour s'assurer que le nombre est exact, les informations doivent être *renforcées* (enregistrées sur le disque ou sur une autre unité de stockage permanente) chaque fois qu'un appel MQGET extrait un message dans une unité de travail pour cette file d'attente. Si cette opération n'est pas effectuée, le gestionnaire de files d'attente échoue et les appels MQGET sont annulées, le nombre peut être incrémenté ou non.

Toutefois, les informations de renforcement pour chaque appel MQGET dans une unité de travail imposent des coûts de traitement supplémentaires. Par conséquent, définissez l'attribut *HardenGetBackout* sur MQQA_BACKOUT_HARDENED uniquement s'il est essentiel que le nombre soit exact.

Sur les systèmes IBM i, UNIX et Windows, le nombre d'annulations de message est toujours renforcé, quelle que soit la valeur de cet attribut.

Valeurs possibles :

MQQA_BACKOUT_HARDENED
Le renforcement est utilisé pour s'assurer que le nombre d'annulations pour les messages de cette file d'attente est exact.

MQQA_BACKOUT_NOT_HARDENED
Le renforcement n'est pas utilisé pour s'assurer que le nombre d'annulations pour les messages de cette file d'attente est correct. Le nombre peut donc être inférieur à ce qu'il devrait être.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_HARDEN_GET_BACKOUT avec l'appel MQINQ.

IndexType (MQLONG)

Indique le type d'index que le gestionnaire de files d'attente gère pour les messages de la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Le type d'index requis dépend de la manière dont l'application extrait les messages et du fait que la file d'attente est une file d'attente partagée ou une file d'attente non partagée (voir Attribut QSGDisp). Les valeurs suivantes sont possibles pour *IndexType*:

MQIT_AUCUN
Aucun index n'est géré par le gestionnaire de files d'attente pour cette file d'attente. Utilisez cette valeur pour les files d'attente qui sont généralement traitées séquentiellement, c'est-à-dire sans utiliser de critères de sélection dans l'appel MQGET.

ID_MSG_MQ
Le gestionnaire de files d'attente gère un index qui utilise les identificateurs des messages de la file d'attente. Utilisez cette valeur pour les files d'attente dans lesquelles l'application extrait généralement les messages en utilisant l'identificateur de message comme critère de sélection dans l'appel MQGET.

ID_CORREL_MQ
Le gestionnaire de files d'attente gère un index qui utilise les identificateurs de corrélation des messages de la file d'attente. Utilisez cette valeur pour les files d'attente dans lesquelles l'application extrait généralement les messages en utilisant l'identificateur de corrélation comme critère de sélection dans l'appel MQGET.

JETON_MSG_MQIT_TOKEN
Le gestionnaire de files d'attente gère un index qui utilise les jetons de message des messages de la file d'attente à utiliser avec les fonctions WLM (Workload Manager) de z/OS.

Vous *devez* spécifier cette option pour les files d'attente gérées par WLM ; ne la spécifiez pour aucun autre type de file d'attente. En outre, n'utilisez pas cette valeur pour une file d'attente dans laquelle une application n'utilise pas les fonctions du gestionnaire de charge de travail z/OS , mais extrait des messages à l'aide du jeton de message comme critère de sélection dans l'appel MQGET.

ID_GROUPE_MQIT

Le gestionnaire de files d'attente gère un index qui utilise les identificateurs de groupe des messages de la file d'attente. Cette valeur *doit* être utilisée pour les files d'attente dans lesquelles l'application extrait des messages à l'aide de l'option MQGMO_LOGICAL_ORDER sur l'appel MQGET.

Une file d'attente avec ce type d'index ne peut pas être une file d'attente de transmission. Une file d'attente partagée avec ce type d'index doit être définie pour être mappée à un objet CFSTRUCT au niveau CFLEVEL (3) ou CFLEVEL (4).

Remarque :

1. L'ordre physique des messages dans une file d'attente avec le type d'index MQIT_GROUP_ID n'est pas défini, car la file d'attente est optimisée pour une extraction efficace des messages à l'aide de l'option MQGMO_LOGICAL_ORDER sur l'appel MQGET. Cela signifie que l'ordre physique des messages n'est généralement pas l'ordre dans lequel les messages sont arrivés dans la file d'attente.
2. Si une file d'attente MQIT_GROUP_ID a une valeur *MsgDeliverySequence* de MQMDS_PRIORITY, le gestionnaire de files d'attente utilise les priorités de message 0 et 1 pour optimiser l'extraction des messages dans l'ordre logique. Par conséquent, le premier message d'un groupe ne doit pas avoir une priorité égale à zéro ou à un ; si tel est le cas, le message est traité comme s'il avait une priorité égale à deux. La zone *Priority* de la structure MQMD n'est pas modifiée.

Pour plus d'informations sur les groupes de messages, voir la description des options de groupe et de segment dans la zone MQGMO-Options.

Le type d'index qui doit être utilisé dans divers cas est indiqué dans Tableau 574, à la page 840 et Tableau 575, à la page 841.

Tableau 574. Valeurs suggérées ou requises pour le type d'index de file d'attente lorsque MQGMO_LOGICAL_ORDER n'est pas spécifié

Critères de sélection lors de l'appel MQGET	**Type d'index pour la file d'attente non partagée**	**Type d'index pour la file d'attente partagée**
Aucun	Tous	Tous
Sélection à l'aide d'un identificateur :		
Identificateur de message	MQIT_MSG_ID suggéré	MQIT_NONE ou MQIT_MSG_ID requis ; MQIT_MSG_ID suggéré
Identificateur de corrélation	MQIT_CORREL_ID suggéré	MQIT_CORREL_ID requis
Identificateur de groupe	MQIT_GROUP_ID suggéré	MQIT_GROUP_ID requis
Sélection à l'aide de deux identificateurs :		
Identificateur de message plus identificateur de corrélation	MQIT_MSG_ID ou MQIT_CORREL_ID suggéré	MQIT_NONE ou MQIT_MSG_ID ou MQIT_CORREL_ID requis (Pour plus d'efficacité, il est suggéré que le type d'index soit choisi pour correspondre à la zone MQMD qui aura les clés les plus distinctes)
Identificateur de message plus identificateur de groupe	MQIT_MSG_ID ou MQIT_GROUP_ID suggéré	Non pris en charge
Identificateur de corrélation plus identificateur de groupe	MQIT_CORREL_ID ou MQIT_GROUP_ID suggéré	Non pris en charge

Tableau 574. Valeurs suggérées ou requises pour le type d'index de file d'attente lorsque MQGMO_LOGICAL_ORDER n'est pas spécifié (suite)

Critères de sélection lors de l'appel MQGET	Type d'index pour la file d'attente non partagée	Type d'index pour la file d'attente partagée
Sélection à l'aide de trois identificateurs :		
Identificateur de message plus identificateur de corrélation plus identificateur de groupe	MQIT_MSG_ID, MQIT_CORREL_ID ou MQIT_GROUP_ID suggéré	Non pris en charge
Sélection à l'aide de critères liés aux groupes :		
Identificateur de groupe plus numéro de séquence de message	MQIT_GROUP_ID requis	MQIT_GROUP_ID requis
Numéro de séquence du message (doit être 1)	MQIT_GROUP_ID requis	MQIT_GROUP_ID requis
Sélection à l'aide d'un jeton de message :		
Jeton de message à utiliser par l'application	Ne pas utiliser MQIT_MSG_TOKEN	
Jeton de message pour l'utilisation de WLM	MQIT_MSG_TOKEN requis	Non pris en charge

Tableau 575. Valeurs suggérées ou requises pour le type d'index de file d'attente lorsque MQGMO_LOGICAL_ORDER est spécifié

Critères de sélection lors de l'appel MQGET	Type d'index pour la file d'attente non partagée	Type d'index pour la file d'attente partagée
Aucun	MQIT_GROUP_ID requis	MQIT_GROUP_ID requis
Sélection à l'aide d'un identificateur :		
Identificateur de message	MQIT_GROUP_ID requis	Non pris en charge
Identificateur de corrélation	MQIT_GROUP_ID requis	Non pris en charge
Identificateur de groupe	MQIT_GROUP_ID requis	MQIT_GROUP_ID requis
Sélection à l'aide de deux identificateurs :		
Identificateur de message plus identificateur de corrélation	MQIT_GROUP_ID requis	Non pris en charge
Identificateur de message plus identificateur de groupe	MQIT_GROUP_ID requis	Non pris en charge
Identificateur de corrélation plus identificateur de groupe	MQIT_GROUP_ID requis	Non pris en charge
Sélection à l'aide de trois identificateurs :		
Identificateur de message plus identificateur de corrélation plus identificateur de groupe	MQIT_GROUP_ID requis	Non pris en charge

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_INDEX_TYPE avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sur z/OS.

InhibitGet (MQLONG)

Cette option contrôle si les opérations d'extraction de cette file d'attente sont autorisées.

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X		

Si la file d'attente est une file d'attente alias, les opérations d'extraction doivent être autorisées pour l'alias et la file d'attente de base au moment de l'opération d'extraction, pour que l'appel MQGET aboutisse. La valeur est l'une des suivantes :

MQQA_GET_INHIBÉE
Les opérations d'extraction sont interdites.

Les appels MQGET échouent avec le code anomalie MQRC_GET_INHIBÉ. Cela inclut les appels MQGET qui spécifient MQGMO_BROWSE_FIRST ou MQGMO_BROWSE_NEXT.

Remarque : Si un appel MQGET fonctionnant dans une unité de travail aboutit, la modification de la valeur de l'attribut *InhibitGet* par la suite en MQQA_GET_INHIBÉE n'empêche pas la validation de l'unité de travail.

MQQA_GET_ALLOWED
Les opérations d'extraction sont autorisées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_INHIBIT_GET avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

InhibitPut (MQLONG)

Indique si les opérations d'insertion pour cette file d'attente sont autorisées.

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X	X	X

S'il existe plusieurs définitions dans le chemin de résolution de nom de file d'attente, les opérations d'insertion doivent être autorisées pour *chaque* définition du chemin (y compris les définitions d'alias de gestionnaire de files d'attente) au moment de l'opération d'insertion, pour que l'appel MQPUT ou MQPUT1 aboutisse. La valeur est l'une des suivantes :

MQQA_PUT_INHIBÉ
Les opérations d'insertion sont interdites.

Les appels MQPUT et MQPUT1 échouent avec le code anomalie MQRC_PUT_INHIBÉ.

Remarque : Si un appel MQPUT fonctionnant dans une unité de travail aboutit, la modification de la valeur de l'attribut *InhibitPut* par la suite en MQQA_PUT_INHIBÉE n'empêche pas la validation de l'unité de travail.

MQQA_PUT_ALLOWED
Les opérations d'insertion sont autorisées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_INHIBIT_PUT avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

InitiationQName (MQCHAR48)

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local ; la file d'attente doit être de type MQQT_LOCAL.

Locale	Modèle	Alias	Eloignée	Cluster
X				

Le gestionnaire de files d'attente envoie un message de déclenchement à la file d'attente d'initialisation lorsque le démarrage de l'application est requis suite à l'arrivée d'un message dans la file d'attente à laquelle appartient cet attribut. La file d'attente d'initialisation doit être surveillée par une application de moniteur de déclenchement qui démarre l'application appropriée après la réception du message de déclenchement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_INITIATION_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

Longueur MaxMsg(MQLONG)

Il s'agit d'une limite supérieure pour la longueur du message *physique* le plus long pouvant être placé dans la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Toutefois, comme l'attribut de file d'attente *MaxMsgLength* peut être défini indépendamment de l'attribut de gestionnaire de files d'attente *MaxMsgLength* , la limite supérieure réelle de la longueur du message physique le plus long pouvant être placé dans la file d'attente est la plus faible de ces deux valeurs.

Si le gestionnaire de files d'attente prend en charge la segmentation, il est possible qu'une application place un message *logique* plus long que le moindre des deux attributs *MaxMsgLength* , mais uniquement si l'application spécifie l'indicateur MQMF_SEGMENTATION_ALLOWED dans MQMD. Si cet indicateur est spécifié, la limite supérieure de la longueur d'un message logique est de 999 999 999 999 octets, mais les contraintes de ressources imposées par le système d'exploitation ou par l'environnement dans lequel l'application est exécutée entraînent une limite inférieure.

Une tentative de placement dans la file d'attente d'un message trop long échoue avec l'un des codes anomalie suivants:

- MQRC_MSG_TOO_BIG_FOR_Q si le message est trop grand pour la file d'attente
- MQRC_MSG_TOO_BIG_FOR_Q_MGR si le message est trop grand pour le gestionnaire de files d'attente, mais pas trop grand pour la file d'attente

La limite inférieure de l'attribut *MaxMsgLength* est zéro ; la limite supérieure est de 100 Mo (104 857 600 octets).

Pour plus d'informations, voir le paramètre MQPUT- BufferLength.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MAX_MSG_LENGTH avec l'appel MQINQ.

MaxQDepth (MQLONG)

Il s'agit de la limite supérieure définie pour le nombre de messages physiques pouvant exister simultanément dans la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Une tentative d'insertion d'un message dans une file d'attente qui contient déjà des messages *MaxQDepth* échoue avec le code anomalie MQRC_Q_FULL.

Le traitement des unités de travail et la segmentation des messages peuvent entraîner un nombre réel de messages physiques dans la file d'attente supérieur à *MaxQDepth*. Toutefois, cela n'affecte pas la possibilité d'extraction des messages ; *tous* les messages de la file d'attente peuvent être extraits à l'aide de l'appel MQGET.

La valeur de cet attribut est supérieure ou égale à zéro. La limite supérieure est déterminée par l'environnement:

- Sous AIX, HP-UX, z/OS, Solaris, Linuxet Windows, la valeur ne peut pas dépasser 999 999 999.
- Sous IBM i, la valeur ne peut pas dépasser 640 000.

Remarque : L'espace de stockage disponible pour la file d'attente peut être épuisé même s'il y a moins de messages *MaxQDepth* dans la file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MAX_Q_DEPTH avec l'appel MQINQ.

Séquence MsgDelivery(MQLONG)

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Détermine l'ordre dans lequel l'appel MQGET renvoie des messages à l'application:

MQMDS_FIFO
Les messages sont renvoyés dans l'ordre FIFO (premier entré, premier sorti).

Un appel MQGET renvoie le *premier* message qui répond aux critères de sélection spécifiés sur l'appel, quelle que soit la priorité du message.

MQMDS_PRIORITE
Les messages sont renvoyés par ordre de priorité.

Un appel MQGET renvoie le message *de priorité la plus élevée* qui répond aux critères de sélection spécifiés dans l'appel. Dans chaque niveau de priorité, les messages sont renvoyés dans l'ordre FIFO (premier entré, premier sorti).

- Sous z/OS, si la file d'attente a un *IndexType* MQIT_GROUP_ID, l'attribut *MsgDeliverySequence* spécifie l'ordre dans lequel les groupes de messages sont renvoyés à l'application. La séquence particulière dans laquelle les groupes sont renvoyés est déterminée par la position ou la priorité du premier message dans chaque groupe. L'ordre physique des messages dans la file d'attente n'est pas défini, car la file d'attente est optimisée pour l'extraction efficace des messages à l'aide de l'option MQGMO_LOGICAL_ORDER sur l'appel MQGET.
- Sous z/OS, si *IndexType* est MQIT_GROUP_ID et *MsgDeliverySequence* est MQMDS_PRIORITY, le gestionnaire de files d'attente utilise les priorités de message zéro et un pour optimiser l'extraction des messages dans l'ordre logique. Par conséquent, le premier message d'un groupe ne doit pas avoir une priorité égale à zéro ou à un ; si tel est le cas, le message est traité comme s'il avait une priorité égale à deux. La zone *Priority* de la structure MQMD n'est pas modifiée.

Si les attributs appropriés sont modifiés alors que la file d'attente contient des messages, la séquence de distribution est la suivante:

- L'ordre dans lequel les messages sont renvoyés par l'appel MQGET est déterminé par les valeurs des attributs *MsgDeliverySequence* et *DefPriority* en vigueur pour la file d'attente au moment où le message arrive dans la file d'attente:
  - Si *MsgDeliverySequence* est MQMDS_FIFO à l'arrivée du message, le message est placé dans la file d'attente comme si sa priorité était *DefPriority*. Cela n'affecte pas la valeur de la zone *Priority* dans le descripteur de message du message ; cette zone conserve la valeur qu'elle avait lors de la première insertion du message.
  - Si *MsgDeliverySequence* est MQMDS_PRIORITY à l'arrivée du message, le message est placé dans la file d'attente à l'emplacement approprié à la priorité donnée par la zone *Priority* dans le descripteur de message.

  Si la valeur de l'attribut *MsgDeliverySequence* est modifiée alors qu'il y a des messages dans la file d'attente, l'ordre des messages dans la file d'attente n'est pas modifié.

  Si la valeur de l'attribut *DefPriority* est modifiée alors qu'il y a des messages dans la file d'attente, les messages ne sont pas nécessairement distribués dans l'ordre FIFO, même si l'attribut *MsgDeliverySequence* est défini sur MQMDS_FIFO; ceux qui ont été placés dans la file d'attente à la priorité la plus élevée sont distribués en premier.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MSG_DELIVERY_SEQUENCE avec l'appel MQINQ.

NonPersistentMessageClass (MQLONG)

Objectif de fiabilité pour les messages non persistants.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Indique les circonstances dans lesquelles les messages non persistants placés dans cette file d'attente sont supprimés:

MQNPM_CLASS_NORMAL
Les messages non persistants sont limités à la durée de vie de la session du gestionnaire de files d'attente ; ils sont supprimés en cas de redémarrage du gestionnaire de files d'attente. Cette valeur est valide uniquement pour les files d'attente non partagées et correspond à la valeur par défaut.

MQNPM_CLASS_HIGH
Le gestionnaire de files d'attente tente de conserver les messages non persistants pendant la durée de vie de la file d'attente. Des messages non persistants peuvent encore être perdus en cas d'échec. Cette valeur est appliquée aux files d'attente partagées.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_NPM_CLASS avec l'appel MQINQ.

Nombre OpenInput(MQLONG)

Nombre de descripteurs actuellement valides pour supprimer des messages de la file d'attente à l'aide de l'appel MQGET.

Locale	Modèle	Alias	Eloignée	Cluster
X				

Il s'agit du nombre total de descripteurs connus du gestionnaire de files d'attente *local* . Si la file d'attente est une file d'attente partagée, le nombre n'inclut pas les ouvertures pour entrée qui ont été effectuées pour la file d'attente dans d'autres gestionnaires de files d'attente du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

Le nombre inclut les descripteurs dans lesquels une file d'attente alias qui se résout en cette file d'attente a été ouverte en entrée. Le nombre n'inclut pas les descripteurs où la file d'attente a été ouverte pour les actions qui n'incluent pas d'entrée (par exemple, une file d'attente ouverte uniquement pour l'exploration).

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_OPEN_INPUT_COUNT avec l'appel MQINQ.

Nombre OpenOutput(MQLONG)

Il s'agit du nombre de descripteurs actuellement valides pour ajouter des messages à la file d'attente au moyen de l'appel MQPUT.

Locale	Modèle	Alias	Eloignée	Cluster
X				

Il s'agit du nombre total de descripteurs connus du gestionnaire de files d'attente *local* ; il n'inclut pas les ouvertures pour les sorties qui ont été effectuées pour cette file d'attente sur les gestionnaires de files d'attente éloignées. Si la file d'attente est une file d'attente partagée, le nombre n'inclut pas les

ouvertures pour sortie qui ont été effectuées pour la file d'attente dans d'autres gestionnaires de files d'attente du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente local.

Le nombre inclut les descripteurs dans lesquels une file d'attente alias qui se résout en cette file d'attente a été ouverte pour la sortie. Le nombre n'inclut pas les descripteurs où la file d'attente a été ouverte pour les actions qui n'incluent pas de sortie (par exemple, une file d'attente ouverte uniquement pour l'interrogation).

La valeur de cet attribut fluctue au fur et à mesure que le gestionnaire de files d'attente fonctionne.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_OPEN_OUTPUT_COUNT avec l'appel MQINQ.

ProcessName (MQCHAR48)

Il s'agit du nom d'un objet de processus défini sur le gestionnaire de files d'attente local. L'objet processus identifie un programme qui peut traiter la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_PROCESS_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_PROCESS_NAME_LENGTH.

PropertyControl (MQLONG)

Indique comment les propriétés de message sont traitées pour les messages extraits des files d'attente à l'aide de l'appel MQGET avec l'option MQGMO_PROPERTIES_AS_Q_DEF.

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X		

La valeur est l'une des suivantes :

MQPROP_ALL
Toutes les propriétés du message sont incluses avec le message lorsqu'il est distribué à l'application. Les propriétés, à l'exception de celles du descripteur de message (ou extension), sont placées dans un ou plusieurs en-têtes MQRFH2 dans les données du message. Si un descripteur de message est fourni, le comportement consiste à renvoyer les propriétés dans le descripteur de message.

COMPATIBILITE_MQPROP_COMPATIBILITÉ
Si le message contient une propriété avec un préfixe mcd., jms., usr. ou mqext., Toutes les propriétés de message sont distribuées à l'application dans un en-tête MQRFH2 . Sinon, toutes les propriétés du message, à l'exception de celles du descripteur de message (ou extension), sont supprimées et ne sont plus accessibles à l'application. Il s'agit de la valeur par défaut ; elle permet aux applications qui attendent les propriétés liées à JMS dans un en-tête MQRFH2 dans les données du message de continuer à travailler sans modification. Si un descripteur de message est fourni, le comportement consiste à renvoyer les propriétés dans le descripteur de message.

MQPROP_FORCE_MQRFH2
Les propriétés sont toujours renvoyées dans les données du message dans un en-tête MQRFH2, indépendamment du fait que l'application indique un descripteur de message. Un descripteur de message valide fourni dans la zone MsgHandle de la structure MQGMO sur l'appel MQGET est ignoré. Les propriétés du message ne sont pas accessibles via l'identificateur de message.

MQPROP_NONE
Toutes les propriétés du message, à l'exception de celles du descripteur de message (ou de l'extension), sont supprimées du message avant que celui-ci ne soit distribué à l'application. Si un descripteur de message est fourni, le comportement consiste à renvoyer les propriétés dans le descripteur de message.

Ce paramètre est applicable aux files d'attente locales, d'alias et de modèle. Pour déterminer sa valeur, utilisez le sélecteur MQIA_PROPERTY_CONTROL avec l'appel MQINQ.

Evénement QDepthHigh(MQLONG)

Indique si les événements Longueur élevée de la file d'attente sont générés.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Un événement Longueur élevée de file d'attente indique qu'une application a inséré un message dans une file d'attente, ce qui a entraîné un nombre de messages dans la file d'attente supérieur ou égal au seuil haut de longueur de file d'attente (voir l'attribut *QDepthHighLimit*).

Remarque : La valeur de cet attribut peut changer dynamiquement.

La valeur est l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_Q_DEPTH_HIGH_EVENT avec l'appel MQINQ.

Cet attribut est pris en charge sur z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

Limite QDepthHigh(MQLONG)

Il s'agit du seuil auquel le nombre de lignes de la file d'attente est comparé pour générer un événement Longueur élevée de la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Cet événement indique qu'une application a inséré un message dans une file d'attente, ce qui a entraîné un nombre de messages dans la file d'attente supérieur ou égal au seuil maximal de longueur de la file d'attente. Voir Attribut d'événementQDepthHigh.

La valeur est exprimée en pourcentage de la longueur maximale de la file d'attente (attribut*MaxQDepth*) et est supérieure ou égale à 0 et inférieure ou égale à 100. La valeur par défaut est 80.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_Q_DEPTH_HIGH_LIMIT avec l'appel MQINQ.

Cet attribut est pris en charge sur z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

Evénement QDepthLow(MQLONG)

Cette option contrôle si les événements Longueur faible de la file d'attente sont générés.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Un événement Longueur basse de file d'attente indique qu'une application a extrait un message d'une file d'attente et que le nombre de messages de la file d'attente est devenu inférieur ou égal au seuil bas de longueur de la file d'attente (voir QDepthLow).

Remarque : La valeur de cet attribut peut changer dynamiquement.

La valeur est l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_Q_DEPTH_LOW_EVENT avec l'appel MQINQ.

Cet attribut est pris en charge sur z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

Limite QDepthLow(MQLONG)

Il s'agit du seuil auquel le nombre de lignes de la file d'attente est comparé pour générer un événement Longueur faible de la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Cet événement indique qu'une application a extrait un message d'une file d'attente et que le nombre de messages de la file d'attente est inférieur ou égal au seuil bas de longueur de la file d'attente. Voir Attribut d'événementQDepthLow.

La valeur est exprimée en pourcentage de la longueur maximale de la file d'attente (attribut*MaxQDepth*) et est supérieure ou égale à 0 et inférieure ou égale à 100. La valeur par défaut est 20.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_Q_DEPTH_LOW_LIMIT avec l'appel MQINQ.

Cet attribut est pris en charge sur z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

Evénement QDepthMax(MQLONG)

Cette option contrôle si les événements File d'attente saturée sont générés. Un événement File d'attente saturée indique qu'une insertion dans une file d'attente a été rejetée car la file d'attente est saturée, c'est-à-dire que la longueur de la file d'attente a déjà atteint sa valeur maximale.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Remarque : La valeur de cet attribut peut changer dynamiquement.

La valeur est l'une des suivantes :

MQEVR_DISABLED
Génération de rapports d'événements désactivée.

MQEVR_ENABLED
Génération de rapports d'événements activée.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_Q_DEPTH_MAX_EVENT avec l'appel MQINQ.

Cet attribut est pris en charge sur z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

QDesc (MQCHAR64)

Utilisez cette zone pour les commentaires descriptifs.

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X	X	X

Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, la zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

Remarque : Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente *CodedCharSetId*), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_Q_DESC avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_DESC_LENGTH.

QName (MQCHAR48)

Il s'agit du nom d'une file d'attente définie sur le gestionnaire de files d'attente local.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	X

Toutes les files d'attente définies sur un gestionnaire de files d'attente partagent le même espace de nom de file d'attente. Par conséquent, une file d'attente MQQT_LOCAL et une file d'attente MQQT_ALIAS ne peuvent pas avoir le même nom.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

QServiceInterval (MQLONG)

Il s'agit de l'intervalle de service utilisé à des fins de comparaison pour générer des événements d'intervalle de service élevé et d'intervalle de service OK.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Voir Attribut d'événementQServiceInterval.

La valeur est exprimée en millisecondes et est supérieure ou égale à zéro et inférieure ou égale à 999 999 999.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_Q_SERVICE_INTERVAL avec l'appel MQINQ.

Cet attribut est pris en charge sur z/OS, mais l'appel MQINQ ne peut pas être utilisé pour déterminer sa valeur.

Evénement QServiceInterval(MQLONG)

Indique si les événements Intervalle de service élevé ou Intervalle de service OK sont générés.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

- Un événement Intervalle de service élevé est généré lorsqu'une vérification indique qu'aucun message n'a été extrait de la file d'attente pendant au moins le temps indiqué par l'attribut *QServiceInterval* .
- Un événement d'intervalle de service OK est généré lorsqu'une vérification indique que des messages ont été extraits de la file d'attente dans le délai indiqué par l'attribut *QServiceInterval* .

Remarque : La valeur de cet attribut peut changer dynamiquement.

La valeur est l'une des suivantes :

MQQSIE_HIGH
Evénements d'intervalle élevé du service de file d'attente activés.

- Les événements d'intervalle de service de file d'attente élevé sont **activés** et
- Les événements d'intervalle de service de file d'attente OK sont **désactivés**.

MQQSIE_OK
Evénements d'intervalle de service de file d'attente OK activés.

- Les événements d'intervalle de service de file d'attente élevé sont **désactivés** et
- Les événements d'intervalle de service de file d'attente OK sont **activés**.

MQQSIE_NONE
Aucun événement d'intervalle de service de file d'attente n'est activé.

- Les événements d'intervalle de service de file d'attente élevé sont **désactivés** et
- Les événements d'intervalle de service de file d'attente OK sont également **désactivés**.

Pour les files d'attente partagées, la valeur de cet attribut est ignorée ; la valeur MQQSIE_NONE est prise en compte.

Pour plus d'informations sur les événements, voir Surveillance des événements .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_Q_SERVICE_INTERVAL_EVENT avec l'appel MQINQ.

Sous z/OS, vous ne pouvez pas utiliser l'appel MQINQ pour déterminer la valeur de cet attribut.

QSGDisp (MQLONG)
Indique la disposition de la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	

La valeur est l'une des suivantes :

MQQSGD_Q_DIR
L'objet possède une disposition de gestionnaire de files d'attente. Cela signifie que la définition d'objet est connue uniquement du gestionnaire de files d'attente local ; la définition n'est pas connue des autres gestionnaires de files d'attente du groupe de partage de files d'attente.

Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut avoir un objet ayant le même nom et le même type que l'objet en cours, mais il s'agit d'objets distincts et il n'y a pas de corrélation entre eux. Leurs attributs ne sont pas contraints d'être identiques les uns aux autres.

MQQSGD_COPY
L'objet est une copie locale d'une définition d'objet maître qui existe dans le référentiel partagé. Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut disposer de sa propre copie de l'objet. Au départ, toutes les copies ont les mêmes attributs, mais en utilisant les commandes MQSC, vous pouvez modifier chaque copie de sorte que ses attributs diffèrent de ceux des autres copies. Les attributs des copies sont resynchronisés lorsque la définition principale du référentiel partagé est modifiée.

MQQSGD_SHARED
L'objet a une disposition partagée. Cela signifie qu'il existe dans le référentiel partagé une seule instance de l'objet connue de tous les gestionnaires de files d'attente du groupe de partage de files d'attente. Lorsqu'un gestionnaire de files d'attente du groupe accède à l'objet, il accède à l'instance partagée unique de l'objet.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_QSG_DISP avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sur z/OS.

QueueAccounting (MQLONG)

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X	X	

Cette option contrôle la collecte des données comptables pour la file d'attente. Pour que les données de comptabilité soient collectées pour cette file d'attente, les données de comptabilité de cette connexion doivent également être activées, à l'aide de l'attribut QMGR ACCTQ ou de la zone Options de la structure MQCNO sur l'appel MQCONNX.

Cet attribut comporte l'une des valeurs suivantes :

MQMON_Q_DIR
Les données comptables de cette file d'attente sont collectées en fonction de la valeur de l'attribut QMGR ACCTQ. Il s'agit du paramètre par défaut.

MQMON_OFF
Ne collectez pas de données de comptabilité pour cette file d'attente.

MQMON_ON
Collectez les données de comptabilité pour cette file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_ACCOUNTING_Q avec l'appel MQINQ.

QueueMonitoring (MQLONG)

Ctrl la collecte des données de surveillance en ligne pour les files d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

La valeur est l'une des suivantes :

MQMON_Q_DIR
Collectez les données de surveillance en fonction de la valeur de l'attribut de gestionnaire de files d'attente *QueueMonitoring* . Il s'agit de la valeur par défaut.

MQMON_OFF
La collecte des données de surveillance en ligne est désactivée pour cette file d'attente.

MQMON_FAIBLE
Si la valeur de l'attribut de gestionnaire de files d'attente *QueueMonitoring* n'est pas MQMON_NONE, la collecte de données de surveillance en ligne est activée, avec un faible débit de collecte de données pour cette file d'attente.

MQMON_MEDIUM
Si la valeur de l'attribut de gestionnaire de files d'attente *QueueMonitoring* n'est pas MQMON_NONE, la collecte de données de surveillance en ligne est activée, avec un taux modéré de collecte de données pour cette file d'attente.

MQMON_ELEVE
Si la valeur de l'attribut de gestionnaire de files d'attente *QueueMonitoring* n'est pas MQMON_NONE, la collecte de données de surveillance en ligne est activée, avec un taux élevé de collecte de données pour cette file d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_MONITORING_Q avec l'appel MQINQ.

QueueStatistics (MQCHAR12)

Locale	Modèle	Alias	Eloignée	Cluster
X	X	X	X	

Permet de contrôler la collecte des données statistiques pour la file d'attente.

Cet attribut comporte l'une des valeurs suivantes :

MQMON_Q_DIR
Les données comptables de cette file d'attente sont collectées en fonction de la valeur de l'attribut QMGR STATQ. Il s'agit du paramètre par défaut.

MQMON_OFF
Désactivez la collecte de données statistiques pour cette file d'attente.

MQMON_ON
Activez la collecte de données statistiques pour cette file d'attente.

Type de file d'attente (MQLONG)

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	X

Il s'agit du type de file d'attente ; il possède l'une des valeurs suivantes:

MQQT_ALIAS (alias MQ)
Définition de file d'attente alias.

MQQT_CLUSTER
File d'attente de cluster.

MQQT_LOCAL
File d'attente locale.

MQQT_REMOTE
Définition locale d'une file d'attente éloignée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_Q_TYPE avec l'appel MQINQ.

RemoteQMgrNom (MQCHAR48)

Locale	Modèle	Alias	Eloignée	Cluster
			X	

Il s'agit du nom du gestionnaire de files d'attente éloignées sur lequel la file d'attente *RemoteQName* est définie. Si la file d'attente *RemoteQName* possède la valeur *QSGDisp* MQQSGD_COPY ou MQQSGD_SHARED, *RemoteQMgrName* peut être le nom du groupe de partage de files d'attente qui possède *RemoteQName*.

Si une application ouvre la définition locale d'une file d'attente éloignée, *RemoteQMgrName* ne doit pas être vide et ne doit pas être le nom du gestionnaire de files d'attente local. Si *XmitQName* est vide, la file d'attente locale portant le même nom que *RemoteQMgrName* est utilisée comme file d'attente de transmission. S'il n'existe pas de file d'attente portant le nom *RemoteQMgrName*, la file d'attente identifiée par l'attribut de gestionnaire de files d'attente *DefXmitQName* est utilisée.

Si cette définition est utilisée pour un alias de gestionnaire de files d'attente, *RemoteQMgrName* est le nom du gestionnaire de files d'attente dont l'alias est utilisé. Il peut s'agir du nom du gestionnaire de files d'attente local. Sinon, si *XmitQName* est vide lors de l'ouverture, il doit y avoir une file d'attente locale portant le même nom que *RemoteQMgrName*; cette file d'attente est utilisée comme file d'attente de transmission.

Si cette définition est utilisée pour un alias de réponse, il s'agit du nom du gestionnaire de files d'attente qui doit être *ReplyToQMgr*.

Remarque : Aucune validation n'est effectuée sur la valeur spécifiée pour cet attribut lorsque la définition de file d'attente est créée ou modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_REMOTE_Q_MGR_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_MGR_NAME_LENGTH.

RemoteQName (MQCHAR48)

Locale	Modèle	Alias	Eloignée	Cluster
			X	

Il s'agit du nom de la file d'attente telle qu'elle est connue sur le gestionnaire de files d'attente éloignées *RemoteQMgrName*.

Si une application ouvre la définition locale d'une file d'attente éloignée, lorsque l'ouverture a lieu, *RemoteQName* ne doit pas être vide.

Si cette définition est utilisée pour une définition d'alias de gestionnaire de files d'attente, lorsque l'ouverture a lieu, la zone *RemoteQName* doit être vide.

Si la définition est utilisée pour un alias de réponse, ce nom correspond au nom de la file d'attente qui doit être *ReplyToQ*.

Remarque : Aucune validation n'est effectuée sur la valeur spécifiée pour cet attribut lorsque la définition de file d'attente est créée ou modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_REMOTE_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

RetentionInterval (MQLONG)

Il s'agit de la durée pendant laquelle la file d'attente doit être conservée. Une fois ce délai écoulé, la file d'attente peut être supprimée.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

La durée est mesurée en heures, à compter de la date et de l'heure de création de la file d'attente. La date et l'heure de création de la file d'attente sont enregistrées dans les attributs *CreationDate* et *CreationTime* .

Ces informations sont fournies pour permettre à une application de nettoyage ou à l'opérateur d'identifier et de supprimer les files d'attente qui ne sont plus nécessaires.

Remarque : Le gestionnaire de files d'attente n'effectue aucune action pour supprimer des files d'attente en fonction de cet attribut ou pour empêcher la suppression de files d'attente avec un intervalle de conservation qui n'est pas arrivé à expiration. Il incombe à l'utilisateur d'effectuer les actions requises.

Utilisez un intervalle de conservation réaliste pour empêcher l'accumulation de files d'attente dynamiques permanentes (voir attributDefinitionType). Toutefois, cet attribut peut également être utilisé avec des files d'attente prédéfinies.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_RETENTION_INTERVAL avec l'appel MQINQ.

Portée (MQLONG)

Contrôle si une entrée de cette file d'attente existe également dans un répertoire de cellules.

Locale	Modèle	Alias	Eloignée	Cluster
X		X	X	

Un répertoire de cellule est fourni par un service de nom installable. La valeur est l'une des suivantes :

MQSCO_Q_DIR
La définition de file d'attente a une portée de type gestionnaire de files d'attente: la définition de la file d'attente ne s'étend pas au-delà du gestionnaire de files d'attente qui la possède. Pour ouvrir la file d'attente à des fins de sortie à partir d'un autre gestionnaire de files d'attente, le nom du gestionnaire de files d'attente propriétaire doit être spécifié ou l'autre gestionnaire de files d'attente doit avoir une définition locale de la file d'attente.

MQSCO_CELL
La définition de file d'attente a une portée de type cellule: la définition de file d'attente est également placée dans un répertoire de cellule accessible à tous les gestionnaires de files d'attente de la cellule. La file d'attente peut être ouverte pour la sortie à partir de n'importe quel gestionnaire de files d'attente de la cellule en spécifiant le nom de la file d'attente. Il n'est pas nécessaire de spécifier le nom du gestionnaire de files d'attente propriétaire de la file d'attente. Toutefois, la définition de file d'attente n'est pas disponible pour les gestionnaires de files d'attente de la cellule qui ont également une définition locale d'une file d'attente portant ce nom, car la définition locale est prioritaire.

Un répertoire de cellule est fourni par un service de nom installable.

Les files d'attente modèles et dynamiques ne peuvent pas avoir de portée cellule.

Cette valeur n'est valide que si un service annuaire prenant en charge un répertoire de cellule a été configuré.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_SCOPE avec l'appel MQINQ.

La prise en charge de cet attribut est soumise aux restrictions suivantes:

- Sous IBM i, l'attribut est pris en charge, mais seul MQSCO_Q_MGR est valide.
- Sous z/OS, l'attribut n'est pas pris en charge.

Partageabilité (MQLONG)
Indique si la file d'attente peut être ouverte en entrée plusieurs fois simultanément.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

La valeur est l'une des suivantes :

MQQA_SHAREABLE
La file d'attente est partageable.

Plusieurs ouvertures avec l'option MQOO_INPUT_SHARED sont autorisées.

MQQA_NOT_SHAREABLE
La file d'attente n'est pas partageable.

Un appel MQOPEN avec l'option MQOO_INPUT_SHARED est traité comme MQOO_INPUT_EXCLUSIVE.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_SHAREABILITY avec l'appel MQINQ.

StorageClass (MQCHAR8)
Il s'agit d'un nom défini par l'utilisateur qui définit la mémoire physique utilisée pour contenir la file d'attente. En pratique, un message n'est écrit sur le disque que s'il doit être paginé hors de sa mémoire tampon.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_STORAGE_CLASS avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_STORAGE_CLASS_LENGTH.

Cet attribut est pris en charge uniquement sur z/OS.

TriggerControl (MQLONG)

Cette option contrôle si les messages de déclenchement sont écrits dans une file d'attente d'initialisation pour démarrer une application afin de traiter la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Il s'agit de l'un des éléments suivants:

MQTC_OFF
Aucun message de déclenchement ne doit être écrit pour cette file d'attente. La valeur de *TriggerType* n'est pas pertinente dans ce cas.

MQTC_ON
Les messages de déclenchement doivent être écrits pour cette file d'attente lorsque les événements de déclenchement appropriés se produisent.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TRIGGER_CONTROL avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

TriggerData (MQCHAR64)

Il s'agit des données de format libre que le gestionnaire de files d'attente insère dans le message de déclenchement lorsqu'un message arrivant dans cette file d'attente entraîne l'écriture d'un message de déclenchement dans la file d'attente d'initialisation.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Le contenu de ces données n'a aucune importance pour le gestionnaire de files d'attente. Il est significatif soit pour l'application du moniteur de déclenchement qui traite la file d'attente d'initialisation, soit pour l'application que le moniteur de déclenchement démarre.

La chaîne de caractères ne doit pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_TRIGGER_DATA avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET. La longueur de cet attribut est donnée par MQ_TRIGGER_DATA_LENGTH.

TriggerDepth (MQLONG)

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Il s'agit du nombre de messages de priorité *TriggerMsgPriority* ou supérieure qui doivent se trouver dans la file d'attente avant qu'un message de déclenchement ne soit écrit. Cela s'applique lorsque *TriggerType* est défini sur MQTT_DEPTH. La valeur de *TriggerDepth* est supérieure ou égale à 1. Cet attribut n'est pas utilisé autrement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TRIGGER_DEPTH avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

Priorité TriggerMsg(MQLONG)

Il s'agit de la priorité de message au-dessous de laquelle les messages ne contribuent pas à la génération de messages de déclenchement (c'est-à-dire que le gestionnaire de files d'attente ignore ces messages lorsqu'il décide de générer ou non un message de déclenchement).

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

TriggerMsgPriority peut être compris entre zéro (valeur la plus faible) et *MaxPriority* (valeur la plus élevée ; voir AttributMaxPriority) ; la valeur zéro entraîne la contribution de tous les messages à la génération de messages de déclenchement.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TRIGGER_MSG_PRIORITY avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

TriggerType (MQLONG)

Cette option contrôle les conditions dans lesquelles les messages de déclenchement sont écrits suite à l'arrivée de messages dans cette file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

Elle a l'une des valeurs suivantes :

MQTT_AUCUN
Aucun message de déclenchement n'est écrit suite à des messages dans cette file d'attente. Cela a le même effet que de définir *TriggerControl* sur MQTC_OFF.

MQTT_PREMIER
Un message de déclenchement est écrit chaque fois que le nombre de messages de priorité *TriggerMsgPriority* ou supérieure dans la file d'attente passe de 0 à 1.

MQTT_EVERY
Un message de déclenchement est écrit chaque fois qu'un message de priorité *TriggerMsgPriority* ou supérieure arrive dans la file d'attente.

MQTT_DEPTH
Un message de déclenchement est écrit chaque fois que le nombre de messages de priorité *TriggerMsgPriority* ou supérieure dans la file d'attente est égal ou supérieur à *TriggerDepth*. Une fois que le message de déclenchement a été écrit, *TriggerControl* est défini sur MQTC_OFF pour empêcher tout déclenchement supplémentaire jusqu'à ce qu'il soit explicitement activé à nouveau.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_TRIGGER_TYPE avec l'appel MQINQ. Pour modifier la valeur de cet attribut, utilisez l'appel MQSET.

Syntaxe (MQLONG)

Indique à quoi sert la file d'attente.

Locale	Modèle	Alias	Eloignée	Cluster
X	X			

La valeur est l'une des suivantes :

MQUS_NORMAL
Il s'agit d'une file d'attente utilisée par les applications lors de l'insertion et de l'extraction de messages ; la file d'attente n'est pas une file d'attente de transmission.

TRANSMIS_MQUS_TRANSMISSION
Il s'agit d'une file d'attente utilisée pour le stockage des messages destinés aux gestionnaires de files d'attente éloignées. Lorsqu'une application envoie un message à une file d'attente éloignée, le gestionnaire de files d'attente local stocke temporairement le message dans la file d'attente de transmission appropriée dans un format spécial. Un agent de canal de transmission lit ensuite le message à partir de la file d'attente de transmission et le transporte vers le gestionnaire de files

d'attente éloignées. Pour plus d'informations sur les files d'attente de transmission, voir Définition d'une file d'attente de transmission.

Seules les applications privilégiées peuvent ouvrir une file d'attente de transmission pour MQOO_OUTPUT afin d'y placer directement des messages. Généralement, seules les applications utilitaires le font. Vérifiez que le format des données de message est correct (voir «MQXQH-en-tête de file d'attente de transmission», à la page 601) ou que des erreurs peuvent se produire lors du processus de transmission. Le contexte n'est pas transmis ou défini sauf si l'une des options de contexte MQPMO_ * _CONTEXT est spécifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_USAGE avec l'appel MQINQ.

XmitQName (MQCHAR48)

Il s'agit du nom de la file d'attente de transmission. Si cet attribut n'est pas vide lorsqu'une ouverture se produit, que ce soit pour une file d'attente éloignée ou pour une définition d'alias de gestionnaire de files d'attente, il indique le nom de la file d'attente de transmission locale à utiliser pour le transfert du message.

Locale	Modèle	Alias	Eloignée	Cluster
			X	

Si *XmitQName* est vide, la file d'attente locale dont le nom est identique à *RemoteQMgrName* est utilisée comme file d'attente de transmission. S'il n'existe pas de file d'attente portant le nom *RemoteQMgrName*, la file d'attente identifiée par l'attribut de gestionnaire de files d'attente *DefXmitQName* est utilisée.

Cet attribut est ignoré si la définition est utilisée en tant qu'alias de gestionnaire de files d'attente et que *RemoteQMgrName* est le nom du gestionnaire de files d'attente local. Il est également ignoré lorsque la définition est utilisée comme définition d'alias de file d'attente de réponses.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_XMIT_Q_NAME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_Q_NAME_LENGTH.

Attributs des listes de noms

Le tableau suivant récapitule les attributs spécifiques aux listes de noms. Les attributs sont décrits par ordre alphabétique.

Les listes de noms sont prises en charge sur tous les systèmes WebSphere MQ , ainsi que sur les clients WebSphere MQ MQI connectés à ces systèmes.

Remarque : Les noms des attributs affichés dans cette section sont des noms descriptifs utilisés avec les appels MQINQ et MQSET ; les noms sont identiques à ceux des commandes PCF. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, des noms abrégés alternatifs sont utilisés ; voir Commandes de script (MQSC) pour plus d'informations.

Tableau 576. Attributs des listes de noms

Attribut	Description
AlterationDate	Date de la dernière modification de la définition
AlterationTime	Heure de la dernière modification de la définition
NameCount	Nombre de noms dans la liste de noms
NamelistDesc	Description de la liste de noms
NamelistName	Nom de la liste de noms
Noms	Liste de noms *NameCount*
NamelistType	Type de liste de noms
QSGDisp	Disposition de groupe de partage de files d'attente

AlterationDate (MQCHAR12)

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ALTERATION_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_DATE_LENGTH.

AlterationTime (MQCHAR8)

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ALTERATION_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_TIME_LENGTH.

NameCount (MQLONG)

Il s'agit du nombre de noms figurant dans la liste de noms. Elle est supérieure ou égale à zéro. La valeur suivante est définie:

MQNC_MAX_NAMELIST_NAME_COUNT
Nombre maximal de noms dans une liste de noms.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_NAME_COUNT avec l'appel MQINQ.

NamelistDesc (MQCHAR64)

Utilisez cette zone pour les commentaires descriptifs ; sa valeur est établie par le processus de définition. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, cette zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

Remarque : Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente *CodedCharSetId*), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_NAMELIST_DESC avec l'appel MQINQ.

La longueur de cet attribut est donnée par MQ_NAMELIST_DESC_LENGTH.

NamelistName (MQCHAR48)

Il s'agit du nom d'une liste de noms définie sur le gestionnaire de files d'attente local. Pour plus d'informations sur les noms de liste de noms, voir la section Autres noms d'objet .

Chaque liste de noms a un nom qui est différent des noms des autres listes de noms appartenant au gestionnaire de files d'attente, mais qui peut dupliquer les noms des autres objets de gestionnaire de files d'attente de types différents (par exemple, les files d'attente).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_NAMELIST_NAME avec l'appel MQINQ.

La longueur de cet attribut est donnée par MQ_NAMELIST_NAME_LENGTH.

NamelistType (MQLONG)

Indique la nature des noms dans la liste de noms et la manière dont la liste de noms est utilisée. Ses valeurs sont l'une des suivantes :

MQNT_AUCUN
Liste de noms sans type affecté.

MQNT_Q
Liste de noms contenant les noms des files d'attente.

MQNT_CLUSTER
Liste de noms contenant les noms des clusters.

INFO MQNT_AUTH_INFO
Liste de noms contenant les noms des objets d'informations d'authentification.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_NAMELIST_TYPE avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sur z/OS.

Noms (MQCHAR48xNameCount)

Il s'agit d'une liste de noms *NameCount* , où chaque nom correspond au nom d'un objet défini dans le gestionnaire de files d'attente local. Pour plus d'informations sur les noms d'objet, voir Règles de dénomination des objets IBM WebSphere MQ .

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_NAMES avec l'appel MQINQ.

La longueur de chaque nom dans la liste est indiquée par MQ_OBJECT_NAME_LENGTH.

QSGDisp (MQLONG)

Indique la disposition de la liste de noms. La valeur est l'une des suivantes :

MQQSGD_Q_DIR
L'objet possède une disposition de gestionnaire de files d'attente: la définition d'objet est connue uniquement du gestionnaire de files d'attente local ; la définition n'est pas connue des autres gestionnaires de files d'attente du groupe de partage de files d'attente.

Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut avoir un objet ayant le même nom et le même type que l'objet en cours, mais il s'agit d'objets distincts et il n'y a pas de corrélation entre eux. Leurs attributs ne sont pas contraints d'être identiques les uns aux autres.

MQQSGD_COPY
L'objet est une copie locale d'une définition d'objet maître qui existe dans le référentiel partagé. Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut disposer de sa propre copie de l'objet. Au départ, toutes les copies ont les mêmes attributs, mais vous pouvez modifier chaque copie à l'aide des commandes MQSC, de sorte que ses attributs diffèrent de ceux des autres copies. Les attributs des copies sont resynchronisés lorsque la définition principale du référentiel partagé est modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_QSG_DISP avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sur z/OS.

Attributs des définitions de processus

Le tableau suivant récapitule les attributs spécifiques aux définitions de processus. Les attributs sont décrits par ordre alphabétique.

Remarque : Les noms des attributs de cette section sont des noms descriptifs utilisés avec les appels MQINQ et MQSET ; les noms sont identiques à ceux des commandes PCF. Lorsque des commandes MQSC sont utilisées pour définir, modifier ou afficher des attributs, des noms abrégés alternatifs sont utilisés ; voir Commandes de script (MQSC) pour plus d'informations.

Tableau 577. Attributs des définitions de processus

Attribut	Description
AlterationDate	Date de la dernière modification de la définition
AlterationTime	Heure de la dernière modification de la définition
ApplId	Identificateur d'application
ApplType	Type d'application

Tableau 577. Attributs des définitions de processus (suite)	
Attribut	**Description**
EnvData	Données d'environnement
ProcessDesc	Description du processus
ProcessName	Nom du processus
QSGDisp	Disposition de groupe de partage de files d'attente
UserData	Données utilisateur

AlterationDate (MQCHAR12)

Il s'agit de la date de la dernière modification de la définition. Le format de la date est YYYY-MM-DD, complété par deux blancs de fin pour que la longueur soit de 12 octets.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ALTERATION_DATE avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_DATE_LENGTH.

AlterationTime (MQCHAR8)

Il s'agit de l'heure à laquelle la définition a été modifiée pour la dernière fois. Le format de l'heure est HH.MM.SS.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ALTERATION_TIME avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_TIME_LENGTH.

ApplId (MQCHAR256)

Il s'agit d'une chaîne de caractères qui identifie l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *ApplId* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par WebSphere MQ requiert que *ApplId* soit le nom d'un programme exécutable. Les remarques suivantes s'appliquent aux environnements indiqués:

- Sous z/OS, *ApplId* doit être:
  - Un identificateur de transaction CICS , pour les applications démarrées à l'aide de la transaction CKTI du moniteur de déclenchement CICS
  - Un identificateur de transaction IMS , pour les applications démarrées à l'aide du moniteur de déclenchement IMS CSQQTRMN
- Sur les systèmes Windows , le nom du programme peut être précédé d'un lecteur et d'un chemin de répertoire.
- Sur les systèmes UNIX , le nom du programme peut être précédé d'un chemin de répertoire.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_APPL_ID avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_PROCESS_APPL_ID_LENGTH.

ApplType (MQLONG)

Identifie la nature du programme à démarrer en réponse à la réception d'un message de déclenchement. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

ApplType peut avoir n'importe quelle valeur, mais les valeurs suivantes sont recommandées pour les types standard ; limitez les types d'application définis par l'utilisateur aux valeurs comprises entre MQAT_USER_FIRST et MQAT_USER_LAST:

MQAT_AIX
Application AIX (même valeur que MQAT_UNIX).

MQAT_BATCH
application de traitement par lots

MQAT_COURTIER
Application de courtier

MQAT_CICS
Transaction CICS .

MQAT_CICS_BRIDGE
Application de pont CICS .

MQAT_CICS_VSE
Transaction CICS/VSE .

MQAT_DOS
WebSphere MQ Application client MQI sur PC DOS.

MQAT_IMS
Application IMS .

MQAT_IMS_BRIDGE
Application de passerelle IMS .

MQAT_JAVA
Application Java.

MQAT_MVS
Application MVS ou TSO (même valeur que MQAT_ZOS).

MQAT_NOTES_AGENT
Lotus Notes Application Agent.

MQAT_NSK
Application HP Integrity NonStop Server .

MQAT_OS390
Application OS/390 (même valeur que MQAT_ZOS).

MQAT_OS400
Application IBM i .

MQAT_RRS_BATCH
Application par lots RRS.

MQAT_UNIX
Application UNIX .

MQAT_INCONNU
Application de type inconnu.

Utilisateur_MQAT
Application utilisateur.

MQAT_VOS
Application Stratus VOS.

MQAT_WINDOWS
Application Windows 16 bits.

MQAT_WINDOWS_NT
Application Windows 32 bits.

MQAT_WLM
Application de gestionnaire de charge de travail z/OS .

MQAT_XCF
XCF.

MQAT_ZOS
Application z/OS .

MQAT_USER_FIRST
Valeur la plus faible pour le type d'application défini par l'utilisateur.

MQAT_USER_LAST
Valeur la plus élevée pour le type d'application défini par l'utilisateur.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_APPL_TYPE avec l'appel MQINQ.

EnvData (MQCHAR128)

Il s'agit d'une chaîne de caractères qui contient des informations relatives à l'environnement concernant l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ; elles sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *EnvData* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par WebSphere MQ ajoute *EnvData* à la liste de paramètres transmise à l'application démarrée. La liste de paramètres comprend la structure MQTMC2 , suivie d'un blanc, suivi de *EnvData* avec les blancs de fin supprimés. Les remarques suivantes s'appliquent aux environnements indiqués:

- Sous z/OS:
  - *EnvData* n'est pas utilisé par les applications de moniteur de déclenchement fournies par WebSphere MQ.
  - Si ApplType est MQAT_WLM, vous pouvez fournir des valeurs par défaut dans EnvData pour les zones ServiceName et ServiceStep dans l'en-tête des informations de travail (MQWIH).
- Sur les systèmes UNIX , *EnvData* peut être défini sur le caractère & pour exécuter l'application démarrée en arrière-plan.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_ENV_DATA avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_PROCESS_ENV_DATA_LENGTH.

ProcessDesc (MQCHAR64)

Utilisez cette zone pour les commentaires descriptifs. Le contenu de la zone n'est pas significatif pour le gestionnaire de files d'attente, mais ce dernier peut exiger que la zone ne contienne que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, la zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

Remarque : Si cette zone contient des caractères qui ne figurent pas dans le jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente *CodedCharSetId*), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_PROCESS_DESC avec l'appel MQINQ.

La longueur de cet attribut est donnée par MQ_PROCESS_DESC_LENGTH.

ProcessName (MQCHAR48)

Il s'agit du nom d'une définition de processus définie sur le gestionnaire de files d'attente local.

Chaque définition de processus a un nom différent de celui des autres définitions de processus appartenant au gestionnaire de files d'attente. Toutefois, le nom de la définition de processus peut être identique à celui d'autres objets de gestionnaire de files d'attente de types différents (par exemple, des files d'attente).

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_PROCESS_NAME avec l'appel MQINQ.

La longueur de cet attribut est donnée par MQ_PROCESS_NAME_LENGTH.

QSGDisp (MQLONG)

Indique la disposition de la définition de processus. La valeur est l'une des suivantes :

MQQSGD_Q_DIR
L'objet possède une disposition de gestionnaire de files d'attente: la définition d'objet est connue uniquement du gestionnaire de files d'attente local ; la définition n'est pas connue des autres gestionnaires de files d'attente du groupe de partage de files d'attente.

Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut avoir un objet ayant le même nom et le même type que l'objet en cours, mais il s'agit d'objets distincts et il n'y a pas de corrélation entre eux. Leurs attributs ne sont pas contraints d'être identiques les uns aux autres.

MQQSGD_COPY
L'objet est une copie locale d'une définition d'objet maître qui existe dans le référentiel partagé. Chaque gestionnaire de files d'attente du groupe de partage de files d'attente peut disposer de sa propre copie de l'objet. Au départ, toutes les copies ont les mêmes attributs, mais vous pouvez modifier chaque copie à l'aide des commandes MQSC, de sorte que ses attributs diffèrent de ceux des autres copies. Les attributs des copies sont resynchronisés lorsque la définition principale du référentiel partagé est modifiée.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQIA_QSG_DISP avec l'appel MQINQ.

Cet attribut est pris en charge uniquement sur z/OS.

UserData (MQCHAR128)

`UserData` est une chaîne de caractères qui contient des informations utilisateur relatives à l'application à démarrer. Ces informations sont destinées à être utilisées par une application de moniteur de déclenchement qui traite les messages de la file d'attente d'initialisation ou l'application qui est démarrée par le moniteur de déclenchement. Les informations sont envoyées à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La signification de *`UserData`* est déterminée par l'application trigger-monitor. Le moniteur de déclenchement fourni par WebSphere MQ transmet *`UserData`* à l'application démarrée dans le cadre de la liste de paramètres. La liste de paramètres comprend la structure MQTMC2 (contenant *`UserData`*), suivie d'un blanc, suivie de *`EnvData`* avec les blancs de fin supprimés.

La chaîne de caractères ne peut pas contenir de valeurs nulles. Il est rempli à droite avec des blancs si nécessaire. Pour Microsoft Windows, la chaîne de caractères ne doit pas contenir de guillemets si la définition de processus doit être transmise à **`runmqtrm`**.

Pour déterminer la valeur de cet attribut, utilisez le sélecteur MQCA_USER_DATA avec l'appel MQINQ. La longueur de cet attribut est donnée par MQ_PROCESS_USER_DATA_LENGTH.

Codes retour

Pour chaque appel WebSphere MQ Message Queue Interface (MQI) et WebSphere MQ Administration Interface (MQAI), un code d' **achèvement** et un code **anomalie** sont renvoyés par le gestionnaire de files d'attente ou par une routine d'exit pour indiquer la réussite ou l'échec de l'appel.

Les applications ne doivent pas dépendre d'erreurs vérifiées dans un ordre spécifique, sauf lorsqu'elles sont spécifiquement mentionnées. Si plusieurs codes achèvement ou codes raison peuvent provenir d'un appel, l'erreur particulière signalée dépend de l'implémentation.

Les applications qui vérifient la réussite après un appel d'API WebSphere MQ doivent toujours vérifier le code achèvement. Ne supposez pas la valeur du code achèvement, en fonction de la valeur du code raison.

Codes achèvement

Le paramètre de code achèvement (*CompCode*) permet à l'appelant de voir rapidement si l'appel a abouti, s'est terminé partiellement ou a échoué.Voici une liste de codes achèvement, avec plus de détails que dans les descriptions d'appel:

MQCC_OK
L'appel a abouti ; tous les paramètres de sortie ont été définis. Le paramètre *Reason* a toujours la valeur MQRC_NONE dans ce cas.

MQCC_WARNING
L'appel s'est terminé partiellement. Certains paramètres de sortie peuvent avoir été définis en plus des paramètres de sortie *CompCode* et *Reason* . Le paramètre *Reason* fournit des informations supplémentaires sur l'exécution partielle.

MQCC_FAILED
Le traitement de l'appel n'a pas abouti. L'état du gestionnaire de files d'attente reste inchangé, sauf mention contraire. Les paramètres de sortie *CompCode* et *Reason* ont été définis ; les autres paramètres sont inchangés, sauf indication contraire.

Il peut s'agir d'une erreur dans le programme d'application ou d'une situation externe au programme. Par exemple, les droits de l'utilisateur peuvent avoir été révoqués. Le paramètre *Reason* fournit des informations supplémentaires sur l'erreur.

Codes raison

Le paramètre de code anomalie (*Reason*) qualifie le paramètre de code achèvement (*CompCode*).

S'il n'y a pas de raison particulière à signaler, MQRC_NONE est renvoyé. Un appel réussi renvoie MQCC_OK et MQRC_NONE.

Si le code achèvement est MQCC_WARNING ou MQCC_FAILED, le gestionnaire de files d'attente indique toujours une raison de qualification ; des détails sont fournis sous chaque description d'appel.

Lorsque des routines d'exit utilisateur définissent des codes achèvement et des raisons, elles doivent respecter ces règles. En outre, les valeurs de raison spéciale définies par les exits utilisateur doivent être inférieures à zéro, afin de s'assurer qu'elles n'entrent pas en conflit avec les valeurs définies par le gestionnaire de files d'attente. Les exits peuvent définir des raisons déjà définies par le gestionnaire de files d'attente, le cas échéant.

Les codes anomalie apparaissent également dans:

- Zone *Reason* de la structure MQDLH
- Zone *Feedback* de la structure MQMD

Pour obtenir une description complète des codes anomalie, voir Codes anomalie .

Règles de validation des options MQI

Cette section répertorie les situations qui produisent un code anomalie MQRC_OPTIONS_ERROR à partir d'un appel MQOPEN, MQPUT, MQPUT1, MQGET, MQCLOSE ou MQSUB.

Appel MQOPEN

Pour les options de l'appel MQOPEN:

- Au moins *un* des éléments suivants doit être spécifié:
  - MQOO_BROWSE
  - MQOO_INPUT_EXCLUSIVE[1]

  - MQOO_INPUT_SHARED[1]
  - MQOO_INPUT_AS_Q_DEF[1]
  - MQOO_INTERROGATION
  - MQOO_SORTIE
  - MQOO_SET
  - MQOO_BIND_ON_OPEN[2]
  - MQOO_BIND_NOT_FIXED[2]
  - MQOO_BIND_ON_GROUP[2]
  - MQOO_BIND_AS_Q_DEF[2]
- Seul *un* des éléments suivants est autorisé:
  - MQOO_READ_AHEAD
  - MQOO_NO_READ_AHEAD
  - MQOO_READ_AHEAD_AS_Q_DEF

1. Seul *un* des éléments suivants est autorisé:
   - MQOO_INPUT_EXCLUSIVE
   - MQOO_INPUT_SHARED
   - MQOO_INPUT_AS_Q_DEF
2. Seul *un* des éléments suivants est autorisé:
   - MQOO_BIND_ON_OPEN
   - MQOO_BIND_NOT_FIXED
   - MQOO_BIND_ON_GROUPE
   - MQOO_BIND_AS_Q_DEF

   Remarque : Les options répertoriées ci-dessus s'excluent mutuellement. Cependant, comme la valeur de MQOO_BIND_AS_Q_DEF est égale à zéro, le fait de la spécifier avec l'une des deux autres options de liaison n'entraîne pas le code anomalie MQRC_OPTIONS_ERROR. MQOO_BIND_AS_Q_DEF est fourni pour la documentation du programme d'aide.

- Si MQOO_SAVE_ALL_CONTEXT est spécifié, l'une des options MQOO_INPUT_ * doit également être spécifiée.
- Si l'une des options MQOO_SET_ * _CONTEXT ou MQOO_PASS_ * _CONTEXT est spécifiée, MQOO_OUTPUT doit également être spécifié.
- Si MQOO_CO_OP est spécifié, MQOO_BROWSE doit également être spécifié
- Si MQOO_NO_MULTICAST est spécifié, MQOO_OUTPUT doit également être spécifié.

Appel MQPUT

Pour les options d'insertion de message:

- La combinaison de MQPMO_SYNCPOINT et MQPMO_NO_SYNCPOINT n'est pas autorisée.
- Seul *un* des éléments suivants est autorisé:
  - MQPMO_CONTEXTE_PAR_DÉFAUT
  - MQPMO_NO_CONTEXT
  - MQPMO_PASS_ALL_CONTEXT
  - MQPMO_PASS_IDENTITY_CONTEXT
  - MQPMO_SET_ALL_CONTEXT
  - MQPMO_SET_IDENTITY_CONTEXT

- Seul *un* des éléments suivants est autorisé:
  - MQPMO_ASYNC_RESPONSE
  - MQPMO_SYNC_REPONSE
  - MQPMO_REPONSE_AS_TOPIC_DEF
  - MQPMO_REPONSE_AS_Q_DEF
- MQPMO_ALTERNATE_USER_AUTHORITY n'est pas autorisé (il est valide uniquement sur l'appel MQPUT1).

Appel MQPUT1

Pour les options put-message, les règles sont les mêmes que pour l'appel MQPUT, à l'exception de ce qui suit:

- MQPMO_ALTERNATE_USER_AUTHORITY est autorisé.
- MQPMO_LOGICAL_ORDER n'est *pas* autorisé.

Appel MQGET

Pour les options get-message:

- Seul *un* des éléments suivants est autorisé:
  - MQGMO_NO_SYNCPOINT
  - MQGMO_SYNCPOINT
  - MQGMO_SYNCPOINT_IF_PERSISTENT
- Seul *un* des éléments suivants est autorisé:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT n'est pas admise avec l'une des options suivantes:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQGMO_LOCK
  - MQGMO_UNLOCK
- MQGMO_SYNCPOINT_IF_PERSISTENT n'est pas autorisé avec l'un des éléments suivants:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQGMO_COMPLETE_MSG
  - MQGMO_UNLOCK
- MQGMO_MARK_SKIP_BACKOUT requiert la spécification de MQGMO_SYNCPOINT.
- La combinaison de MQGMO_WAIT et MQGMO_SET_SIGNAL n'est pas autorisée.
- Si MQGMO_LOCK est spécifié, l'un des éléments suivants doit également être spécifié:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT

- Si MQGMO_UNLOCK est spécifié, seuls les éléments suivants sont autorisés:
  - MQGMO_NO_SYNCPOINT
  - MQGMO_NO_WAIT

Appel MQCLOSE

Pour les options de l'appel MQCLOSE:

- La combinaison de MQCO_DELETE et MQCO_DELETE_PURGE n'est pas autorisée.
- Une seule des options suivantes est autorisée:
  - MQCO_KEEP_SUB
  - SOUS-TITRE_REMOVE_MQQUE

Appel MQSUB

Pour les options de l'appel MQSUB:

- Au moins un des éléments suivants doit être spécifié:
  - MQSO_ALTER
  - MQSO_RESUME
  - MQSO_CREER
- Une seule des options suivantes est autorisée:
  - MQSO_DURABLE
  - MQSO_NON_DURABLE

  Remarque : Les options répertoriées ci-dessus s'excluent mutuellement. Toutefois, comme la valeur de MQSO_NON_DURABLE est égale à zéro, sa spécification avec MQSO_DURABLE ne génère pas de code anomalie MQRC_OPTIONS_ERROR. MQSO_NON_DURABLE est fourni pour aider la documentation du programme.
- La combinaison de MQSO_GROUP_SUB et de MQSO_MANAGED n'est pas autorisée.
- MQSO_GROUP_SUB requiert la spécification de MQSO_SET_CORREL_ID.
- Une seule des options suivantes est autorisée:
  - ID utilisateur MQSO_ANY_USERID
  - ID_UTILISATEUR_FIXE_MQSO_FIXE
- MQSO_NEW_PUBLICATIONS_ONLY est uniquement autorisé en combinaison avec MQSO_CREATE.
- La combinaison de MQSO_PUBLICATIONS_ON_REQUEST et de SubLevel supérieure à 1 n'est pas autorisée.
- Une seule des options suivantes est autorisée:
  - MQSO_WILDCARD_CHAR
  - MQSO_WILDCARD_TOPIC
- MQSO_NO_MULTICAST requiert la spécification de MQSO_MANAGED.

Messages de commande de publication / abonnement en file d'attente

Une application peut utiliser des messages de commande MQRFH2 pour contrôler une application de publication / abonnement en file d'attente.

Une application qui utilise MQRFH2 pour la publication / l'abonnement peut envoyer les messages de commande suivants à SYSTEM.BROKER.CONTROL.QUEUE:

- «Supprimer le message de publication», à la page 868
- «Désenregistrer le message d'abonné», à la page 869
- «Publier le message», à la page 873
- «Enregistrement du message d'abonnement», à la page 876
- «Message de demande de mise à jour», à la page 881

Si vous écrivez des applications de publication / abonnement en file d'attente, vous devez comprendre ces messages, le message de réponse du gestionnaire de files d'attente et le descripteur de message (MQMD) ; voir les informations suivantes:

- «Message de réponse du gestionnaire de files d'attente», à la page 883
- «Paramètres MQMD pour les publications transmises par un gestionnaire de files d'attente», à la page 889
- «Paramètres MQMD dans les messages de réponse du gestionnaire de files d'attente», à la page 890
- «Codes raison de publication / abonnement», à la page 884

Les commandes sont contenues dans un dossier <psc> dans la zone **NameValueData** de l'en-tête MQRFH2 . Le message qui peut être envoyé par un courtier en réponse à un message de commande est contenu dans un dossier <pscr> .

Les descriptions de chaque commande répertorient les propriétés qui peuvent être contenues dans un dossier. Sauf indication contraire, les propriétés sont facultatives et ne peuvent apparaître qu'une seule fois.

Les noms des propriétés sont affichés sous la forme <Command>.
Les valeurs doivent être au format chaîne, par exemple: Publish.
Une constante de type chaîne représentant la valeur d'une propriété est affichée entre parenthèses, par exemple: (MQPSC_PUBLISH).
Les constantes de chaîne sont définies dans le fichier d'en-tête cmqpsc.h qui est fourni avec le gestionnaire de files d'attente.

Supprimer le message de publication

Le message de commande **Delete Publication** est envoyé à un gestionnaire de files d'attente à partir d'un diffuseur de publications ou d'un autre gestionnaire de files d'attente pour lui demander de supprimer les publications conservées pour les rubriques spécifiées.

Ce message est envoyé à une file d'attente surveillée par l'interface de publication / abonnement en file d'attente du gestionnaire de files d'attente.

La file d'entrée doit être la file d'attente à laquelle la publication d'origine a été envoyée.

Si vous disposez des droits pour certaines, mais pas pour toutes, des rubriques qui sont spécifiées dans le message de commande **Delete Publication** , seules ces rubriques sont supprimées. Un message **Broker Response** indique les rubriques qui ne sont pas supprimées.

De même, si une commande **Publish** contient plusieurs rubriques, une commande **Delete Publication** correspondant à certaines de ces rubriques, mais pas à toutes, supprime uniquement les publications des rubriques spécifiées dans la commande **Delete Publication** .

Voir «Paramètres MQMD pour les publications transmises par un gestionnaire de files d'attente», à la page 889 pour plus de détails sur les paramètres de descripteur de message (MQMD) nécessaires lors de l'envoi d'un message de commande au gestionnaire de files d'attente.

Propriétés

< Command> (*MQPSC_COMMAND*)
La valeur est DeletePub(*MQPSC_DELETE_PUBLICATION*).

Cette propriété doit être spécifiée.

< Topic> (*MQPSC_TOPIC*)
La valeur est une chaîne qui contient une rubrique pour laquelle les publications conservées doivent être supprimées. Des caractères génériques peuvent être inclus dans la chaîne pour supprimer des publications dans plusieurs rubriques.

Cette propriété doit être spécifiée ; elle peut être répétée pour autant de rubriques que nécessaire.

<DelOpt> (*MQPSC_DELETE_OPTION*)
La propriété des options de suppression peut prendre l'une des valeurs suivantes:

Local (*MQPSC_LOCAL*)

Toutes les publications conservées pour les rubriques spécifiées sont supprimées du gestionnaire de files d'attente local (c'est-à-dire du gestionnaire de files d'attente auquel ce message est envoyé), qu'elles aient été publiées avec l'option `Local` ou non.

Les publications des autres gestionnaires de files d'attente ne sont pas affectées.

Aucun (*MQPSC_NONE*)

Toutes les options prennent leurs valeurs par défaut. Cela a le même effet que l'omission de la propriété `DelOpt` . Si d'autres options sont spécifiées en même temps, `Aucun` est ignoré.

Par défaut, si cette propriété est omise, toutes les publications conservées pour les rubriques spécifiées sont supprimées dans tous les gestionnaires de files d'attente du réseau, qu'elles aient été publiées ou non avec l'option `Local` .

Exemple

Voici un exemple de `NameValueData` pour un message de commande **`Delete Publication`** . Il est utilisé par l'exemple d'application pour supprimer, au niveau du gestionnaire de files d'attente local, la publication conservée qui contient le score le plus récent dans la correspondance entre Team1 et Team2.

```
<psc>
 <Command>DeletePub</Command>
 <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
 <DelOpt>Local</DelOpt>
</psc>
```

Désenregistrer le message d'abonné

Le message de commande **`Deregister Subscriber`** est envoyé à un gestionnaire de files d'attente par un abonné, ou par une autre application pour le compte d'un abonné, pour indiquer qu'il ne souhaite plus recevoir de messages correspondant aux paramètres indiqués.

Ce message est envoyé à SYSTEM.BROKER.CONTROL.QUEUE, file d'attente de contrôle du gestionnaire de files d'attente. L'utilisateur doit disposer des droits nécessaires pour placer un message dans cette file d'attente.

Pour plus de détails sur les paramètres de descripteur de message (MQMD) requis lors de l'envoi d'un message de commande au gestionnaire de files d'attente, voir Paramètres MQMD pour les publications transmises par un gestionnaire de files d'attente .

Un abonnement individuel peut être désenregistré en spécifiant les valeurs de rubrique, de point d'abonnement et de filtre correspondantes de l'abonnement d'origine. Si l'une des valeurs n'a pas été spécifiée (c'est-à-dire qu'elle a pris les valeurs par défaut) dans l'abonnement d'origine, elle doit être omise lorsque l'abonnement est désenregistré.

Tous les abonnements d'un abonné ou d'un groupe d'abonnés peuvent être désenregistrés à l'aide de l'option `DeregAll` . Par exemple, si `DeregAll` est spécifié, avec un point d'abonnement (mais pas de rubrique ni de filtre), tous les abonnements de l'abonné sur le point d'abonnement spécifié sont désenregistrés, indépendamment de la rubrique et du filtre. Toute combinaison de rubrique, de filtre et de point d'abonnement est autorisée ; si les trois sont spécifiés, un seul abonnement peut correspondre et l'option `DeregAll` est ignorée.

Le message doit être envoyé par l'abonné qui a enregistré l'abonnement ; ceci est confirmé en vérifiant l'ID utilisateur de l'abonné.

Les abonnements peuvent également être désenregistrés par un administrateur système à l'aide de commandes MQSC ou PCF. Toutefois, les abonnements enregistrés avec une file d'attente dynamique temporaire sont associés à la file d'attente, et pas seulement au nom de la file d'attente. Si la file d'attente est supprimée, explicitement ou par déconnexion de l'application du gestionnaire de files d'attente, il n'est plus possible d'utiliser la commande **Deregister Subscriber** pour désenregistrer les abonnements de cette file d'attente. Les abonnements peuvent être désenregistrés à l'aide du plan de travail du développeur et ils sont supprimés automatiquement par le gestionnaire de files d'attente lors de la prochaine mise en correspondance d'une publication avec l'abonnement ou lors du prochain redémarrage du gestionnaire de files d'attente. Dans des circonstances normales, les applications doivent désenregistrer leurs abonnements avant de supprimer la file d'attente ou de se déconnecter du gestionnaire de files d'attente.

Si un abonné envoie un message pour désenregistrer un abonnement et qu'il reçoit un message de réponse indiquant que le traitement a abouti, certaines publications peuvent encore atteindre la file d'attente de l'abonné si elles étaient traitées par le gestionnaire de files d'attente en même temps que l'abonnement a été désenregistré. Si les messages ne sont pas supprimés de la file d'attente, il peut y avoir une accumulation de messages non traités dans la file d'attente de l'abonné. Si l'application exécute une boucle qui inclut un appel MQGET avec l'ID `CorrelId` approprié après une mise en veille pendant un certain temps, ces messages sont effacés de la file d'attente.

De même, si l'abonné utilise une file d'attente dynamique permanente, désenregistre et ferme la file d'attente avec l'option *MQCO_DELETE_PURGE* sur un appel MQCLOSE, la file d'attente peut ne pas être vide. Si des publications du gestionnaire de files d'attente ne sont pas encore validées lors de la suppression de la file d'attente, un code retour MQRC_Q_NOT_EMPTY est émis par l'appel MQCLOSE. L'application peut éviter ce problème en mettant en veille et en émettant à nouveau l'appel MQCLOSE de temps à autre.

Propriétés

< Command> (*MQPSC_COMMAND*)
La valeur est `DeregSub` (*MQPSC_REGISTER_SUBSCRIBER*).

Cette propriété doit être spécifiée.

< Topic> (*MQPSC_TOPIC*)
La valeur est une chaîne qui contient la rubrique à désenregistrer.

Cette propriété peut, si vous le souhaitez, être répétée si plusieurs rubriques doivent être désenregistrées. Elle peut être omise si `DeregAll` est spécifié dans <RegOpt>.

Les rubriques spécifiées peuvent être un sous-ensemble de celles qui sont enregistrées si l'abonné souhaite conserver des abonnements pour d'autres rubriques. Les caractères génériques sont autorisés, mais une chaîne de rubrique qui contient des caractères génériques doit correspondre exactement à la chaîne correspondante qui a été spécifiée dans le message de commande **Deregister Subscriber** .

<SubPoint> (*MQPSC_SUBSCRIPTION_POINT*)
La valeur est une chaîne qui spécifie le point d'abonnement à partir duquel l'abonnement doit être déconnecté.

Cette propriété ne doit pas être répétée. Elle peut être omise si < Topic> est spécifié ou si `DeregAll` est spécifié dans <RegOpt>. Si vous omettez cette propriété, les événements suivants se produisent:

- Si vous ne spécifiez **pas** `DeregAll`, les abonnements correspondant à la propriété < Topic> (et à la propriété < Filter > , le cas échéant) sont désenregistrés du point d'abonnement par défaut.
- Si vous spécifiez `DeregAll`, tous les abonnements (correspondant aux propriétés < Topic> et < Filter > , le cas échéant) sont désenregistrés de tous les points d'abonnement.

Notez que vous ne pouvez pas spécifier explicitement le point d'abonnement par défaut. Par conséquent, il n'est pas possible de désenregistrer tous les abonnements à partir de ce point d'abonnement uniquement ; vous devez spécifier les rubriques.

<SubIdentity> (*MQPSC_SUBSCRIPTION_IDENTITY*)
Il s'agit d'une chaîne de longueur variable d'une longueur maximale de 64 caractères. Il est utilisé pour représenter une application ayant un intérêt dans un abonnement. Le gestionnaire de files d'attente gère un ensemble d'identités d'abonné pour chaque abonnement. Chaque abonnement peut permettre à son ensemble d'identités de ne contenir qu'une seule identité ou un nombre illimité d'identités.

Si `SubIdentity` se trouve dans l'ensemble d'identités de l'abonnement, il est supprimé de l'ensemble. Si l'ensemble d'identités devient vide à la suite de cette opération, l'abonnement est supprimé du gestionnaire de files d'attente, sauf si `LeaveOnly` est spécifié comme valeur de la propriété `RegOpt` . Si l'ensemble d'identités contient toujours d'autres identités, l'abonnement n'est pas supprimé du gestionnaire de files d'attente et le flux de publication n'est pas interrompu.

Si `SubIdentity` est spécifié, mais que `SubIdentity` ne figure pas dans l'identité définie pour l'abonnement, la commande **`Deregister Subscriber`** échoue avec le code retour *MQRCCF_SUB_IDENTITY_ERROR*.

< Filtre > (*MQPSC_FILTER*)
La valeur est une chaîne spécifiant le filtre à désenregistrer. Il doit correspondre exactement, y compris la casse et les espaces, à un filtre d'abonnement précédemment enregistré.

Cette propriété peut éventuellement être répétée si plusieurs filtres doivent être désenregistrés. Elle peut être omise si `< Topic>` est spécifié ou si `DeregAll` est spécifié dans `<RegOpt>`.

Les filtres spécifiés peuvent être un sous-ensemble de ceux enregistrés si l'abonné souhaite conserver des abonnements pour d'autres filtres.

<RegOpt> (*MQPSC_REGISTRATION_OPTION*)
La propriété des options d'enregistrement peut prendre les valeurs suivantes:

`DeregAll`
(*MQPSC_DEREGISTER_ALL*)

Tous les abonnements correspondants enregistrés pour cet abonné doivent être désenregistrés.

Si vous spécifiez `DeregAll`:

- `< Topic>`, `<SubPoint>`et `< Filtre >` peuvent être omis.
- `< Topic>` et `< Filtre >` peuvent être répétés, si nécessaire.
- `<SubPoint>` ne doit pas être répété.

Si vous **ne** spécifiez pas `DeregAll`:

- `< Topic>` doit être spécifié et peut être répété si nécessaire.
- `<SubPoint>` et `< Filtre >` peuvent être omis.
- `<SubPoint>` ne doit pas être répété.
- `< Filtre >` peut être répété, si nécessaire.

Si les rubriques et les filtres sont répétés, tous les abonnements correspondant à toutes les combinaisons des deux sont supprimés. Par exemple, une commande **`Deregister Subscriber`** qui spécifie trois rubriques et trois filtres tente de supprimer neuf abonnements.

`IDCorrelAs`
(*MQPSC_CORREL_ID_AS_IDENTITY*)

L'ID `CorrelId` du descripteur de message (MQMD), qui ne doit pas être égal à zéro, est utilisé pour identifier l'abonné. Il doit correspondre à l'ID `CorrelId` utilisé dans l'abonnement d'origine.

`FullResp`
(*MQPSC_FULL_RESPONSE*)

LorsqueFullResp est spécifié, tous les attributs de l'abonnement sont renvoyés dans le message de réponse, si la commande n'échoue pas.

Lorsque FullResp est spécifié, DeregAll n'est pas autorisé dans la commande **Deregister Subscriber** . Il n'est pas non plus possible de spécifier plusieurs rubriques. La commande échoue avec le code retour *MQRCCF_REG_OPTIONS_ERROR*, dans les deux cas.

LeaveOnly
(*MQPSC_LEAVE_ONLY*)

Lorsque vous le spécifiez avec une SubIdentity qui se trouve dans l'ensemble d'identités de l'abonnement, SubIdentity est supprimé de l'ensemble d'identités de l'abonnement. L'abonnement n'est pas supprimé du gestionnaire de files d'attente, même si l'ensemble d'identités résultant est vide. Si la valeur SubIdentity ne figure pas dans l'ensemble d'identités, la commande échoue avec le code retour *MQRCCF_SUB_IDENTITY_ERROR*.

Si LeaveOnly est spécifié sans SubIdentity, la commande échoue avec le code retour *MQRCCF_REG_OPTIONS_ERROR*.

Si ni LeaveOnly ni SubIdentity ne sont spécifiés, l'abonnement est supprimé quel que soit le contenu de l'identité définie pour l'abonnement.

NONE
(MQPSC_NONE)

Toutes les options prennent leurs valeurs par défaut. Cela a le même effet que l'omission de la propriété des options d'enregistrement. Si d'autres options sont spécifiées en même temps, Aucun est ignoré.

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

Lorsqu'elle est spécifiée, l'identité de l'abonné (file d'attente, gestionnaire de files d'attente et ID corrélation) n'est pas limitée à un seul ID utilisateur. Ce comportement diffère du comportement existant du gestionnaire de files d'attente qui associe l'ID utilisateur du message d'enregistrement d'origine à l'identité de l'abonné et empêche ensuite tout autre utilisateur d'utiliser cette identité. Si un nouvel abonné tente d'utiliser la même identité, le code retour *MQRCCF_DUPLICATE_SUBSCRIPTION* est renvoyé.

Tout utilisateur peut modifier ou désenregistrer l'abonnement lorsqu'il dispose des droits appropriés, en évitant la vérification existante que l'ID utilisateur doit correspondre à celui de l'abonné d'origine.

Pour ajouter cette option à un abonnement existant, la commande doit provenir du même ID utilisateur que l'abonnement d'origine lui-même.

Si VariableUserID est défini pour l'abonnement à désenregistrer, il doit être défini à l'heure de désenregistrement pour indiquer l'abonnement en cours de désenregistrement. Sinon, l'ID utilisateur de la commande **Deregister Subscriber** est utilisé pour identifier l'abonnement. Elle est remplacée, avec les autres identificateurs d'abonné, si un nom d'abonnement est fourni.

Par défaut, si cette propriété est omise, aucune option d'enregistrement n'est définie.

<QMgrName> (*MQPSC_Q_MGR_NAME*)
La valeur est le nom du gestionnaire de files d'attente pour la file d'attente de souscription. Il doit correspondre au QMgrName utilisé dans l'abonnement d'origine.

Si cette propriété est omise, la valeur par défaut est le nom ReplyToQMgr dans le descripteur de message (MQMD). Si le nom obtenu est vide, le nom du gestionnaire de files d'attente est utilisé par défaut.

< nom_Q> (*MQPSC_Q_NAME*)
La valeur est le nom de la file d'attente de souscription. Il doit correspondre au QName utilisé dans l'abonnement d'origine.

Si cette propriété est omise, la valeur par défaut est le nom ReplyToQ dans le descripteur de message (MQMD), qui ne doit pas être vide.

<SubName> (*MQPSC_SUBSCRIPTION_NAME*)
Si vous spécifiez SubName dans une commande **Deregister Subscriber** , la valeur SubName est prioritaire sur toutes les autres zones d'identificateur à l'exception de l'ID utilisateur, sauf si VariableUserId est défini sur l'abonnement lui-même. Si VariableUserId n'est pas défini, la commande **Deregister Subscriber** aboutit uniquement si l'ID utilisateur du message de commande correspond à celui de l'abonnement, sinon la commande échoue avec le code retour *MQRCCF_DUPLICATE_IDENTITY*.

S'il existe un abonnement qui correspond à l'identité traditionnelle de cette commande mais qui ne comporte pas de SubName , la commande **Deregister Subscriber** échoue avec le code retour *MQRCCF_SUB_NAME_ERROR*. Si une tentative est effectuée pour désenregistrer un abonnement comportant un SubName à l'aide d'un message de commande qui correspond à l'identité traditionnelle mais sans SubName spécifié, la commande aboutit.

<SubUserde sous-utilisateur > (*MQPSC_SUBSCRIPTION_USER_DATA*)
Il s'agit d'une chaîne de texte de longueur variable. La valeur est stockée par le gestionnaire de files d'attente avec l'abonnement mais n'a aucune influence sur la distribution de la publication à l'abonné. La valeur peut être modifiée en s'enregistrant à nouveau dans le même abonnement avec une nouvelle valeur. Cet attribut est destiné à l'utilisation de l'application.

SubUserLes données sont renvoyées dans les informations de métarubrique (MQCACF_REG_SUB_USER_DATA) pour un abonnement, si des données SubUsersont présentes.

Exemple

Voici un exemple de NameValueData pour un message de commande **Deregister Subscriber** . Dans cet exemple, l'exemple d'application désenregistre son abonnement aux rubriques qui contiennent le score le plus récent pour toutes les correspondances. L'identité de l'abonné, y compris CorrelId, est extraite des valeurs par défaut du MQMD.

```
<psc>
 <Command>DeregSub</Command>
 <RegOpt>CorrelAsId</RegOpt>
 <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

Publier le message

Le message de commande **Publish** est inséré dans une file d'attente, ou depuis un gestionnaire de files d'attente vers un abonné, pour publier des informations sur une ou plusieurs rubriques spécifiées.

Le droit de placer un message dans une file d'attente et le droit de publier des informations sur une ou plusieurs rubriques spécifiées sont nécessaires.

Si l'utilisateur est autorisé à publier des informations sur certaines, mais pas sur toutes, rubriques, seules ces rubriques sont utilisées pour la publication ; une réponse d'avertissement indique les rubriques qui ne sont pas utilisées pour la publication.

Si un abonné possède des abonnements correspondants, le gestionnaire de files d'attente transmet le message **Publish** aux files d'attente d'abonné définies dans les messages de commande **Register Subscriber** correspondants.

Voir Message de réponse du gestionnaire de files d'attente pour plus de détails sur les paramètres de descripteur de message (MQMD) requis lors de l'envoi d'un message de commande au gestionnaire de files d'attente et utilisés lorsqu'un gestionnaire de files d'attente transmet une publication à un abonné.

Le gestionnaire de files d'attente réachemine le message **Publish** vers d'autres gestionnaires de files d'attente du réseau qui possèdent des abonnements correspondants, sauf s'il s'agit d'une publication locale.

Les données de publication, le cas échéant, sont incluses dans le corps du message. Les données peuvent être décrites dans un dossier <mcd> dans la zone NameValueData de l'en-tête MQRFH2 .

Propriétés

< Command> (*MQPSC_COMMAND*)
La valeur est Publish(*MQPSC_PUBLISH*).

Cette propriété doit être spécifiée.

< Topic> (*MQPSC_TOPIC*)
La valeur est une chaîne qui contient une rubrique qui catégorise cette publication. Aucun caractère générique n'est autorisé.

Vous devez ajouter la rubrique à la liste de noms SYSTEM.QPUBSUB.QUEUE.NAMELIST, voir Ajout d'un flux pour savoir comment effectuer cette tâche.

Cette propriété doit être spécifiée et peut éventuellement être répétée pour autant de rubriques que nécessaire.

<SubPoint> (*MQPSC_SUBSCRIPTION_POINT*)
Point d'abonnement sur lequel la publication est publiée.

Dans WebSphere Event Broker V6, la valeur de la propriété <SubPoint> est la valeur de l'attribut de point d'abonnement du noeud Publication qui gère la publication.

Dans WebSphere MQ V7.0.1, la valeur de la propriété <SubPoint> doit correspondre au nom d'un point d'abonnement. Voir Ajout d'un point d'abonnement .

<PubOpt> (*MQPSC_PUBLICATION_OPTION*)
La propriété des options de publication peut prendre les valeurs suivantes:

RetainPub
(*MQPSC_RETAIN_PUB*)

Le gestionnaire de files d'attente doit conserver une copie de la publication. Si cette option n'est pas définie, la publication est supprimée dès que le gestionnaire de files d'attente l'a envoyée à tous ses abonnés actuels.

IsRetainedconservée
(*MQPSC_IS_RETAINED_PUB*)

(ne peut être défini que par un gestionnaire de files d'attente.) Cette publication a été conservée par le gestionnaire de files d'attente. Le gestionnaire de files d'attente définit cette option pour signaler à un abonné que cette publication a été publiée précédemment et qu'elle a été conservée, à condition que l'abonnement ait été enregistré avec l'option InformIfConservation . Il est défini uniquement en réponse à un message de commande Register Subscriber ou Request Update . Cette option n'est pas définie pour les publications conservées qui sont envoyées directement aux abonnés.

Local
(*MQPSC_LOCAL*)

Cette option indique au gestionnaire de files d'attente que cette publication ne doit pas être envoyée à d'autres gestionnaires de files d'attente. Tous les abonnés enregistrés sur ce gestionnaire de files d'attente reçoivent cette publication s'ils possèdent des abonnements correspondants.

OtherSubsuniquement
(*MQPSC_OTHER_SUBS_ONLY*)

Cette option permet un traitement plus simple des applications de type conférence, où un diffuseur de publications est également abonné à la même rubrique. Il indique au gestionnaire de files d'attente de ne pas envoyer la publication à la file d'attente de l'abonné du diffuseur de publications même s'il possède un abonnement correspondant. La file d'attente

d'abonné du diffuseur de publications se compose de son QMgrName, de son QNameet de son CorrelIdfacultatif, comme décrit dans la liste suivante.

IDCorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

CorrelId dans le MQMD (qui doit être différent de zéro) fait partie de la file d'attente d'abonné du diffuseur de publications, dans les applications où le diffuseur de publications est également un abonné.

NONE
(*MQPSC_NONE*)

Toutes les options prennent leurs valeurs par défaut. Cette opération a le même effet que l'omission de la propriété des options de publication. Si d'autres options sont spécifiées en même temps, Aucun est ignoré.

Vous pouvez avoir plusieurs options de publication en introduisant des éléments <PubOpt> supplémentaires.

Par défaut, si cette propriété est omise, aucune option de publication n'est définie.

<PubTime> (*MQPSC_PUBLISH_TIMESTAMP*)
La valeur est un horodatage de publication facultatif défini par le diffuseur de publications. Sa longueur est de 16 caractères avec le format suivant:

```
YYYYMMDDHHMMSSTH
```

en utilisant le temps universel. Ces informations ne sont pas vérifiées par le gestionnaire de files d'attente avant d'être envoyées aux abonnés.

<SeqNum> (*MQPSC_SEQUENCE_NUMBER*)
La valeur est un numéro de séquence facultatif défini par le diffuseur de publications.

Il doit être incrémenté de 1 à chaque publication. Toutefois, cela n'est pas vérifié par le gestionnaire de files d'attente, qui se contente de transmettre ces informations aux abonnés.

Si des publications d'une même rubrique sont publiées dans des gestionnaires de files d'attente interconnectés différents, il incombe aux diffuseurs de s'assurer que les numéros de séquence, s'ils sont utilisés, sont significatifs.

<QMgrName> (*MQPSC_Q_MGR_NAME*)
La valeur est une chaîne contenant le nom du gestionnaire de files d'attente pour la file d'attente d'abonné du diffuseur de publications, dans les applications où le diffuseur de publications est également un abonné (voir OtherSubsuniquement).

Si cette propriété est omise, la valeur par défaut est le nom ReplyToQMgr dans le descripteur de message (MQMD). Si le nom obtenu est vide, le nom du gestionnaire de files d'attente est utilisé par défaut.

< nom_Q> (*MQPSC_Q_NAME*)
La valeur est une chaîne contenant le nom de la file d'attente d'abonné du diffuseur de publications, dans les applications où le diffuseur de publications est également abonné (voir OtherSubsuniquement).

Si cette propriété est omise, la valeur par défaut est le nom ReplyToQ dans le descripteur de message (MQMD), qui ne doit pas être vide si OtherSubsOnly est défini.

Exemple

Voici quelques exemples de *NameValueData* pour un message de commande **Publish** .

Le premier exemple concerne une publication envoyée par le simulateur de correspondance dans l'exemple d'application pour indiquer qu'une correspondance a démarré.

```
<psc>
 <Command>Publish</Command>
 <Topic>Sport/Soccer/Event/MatchStarted</Topic>
</psc>
```

Le deuxième exemple concerne une publication conservée. Le dernier score de la correspondance entre Team1 et Team2 est publié.

```
<psc>
 <Command>Publish</Command>
 <PubOpt>RetainPub</PubOpt>
 <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
</psc>
```

Enregistrement du message d'abonnement

Le message de commande **Register Subscriber** est envoyé à un gestionnaire de files d'attente par un abonné, ou par une autre application pour le compte d'un abonné, pour indiquer qu'il souhaite s'abonner à une ou plusieurs rubriques à un point d'abonnement. Un filtre de contenu de message peut également être spécifié.

Dans les expressions de filtre de publication / abonnement, l'imbrication de parenthèses entraîne une diminution exponentielle des performances. Evitez d'imbriquer des parenthèses à une profondeur supérieure à environ 6.

Le message est envoyé à SYSTEM.BROKER.CONTROL.QUEUE, qui est la file d'attente de contrôle du gestionnaire de files d'attente. Le droit d'insertion d'un message dans cette file d'attente est requis, en plus du droit d'accès (défini par l'administrateur système du gestionnaire de files d'attente) pour la ou les rubriques de l'abonnement.

Si l'utilisateur dispose de droits sur certaines rubriques, mais pas sur toutes, seules les personnes disposant de droits d'accès sont enregistrées ; une réponse d'avertissement indique celles qui ne sont pas enregistrées.

Pour plus de détails sur les paramètres de descripteur de message (MQMD) nécessaires lors de l'envoi d'un message de commande au gestionnaire de files d'attente, voir «Paramètres MQMD dans les messages de commande envoyés au gestionnaire de files d'attente», à la page 888 .

Si la file d'attente de réponses est une file d'attente dynamique temporaire, l'abonnement est désenregistré automatiquement par le gestionnaire de files d'attente lorsque la file d'attente est fermée.

Propriétés

< Command> (*MQPSC_COMMAND*)
La valeur est RegSub (*MQPSC_REGISTER_SUBSCRIBER*). Cette propriété doit être spécifiée.

< Topic> (*MQPSC_TOPIC*)
Rubrique pour laquelle l'abonné souhaite recevoir des publications. Les caractères génériques peuvent être spécifiés dans le cadre de la rubrique.

Si vous utilisez la commande MQSC **display sub** pour examiner l'abonnement créé de cette manière, la valeur de la balise < Topic> s'affiche en tant que propriété TOPICSTR de l'abonnement.

Cette propriété est obligatoire et peut éventuellement être répétée pour autant de rubriques que nécessaire.

<SubPoint> (*MQPSC_SUBSCRIPTION_POINT*)
La valeur est le point d'abonnement auquel l'abonnement est associé.

Si cette propriété est omise, le point d'abonnement par défaut est utilisé.

Dans WebSphere Event Broker V6, la valeur de la propriété <SubPoint> doit correspondre à la valeur de l'attribut Point d'abonnement des noeuds de publication abonnés.

Dans WebSphere MQ V7.0.1, la valeur de la propriété <SubPoint> doit correspondre au nom d'un point d'abonnement. Voir Ajout d'un point d'abonnement .

< Filtre > (*MQPSC_FILTER*)
La valeur est une expression SQL utilisée comme filtre sur le contenu des messages de publication. Si une publication sur la rubrique spécifiée correspond au filtre, elle est envoyée à l'abonné. Cette propriété correspond à la chaîne de sélection utilisée dans les appels MQSUB et MQOPEN. Pour plus d'informations, voir Sélection sur le contenu d'un message

Si cette propriété est omise, aucun filtrage de contenu n'est effectué.

<RegOpt> (*MQPSC_REGISTRATION_OPTION*)
Cette propriété Options d'enregistrement peut prendre les valeurs suivantes:

AddName
(*MQPSC_ADD_NAME*)

Lorsqu'il est spécifié pour un abonnement existant qui correspond à l'identité traditionnelle de cette commande Register Subscription, mais sans valeur SubName en cours, le SubName spécifié dans cette commande est ajouté à l'abonnement.

Si AddName est spécifié, la zone SubName est obligatoire, sinon MQRCCF_REG_OPTIONS_ERROR est renvoyé.

IDCorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

CorrelId dans le descripteur de message (MQMD) est utilisé lors de l'envoi de publications correspondantes à la file d'attente de l'abonné. CorrelId ne doit pas être égal à zéro,

FullResp
(*MQPSC_FULL_RESPONSE*)

Lorsqu'il est spécifié, tous les attributs de l'abonnement sont renvoyés dans le message de réponse, si la commande n'échoue pas.

FullResp est valide uniquement lorsque le message de commande fait référence à un abonnement unique. Par conséquent, une seule rubrique est autorisée dans la commande ; sinon, la commande échoue avec le code retour *MQRCCF_REG_OPTIONS_ERROR*.

InformIfRet
(*MQPSC_INFORM_IF_RETAINED*)

Le gestionnaire de files d'attente informe l'abonné si une publication est conservée lorsqu'il envoie un message de publication en réponse à un message de commande **Register Subscriber** ou **Request Update** . Pour ce faire, le gestionnaire de files d'attente inclut l'option de publication IsRetainedPub dans le message.

JoinExcl
(*MQPSC_JOIN_EXCLUSIVE*)

Cette option indique que la SubIdentity spécifiée doit être ajoutée en tant que membre exclusif de l'ensemble d'identités pour l'abonnement et qu'aucune autre identité ne peut être ajoutée à l'ensemble.

Si l'identité a déjà rejoint'shared'et est la seule entrée de l'ensemble, l'ensemble est remplacé par un verrou exclusif détenu par cette identité. Sinon, si l'abonnement possède actuellement d'autres identités dans l'ensemble d'identités (avec accès partagé), la commande échoue avec le code retour *MQRCCF_SUBSCRIPTION_IN_USE*.

JoinShared
(*MQPSC_JOIN_SHARED*)

Cette option indique que la SubIdentity spécifiée doit être ajoutée à l'ensemble d'identités de l'abonnement.

Si l'abonnement est actuellement verrouillé exclusivement (à l'aide de l'option JoinExcl), la commande échoue avec le code retour *MQRCCF_SUBSCRIPTION_LOCKED*, sauf si l'identité de l'abonnement verrouillé est identique à celle de ce message de commande. Dans ce cas, le verrou est automatiquement remplacé par un verrou partagé.

Local
(*MQPSC_LOCAL*)

L'abonnement est local et n'est pas distribué aux autres gestionnaires de files d'attente du réseau. Les publications effectuées dans d'autres gestionnaires de files d'attente ne sont pas distribuées à cet abonné, à moins qu'il ne dispose également d'un abonnement global correspondant.

NewPubsuniquement
(*MQPSC_NEW_PUBS_ONLY*)

Les publications conservées qui existent au moment de l'enregistrement de l'abonnement ne sont pas envoyées à l'abonné ; seules les nouvelles publications sont envoyées.

Si un abonné se réenregistre et modifie cette option de sorte qu'elle ne soit plus définie, une publication qui lui a déjà été envoyée peut être envoyée à nouveau.

NoAlter
(*MQPSC_NO_ALTER*)

Les attributs d'un abonnement correspondant existant ne sont pas modifiés.

Lorsqu'un abonnement est créé, cette option est ignorée. Toutes les autres options spécifiées s'appliquent au nouvel abonnement.

Si `SubIdentity` comporte également l'une des options de jointure (`JoinExcl` ou `JoinShared`), l'identité est ajoutée à l'ensemble d'identités, que `NoAlter` soit spécifié ou non.

NONE
(*MQPSC_NONE*)

Toutes les options d'enregistrement prennent leurs valeurs par défaut.

Si l'abonné est déjà enregistré, ses options sont réinitialisées à leurs valeurs par défaut (notez que cela n'a *pas* le même effet que l'omission de la propriété des options d'enregistrement) et l'expiration de l'abonnement est mise à jour à partir du MQMD du message **Register Subscriber** .

Si d'autres options d'enregistrement sont spécifiées en même temps, `Aucun` est ignoré.

NonPers
(*MQPSC_NON_PERSISTENT*)

Les publications correspondant à cet abonnement sont distribuées à l'abonné sous forme de messages non persistants.

Pers
(*MQPSC_persISTENT*)

Les publications correspondant à cet abonnement sont distribuées à l'abonné sous forme de messages persistants.

PersAsPub
(*MQPSC_PERSISTENT_AS_PUBLISH*)

Les publications correspondant à cet abonnement sont distribuées à l'abonné avec la persistance spécifiée par le diffuseur de publications. Il s'agit du comportement par défaut.

File d'attentePersAs
(*MQPSC_PERSISTENT_AS_Q*)

Les publications correspondant à cet abonnement sont distribuées à l'abonné avec la persistance spécifiée dans la file d'attente de l'abonné.

PubOnReqOnly
(*MQPSC_PUB_ON_REQUEST_ONLY*)

Le gestionnaire de files d'attente n'envoie pas de publications à l'abonné, sauf en réponse à un message de commande **Request Update** .

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

Lorsqu'elle est spécifiée, l'identité de l'abonné (file d'attente, gestionnaire de files d'attente et ID corrélation) n'est pas limitée à un seul ID utilisateur. Ce comportement diffère du comportement existant du gestionnaire de files d'attente qui associe l'ID utilisateur du message d'enregistrement d'origine à l'identité de l'abonné et empêche ensuite tout autre utilisateur d'utiliser cette identité. Si un nouvel abonné tente d'utiliser la même identité, *MQRCCF_DUPLICATE_SUBSCRIPTION* est renvoyé.

Cela permet à n'importe quel utilisateur de modifier ou de désenregistrer l'abonnement si l'utilisateur dispose des droits appropriés. Il n'est donc pas nécessaire de vérifier que l'ID utilisateur correspond à celui de l'abonné d'origine.

Pour ajouter cette option à un abonnement existant, la commande doit provenir du même ID utilisateur que l'abonnement d'origine lui-même.

Si `VariableUserId` est défini pour l'abonnement de la commande **`Request Update`** , il doit être défini lors de la mise à jour de la demande pour indiquer l'abonnement auquel il est fait référence. Sinon, l'ID utilisateur de la commande **`Request Update`** est utilisé pour identifier l'abonnement. Elle est remplacée, avec les autres identificateurs d'abonné, si un nom d'abonnement est fourni.

Si un message de commande **`Register Subscriber`** sans ce jeu d'options fait référence à un abonnement existant ayant ce jeu d'options, l'option est supprimée de cet abonnement et l'ID utilisateur de l'abonnement est désormais corrigé. S'il existe déjà un abonné ayant la même identité (file d'attente, gestionnaire de files d'attente et identificateur de corrélation) mais avec un ID utilisateur différent associé, la commande échoue avec le code retour *MQRCCF_DUPLICATE_IDENTITY* car un seul ID utilisateur peut être associé à une identité d'abonné.

Si la propriété des options d'enregistrement est omise et que l'abonné est déjà enregistré, ses options d'enregistrement ne sont pas modifiées et l'expiration de l'abonnement est mise à jour à partir du MQMD du message **`Register Subscriber`** .

Si l'abonné n'est pas déjà enregistré, un nouvel abonnement est créé avec toutes les options d'enregistrement prenant leurs valeurs par défaut.

Les valeurs par défaut sont `PersAsPub` et aucune autre option n'est définie.

<QMgrName> (*MQPSC_Q_MGR_NAME*)
La valeur est le nom du gestionnaire de files d'attente pour la file d'attente de souscription, à laquelle les publications correspondantes sont envoyées par le gestionnaire de files d'attente.

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQMgr` dans le descripteur de message (MQMD). Si le nom obtenu est vide, la valeur par défaut est `QMgrName`du gestionnaire de files d'attente.

< nom_Q> (*MQPSC_Q_NAME*)
La valeur est le nom de la file d'attente de souscription, à laquelle les publications correspondantes sont envoyées par le gestionnaire de files d'attente.

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQ` dans le descripteur de message (MQMD), qui ne doit pas être vide dans ce cas.

Si la file d'attente est une file d'attente dynamique temporaire, la distribution non permanente des publications (`NonPers`) doit être spécifiée dans la propriété `<RegOpt>` .

Si la file d'attente est une file d'attente dynamique temporaire, l'abonnement est désenregistré automatiquement par le gestionnaire de files d'attente lorsque la file d'attente est fermée.

<SubName> (*MQPSC_SUBSCRIPTION_NAME*)
Il s'agit d'un nom donné à un abonnement particulier. Vous pouvez l'utiliser à la place du gestionnaire de files d'attente, de la file d'attente et du correlId facultatif pour faire référence à un abonnement.

Si un abonnement existe déjà avec ce **SubName**, tous les autres attributs de l'abonnement (Topic, QMgrName, QName, CorrelId, UserId, RegOpts, UserSubData et Expiry) sont remplacés par les attributs, s'ils sont spécifiés, qui sont transmis dans le nouveau message de commande `Register Subscriber` . Toutefois, si **SubName** est utilisé alors qu'aucune zone QName n'est spécifiée et qu'une file d'attente ReplyToest spécifiée dans l'en-tête MQMD, la file d'attente de souscription est remplacée par la file d'attente ReplyTo.

Si un abonnement correspondant à l'identité traditionnelle de cette commande existe déjà, mais qu'il n'a pas de **SubName**, la commande d'enregistrement échoue avec le code retour *MQRCCF_DUPLICATE_SUBSCRIPTION*, sauf si l'option **AddName** est spécifiée.

Si vous tentez de modifier un abonnement nommé existant à l'aide d'une autre commande `Register Subscriber` qui spécifie le même **SubName**et que les valeurs de Topic, QMgrName, QName et CorrelId dans la nouvelle commande correspondent à un abonnement existant différent, avec ou sans SubName défini, la commande échoue avec le code retour *MQRCCF_DUPLICATE_SUBSCRIPTION*. Cela permet d'éviter que deux noms d'abonnement se réfèrent au même abonnement.

<SubIdentity> (*MQPSC_SUBSCRIPTION_IDENTITY*)
Cette chaîne est utilisée pour représenter une application ayant un intérêt dans un abonnement. Il s'agit d'une chaîne de caractères de longueur variable d'une longueur maximale de 64 caractères, facultative. Le gestionnaire de files d'attente gère un ensemble d'identités d'abonné pour chaque abonnement. Chaque abonnement peut autoriser son ensemble d'identités à contenir une seule identité ou un nombre illimité d'identités (voir les options **JoinShared** et **JoinExcl**).

Une commande d'abonnement qui spécifie l'option **JoinShared** ou **JoinExcl** ajoute l'option **SubIdentity** à l'ensemble d'identités de l'abonnement, si ce n'est pas déjà fait et si l'ensemble d'identités existant autorise une telle action, c'est-à-dire qu'aucun autre abonné ne s'est joint exclusivement ou que l'ensemble d'identités est vide.

Toute modification des attributs de l'abonnement résultant d'une commande `Register Subscriber` dans laquelle une **SubIdentity** est spécifiée ne réussit que si elle est le seul membre de l'ensemble des identités de cet abonnement. Sinon, la commande échoue avec le code retour *MQRCCF_SUBSCRIPTION_IN_USE*. Cela empêche les attributs d'un abonnement de changer sans que d'autres abonnés intéressés soient informés.

Si vous spécifiez une chaîne de caractères de plus de 64 caractères, la commande échoue avec le code retour *MQRCCF_SUB_IDENTITY_ERROR*.

<SubUserde sous-utilisateur >(*MQPSC_SUBSCRIPTION_USER_DATA*)
Il s'agit d'une chaîne de texte de longueur variable. La valeur est stockée par le gestionnaire de files d'attente avec l'abonnement, mais n'a aucune influence sur la distribution des publications à l'abonné. La valeur peut être modifiée en s'enregistrant à nouveau dans le même abonnement avec une nouvelle valeur. Cet attribut est destiné à l'utilisation de l'application.

SubUserData est renvoyé dans les informations de métarubrique (*MQCACF_REG_SUB_USER_DATA*) pour un abonnement, le cas échéant.

Si vous spécifiez plusieurs valeurs d'option d'enregistrement `NonPers, PersAsPub, PersAsQueue, and Pers`, seule la dernière est utilisée. Vous ne pouvez pas combiner ces options dans un abonnement individuel.

Exemple

Voici un exemple de `NameValueData` pour un message de commande **Register Subscriber** . Dans l'exemple d'application, le service de résultats utilise ce message pour enregistrer un abonnement aux rubriques contenant les derniers scores dans toutes les correspondances, avec l'option'Persistent as publish'définie. L'identité de l'abonné, y compris `CorrelId`, est extraite des valeurs par défaut du MQMD.

```
<psc>
 <Command>RegSub</Command>
 <RegOpt>PersAsPub</RegOpt>
 <RegOpt>CorrelAsId</RegOpt>
```

```
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
 </psc>
```

Message de demande de mise à jour

Le message de commande **Request Update** est envoyé d'un abonné à un gestionnaire de files d'attente pour demander les publications conservées en cours pour la rubrique et le point d'abonnement spécifiés qui correspondent au filtre donné (facultatif).

Ce message est envoyé à *SYSTEM.BROKER.CONTROL.QUEUE*, file d'attente de contrôle du gestionnaire de files d'attente. Le droit d'insertion d'un message dans cette file d'attente est requis, en plus du droit d'accès à la rubrique dans la mise à jour de la demande ; ce droit est défini par l'administrateur système du gestionnaire de files d'attente.

Cette commande est normalement utilisée si l'abonné a spécifié l'option PubOnReqOnly lors de son enregistrement. Si le gestionnaire de files d'attente possède des publications conservées correspondantes, elles sont envoyées à l'abonné. Si le gestionnaire de files d'attente ne possède pas de publications conservées correspondantes, la demande échoue avec le code retour *MQRCCF_NO_RETAINED_MSG*. Le demandeur doit avoir préalablement enregistré un abonnement avec les mêmes valeurs de rubrique, SubPointet de filtre.

Propriétés

< Command> (*MQPSC_COMMAND*)
La valeur est ReqUpdate (*MQPSC_REQUEST_UPDATE*). Cette propriété doit être spécifiée.

< Topic> (*MQPSC_TOPIC*)
La valeur est la rubrique que l'abonné demande ; les caractères génériques sont autorisés.

Cette propriété doit être spécifiée, mais une seule occurrence est autorisée dans ce message.

<SubPoint> (*MQPSC_SUBSCRIPTION_POINT*)
La valeur est le point d'abonnement auquel l'abonnement est associé.

Si cette propriété est omise, le point d'abonnement par défaut est utilisé.

< Filtre > (*MQPSC_FILTER*)
La valeur est une expression ESQL utilisée comme filtre sur le contenu des messages de publication. Si une publication sur la rubrique spécifiée correspond au filtre, elle est envoyée à l'abonné.

La propriété < Filtre > doit avoir la même valeur que celle spécifiée sur l'abonnement d'origine pour lequel vous demandez une mise à jour.

Si cette propriété est omise, aucun filtrage de contenu n'est effectué.

<RegOpt> (*MQPSC_REGISTRATION_OPTION*)
La propriété des options d'enregistrement peut prendre la valeur suivante:

IDCorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

Le CorrelId du descripteur de message (MQMD), qui doit être différent de zéro, est utilisé lors de l'envoi de publications correspondantes à la file d'attente de l'abonné.

NONE
(*MQPSC_NONE*)

Toutes les options prennent leurs valeurs par défaut. Cela a le même effet que l'omission de la propriété <RegOpt> . Si d'autres options sont spécifiées en même temps, Aucun est ignoré.

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

Lorsqu'elle est spécifiée, l'identité de l'abonné (file d'attente, gestionnaire de files d'attente et ID corrélation) n'est pas limitée à un seul ID utilisateur. Ce comportement diffère du comportement existant du gestionnaire de files d'attente qui associe l'ID utilisateur du message d'enregistrement d'origine à l'identité de l'abonné et empêche ensuite tout autre utilisateur d'utiliser cette identité.

Si un nouvel abonné tente d'utiliser la même identité, la commande échoue avec le code retour *MQRCCF_DUPLICATE_SUBSCRIPTION*.

Cela permet à tout utilisateur de modifier ou de désenregistrer l'abonnement lorsqu'il dispose des droits appropriés. Par conséquent, il n'est pas nécessaire de vérifier que l'ID utilisateur correspond à celui de l'abonné d'origine.

Pour ajouter cette option à un abonnement existant, la commande doit provenir du même ID utilisateur que l'abonnement d'origine.

Si `VariableUserId` est défini pour l'abonnement de la commande **Request Update** , il doit être défini lors de la mise à jour de la demande pour indiquer l'abonnement auquel il est fait référence. Sinon, l'ID utilisateur de la commande **Request Update** est utilisé pour identifier l'abonnement. Elle est remplacée, avec les autres identificateurs d'abonné, si un nom d'abonnement est fourni.

Par défaut, si cette propriété est omise, aucune option d'enregistrement n'est définie.

<QMgrName> (*MQPSC_Q_MGR_NAME*)
La valeur est le nom du gestionnaire de files d'attente de la file d'attente de souscription, à laquelle la publication conservée correspondante est envoyée par le gestionnaire de files d'attente.

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQMgr` dans le descripteur de message (MQMD). Si le nom obtenu est vide, la valeur par défaut est `QMgrName`du gestionnaire de files d'attente.

< nom_Q> (*MQPSC_Q_NAME*)
La valeur est le nom de la file d'attente de souscription, à laquelle la publication conservée correspondante est envoyée par le gestionnaire de files d'attente.

Si cette propriété est omise, la valeur par défaut est le nom `ReplyToQ` dans le descripteur de message (MQMD), qui ne doit pas être vide dans ce cas.

<SubName> (*MQPSC_SUBSCRIPTION_NAME*)
Il s'agit d'un nom donné à un abonnement particulier. Si elle est spécifiée dans une commande **Request Update** , la valeur `SubName` est prioritaire sur toutes les autres zones d'identificateur à l'exception de l'ID utilisateur, sauf si `VariableUserId` est défini sur l'abonnement lui-même. Si `VariableUserId` n'est pas défini, la commande *Request Update* aboutit uniquement si l'ID utilisateur du message de commande correspond à celui de l'abonnement. Si l'ID utilisateur du message de commande ne correspond pas à celui de l'abonnement, la commande échoue avec le code retour *MQRCCF_DUPLICATE_IDENTITY*.

Si `VariableUserId` est défini et que l'ID utilisateur est différent de celui de l'abonnement, la commande aboutit si l'ID utilisateur du nouveau message de commande est autorisé à parcourir la file d'attente de flux et à l'insérer dans la file d'attente d'abonné de l'abonnement. Sinon, la commande échoue avec le code retour *MQRCCF_NOT_AUTHORIZED*.

S'il existe un abonnement correspondant à l'identité traditionnelle de cette commande, mais sans `SubName`, la commande**Request Update** échoue avec le code retour *MQRCCF_SUB_NAME_ERROR*.

Si une tentative de demande de mise à jour est effectuée pour un abonnement dont le `SubName` utilise un message de commande qui correspond à l'identité traditionnelle, mais sans `SubName` spécifié, la commande aboutit.

Exemple

Voici un exemple de `NameValueData` pour un message de commande **Request Update** . Dans l'exemple d'application, le service de résultats utilise ce message pour demander des publications conservées contenant les derniers scores pour toutes les équipes. L'identité de l'abonné, y compris `CorrelId`, est extraite des valeurs par défaut du MQMD.

```
<psc>
 <Command>ReqUpdate</Command>
 <RegOpt>CorrelAsId</RegOpt>
```

```
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
 </psc>
```

Message de réponse du gestionnaire de files d'attente

Un message **Queue Manager Response** est envoyé d'un gestionnaire de files d'attente à la file d'attente ReplyToQ d'un diffuseur de publications ou d'un abonné, pour indiquer la réussite ou l'échec d'un message de commande reçu par le gestionnaire de files d'attente si le descripteur de message de commande indique qu'une réponse est requise.

Le message de réponse est contenu dans la zone NameValueData de l'en-tête MQRFH2 , dans un dossier <pscr> .

En cas d'avertissement ou d'erreur, le message de réponse contient le dossier <psc> du message de commande ainsi que le dossier <pscr> . Les données de message, le cas échéant, ne sont pas contenues dans le message de réponse du gestionnaire de files d'attente. Dans le cas d'une erreur, aucun des messages à l'origine d'une erreur n'a été traité ; dans le cas d'un avertissement, certains messages peuvent avoir été traités avec succès.

En cas d'échec lors de l'envoi d'une réponse:

- Pour les messages de publication, le gestionnaire de files d'attente tente d'envoyer la réponse à la file d'attente de rebut WebSphere MQ en cas d'échec de MQPUT. Cela permet d'envoyer la publication aux abonnés même si la réponse ne peut pas être renvoyée au diffuseur de publications.
- Pour les autres messages ou si la réponse de publication ne peut pas être envoyée à la file d'attente de rebut, une erreur est consignée et le message de commande est normalement annulé. Cela dépend de la manière dont le noeud MQInput a été configuré.

Propriétés

< Completion> (*MQPSCR_COMPLETION*)
Le code achèvement, qui peut prendre l'une des trois valeurs suivantes:

OK
La commande a abouti

avertissement
Commande terminée mais avec avertissement

error
La commande a échoué

< Réponse> (*MQPSCR_RESPONSE*)
La réponse à un message de commande, si cette commande a généré le code achèvement warning ou error. Il contient une propriété < Reason> et peut contenir d'autres propriétés qui indiquent la cause de l'avertissement ou de l'erreur.

Dans le cas d'une ou de plusieurs erreurs, il n'y a qu'un seul dossier de réponse, indiquant la cause de la première erreur uniquement. Dans le cas d'un ou de plusieurs avertissements, il existe un dossier de réponses pour chaque avertissement.

< raison> (*MQPSCR_REASON*)
Code raison qualifiant le code achèvement, si le code achèvement est un avertissement ou une erreur. Il est défini sur l'un des codes d'erreur répertoriés dans l'exemple suivant. La propriété < Reason> est contenue dans un dossier < Response> . Le code anomalie peut être suivi de toute propriété valide du dossier <psc> (par exemple, un nom de rubrique), indiquant la cause de l'erreur ou de l'avertissement. Si vous obtenez un code raison de? ????, Vérifiez que les données sont correctes, par exemple, les crochets correspondants (< >).

Exemples

Voici quelques exemples de NameValueData dans un message **Queue Manager Response** . Une réponse réussie peut être la suivante:

```
<pscr>
  <Completion>ok</Completion>
</pscr>
```

Voici un exemple de réponse d'échec ; l'échec est une erreur de filtre. La première chaîne NameValueData contient la réponse ; la seconde contient la commande d'origine.

```
<pscr>
  <Completion>error</Completion>
  <Response>
    <Reason>3150</Reason>
  </Reponse>
</pscr>

<psc>
 ...
 command message (to which
 the queue manager is responding)
 ...
</psc>
```

Voici un exemple de réponse d'avertissement (due à des sujets non autorisés). La première chaîne NameValueData contient la réponse ; la deuxième chaîne NameValueData contient la commande d'origine.

```
<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>
```

Codes raison de publication / abonnement

Ces codes anomalie peuvent être renvoyés dans la zone Raison d'un dossier de réponse de publication / abonnement <pscr> . Les constantes qui peuvent être utilisées pour représenter ces codes dans les langages de programmation C ou C + + sont également répertoriées.

Les constantes MQRC_ requièrent le fichier d'en-tête WebSphere MQ cmqc.h . Les constantes MQRCCF_ requièrent le fichier d'en-tête WebSphere MQ cmqcfc.h (à l'exception de *MQRCCF_FILTER_ERROR* et *MQRCCF_INJUSTEMENT _user*, qui requièrent le fichier d'en-tête cmqpsc.h).

Code anomalie et texte	Explication	Emis par
2336 MQRC_RFH_ERREUR DE COMMANDE	Les valeurs valides pour la zone < Command> d'un dossier <psc> sont: RegSub, DeregSub, Publish, DeletePubet ReqUpdate. Toute autre valeur entraîne l'émission de ce code d'erreur.	Toute commande

Code anomalie et texte	Explication	Emis par
2337 MQRC_RFH_ ERREUR DE PARM	Les dossiers <psc> et <mcd> ont tous deux un ensemble de paramètres valides qui peuvent être spécifiés dans ces deux dossiers. Vérifiez les descriptions de ces dossiers et assurez-vous que vous n'avez pas indiqué de paramètres incorrects.	Toute commande
2338 MQRC_RFH_DUPLICATE_PARM	Certains paramètres (par exemple, Rubrique) d'un dossier <psc> peuvent être répétés, tandis que d'autres (par exemple, Commande) ne peuvent pas être répétés. Vérifiez que vous n'avez pas dupliqué de paramètre non reproductible.	Toute commande
2339 MQRC_RFH_PARM_MISSING	Certains paramètres dans les dossiers <psc> ou <mcd> sont facultatifs et peuvent être omis ; d'autres sont obligatoires et ne doivent pas être omis. Vérifiez que vous avez inclus tous les paramètres obligatoires dans vos dossiers <psc> et <mcd> .	Toute commande
2551 MQRC_SELECTION_NOT_AVAILABLE	Aucun fournisseur de sélection de messages étendue n'était disponible pour déterminer quels abonnés avec un filtre spécifié devaient recevoir la publication.	Publier, enregistrer un abonné et demander la mise à jour
	Aucun fournisseur de sélection de message étendu n'est disponible pour traiter le filtre de l'abonné spécifié.	Enregistrer l'abonné et demander la mise à jour
2554 MQRC_CONTENT_ERREUR	Un fournisseur de sélection de messages étendue a détecté une erreur dans la publication en cours ou conservée.	Publier et demander la mise à jour
3008 Echec de la commande MQRCCF_COMMAND_FAILED	Une erreur interne s'est produite et a empêché l'exécution correcte de la commande. L'erreur peut se produire si la commande est réémise. Le journal des événements système du gestionnaire de files d'attente contient des informations qui doivent être utilisées pour signaler le problème à IBM.	Toute commande
3072 MQRCCF_TOPIC_ERREUR	Une ou plusieurs des valeurs que vous avez fournies pour le paramètre Topic sont incorrectes. Vérifiez que vos valeurs de rubrique sont conformes aux restrictions spécifiées.	Toute commande

Code anomalie et texte	Explication	Emis par
3073 MQRCCF_NOT_REGISTERED	La combinaison de SubPoint, Topic et Filter que vous avez spécifiée dans votre commande DeregSub ou ReqUpdate n'était pas une combinaison avec laquelle vous vous étiez précédemment enregistré ou, pour la commande DeregSub si l'option DeregAll était spécifiée, l'une des propriétés SubPoint, Topic ou Filter n'était pas utilisée pour désenregistrer un abonnement.	Instructions de désenregistrement d'abonné et de mise à jour de demande
3074 MQRCCF_Q_MGR_NAME_ERROR	Le gestionnaire de files d'attente spécifié n'est pas valide, ou le gestionnaire de files d'attente n'est pas disponible ou n'existe pas.	Instructions de désenregistrement d'un abonné, de publication, d'enregistrement d'un abonné et de demande de mise à jour
3076 MQRCCF_Q_NOM_ERREUR	Le nom de file d'attente spécifié n'est pas valide ou la file d'attente n'existe pas sur le gestionnaire de files d'attente spécifié.	Instructions de désenregistrement d'un abonné, de publication, d'enregistrement d'un abonné et de demande de mise à jour
3077 MQRCCF_NO_RETAINED_MSG	Il n'y a pas eu de messages conservés pour la rubrique que vous avez spécifiée. Il peut s'agir ou non d'une erreur, selon la conception de votre programme d'application.	Commande de demande de mise à jour
3079 MQRCCF_INCORRECT_Q	Les commandes RegSub, DeregSubet ReqUpdate sont toujours envoyées à SYSTEM.BROKER.CONTROL.QUEUE du gestionnaire de files d'attente auquel ils sont destinés. Les commandes de publication et de suppression de publication sont envoyées à la file d'attente d'entrée du flux de messages de publication / abonnement auquel elles sont destinées ; elles sont déterminées lorsque le flux de messages est conçu. Ce code d'erreur est renvoyé si une commande est envoyée à la mauvaise file d'attente.	Toute commande

Code anomalie et texte	Explication	Emis par
3080 MQRCCF_CORREL_ID_ERREUR	Vous avez spécifié l'ID CorrelAscomme l'un de vos paramètres RegOpt . Toutefois, la zone CorrelId du MQMD ne contient pas d'identificateur de corrélation valide (c'est-à-dire qu'elle est définie sur MQCI_NONE).	Instructions de désenregistrement d'un abonné et d'enregistrement d'un abonné
3081 MQRCCF_NON_AUTORISÉ	Vous n'êtes pas autorisé à effectuer l'action demandée. Les paramètres d'autorisation du gestionnaire de files d'attente sont gérés par l'administrateur système à l'aide de l'éditeur de hiérarchie des rubriques.	Commandes de publication et d'enregistrement d'abonné
3083 MQRCCF_REG_OPTIONS_ERREUR	Vous avez spécifié un paramètre RegOpt non reconnu dans le dossier <psc> qui contient votre commande RegSub ou DeregSub .	Instructions de désenregistrement d'un abonné et d'enregistrement d'un abonné
3084 MQRCCF_PUB_OPTIONS_ERROR	Vous avez spécifié un paramètre PubOpt non reconnu dans le dossier <psc> qui contient votre commande de publication.	Commande de publication
3087 MQRCCF_DEL_OPTIONS_ERROR	Vous avez spécifié un paramètre DelOpt non reconnu dans le dossier <psc> qui contient la commande DeletePub .	Instruction de suppression d'une publication
3150 MQRCCF_FILTER_ERREUR	La valeur indiquée pour le paramètre Filter n'est pas valide. Consultez la section qui décrit la syntaxe valide pour les expressions de filtre et assurez-vous que votre expression est conforme.	Instructions de désenregistrement d'un abonné, d'enregistrement d'un abonné et de demande de mise à jour
3151 MQRCCF_UTILISATEUR_FAUT_ERREUR	Un abonnement correspondant à celui indiqué existe déjà ; cependant, il a été enregistré par un autre utilisateur. Un abonnement ne peut être modifié ou désenregistré que par l'utilisateur qui l'a enregistré à l'origine.	Instructions de désenregistrement d'un abonné, d'enregistrement d'un abonné et de demande de mise à jour
3152 MQRCCF_ABONNEMENT_DOUBLON	Un abonnement correspondant existe déjà avec un nom d'abonnement différent.	
3153 MQRCCF_SUB_NAME_ERREUR	Le format du nom d'abonnement n'est pas valide ou un abonnement correspondant existe déjà sans nom d'abonnement.	

Code anomalie et texte	Explication	Emis par
3154 MQRCCF_SUB_IDENTITY_ERROR	Le paramètre d'identité d'abonnement est erroné. Soit la valeur fournie dépasse la longueur maximale autorisée, soit l'identité de l'abonnement n'est pas actuellement membre de l'ensemble d'identités de l'abonnement et une option d'enregistrement de jointure n'a pas été spécifiée.	
3155 MQRCCF_SUBSCRIPTION_IN_USE	Une tentative de modification ou de désenregistrement d'un abonnement a été effectuée par un membre de l'ensemble d'identités alors qu'il n'était pas le seul membre de cet ensemble.	
3156 MQRCCF_SUBSCRIPTION_LOCKED	L'abonnement est actuellement verrouillé exclusivement par une autre identité.	
3157 MQRCCF_ALREADY_JOINT	Une option d'enregistrement de jointure a été spécifiée, mais l'identité de l'abonné était déjà membre de l'ensemble d'identités de l'abonnement.	

Paramètres MQMD dans les messages de commande envoyés au gestionnaire de files d'attente

Les applications qui envoient des messages de commande au gestionnaire de files d'attente utilisent les paramètres suivants des zones du descripteur de message (MQMD). Les zones qui restent comme valeur par défaut ou qui peuvent être définies sur n'importe quelle valeur valide de la manière habituelle ne sont pas répertoriées ici.

Report
Voir MsgType et CorrelId.

MsgType
MsgType doit être défini sur *MQMT_REQUEST* ou *MQMT_DATAGRAM*. *MQRC_MSG_TYPE_ERROR* est renvoyé si MsgType n'est pas défini sur l'une de ces valeurs.

MsgType doit être défini sur *MQMT_REQUEST* pour un message de commande si une réponse est toujours requise. Les indicateurs MQRO_PAN et MQRO_NAN de la zone Rapport ne sont pas significatifs dans ce cas.

Si MsgType est défini sur *MQMT_DATAGRAM*, les réponses dépendent de la définition des indicateurs MQRO_PAN et MQRO_NAN dans la zone Rapport :

- MQRO_PAN seul signifie que le gestionnaire de files d'attente envoie une réponse uniquement si la commande aboutit.
- MQRO_NAN seul signifie que le gestionnaire de files d'attente envoie une réponse uniquement si la commande échoue.
- Si une commande se termine avec un avertissement, une réponse est envoyée si MQRO_PAN ou MQRO_NAN est défini.
- MQRO_PAN + MQRO_NAN signifie que le gestionnaire de files d'attente envoie une réponse indiquant si la commande aboutit ou échoue. Cela a le même effet du point de vue du gestionnaire de files d'attente que de définir MsgType sur MQMT_REQUEST.

- Si ni MQRO_PAN ni MQRO_NAN n'est défini, aucune réponse n'est jamais envoyée.

Format
Défini sur MQFMT_RF_HEADER_2

MsgId
Cette zone est normalement définie sur MQMI_NONE, de sorte que le gestionnaire de files d'attente génère une valeur unique.

CorrelId
Cette zone peut être définie sur n'importe quelle valeur. Si l'identité de l'expéditeur inclut un `CorrelId`, spécifiez cette valeur, avec MQRO_PASS_CORREL_ID dans la zone `Rapport` , pour vous assurer qu'elle est définie dans tous les messages de réponse envoyés par le gestionnaire de files d'attente à l'expéditeur.

ReplyToQ
Cette zone définit la file d'attente à laquelle les réponses, le cas échéant, doivent être envoyées. Il peut s'agir de la file d'attente de l'expéditeur, ce qui présente l'avantage que le paramètre `QName` peut être omis du message. Toutefois, si des réponses doivent être envoyées à une autre file d'attente, le paramètre `QName` est nécessaire.

ReplyToQMgr
Cette zone définit le gestionnaire de files d'attente pour les réponses. Si vous laissez cette zone vide (valeur par défaut), le gestionnaire de files d'attente local place son propre nom dans cette zone.

Paramètres MQMD pour les publications transmises par un gestionnaire de files d'attente

Un gestionnaire de files d'attente utilise ces paramètres de zones dans le descripteur de message (MQMD) lorsqu'il envoie une publication à un abonné. Toutes les autres zones du MQMD sont définies sur leurs valeurs par défaut.

Report
`Rapport` est défini sur MQRO_NONE.

MsgType
`MsgType` est défini sur MQMT_DATAGRAM.

Expiration
`Expiration` est définie sur la valeur du message `Publish` reçu du diffuseur de publications. Dans le cas d'un message conservé, le temps restant est réduit de l'heure approximative à laquelle le message a été envoyé au gestionnaire de files d'attente.

Format
`Format` est défini sur MQFMT_RF_HEADER_2

MsgId
`MsgId` est défini sur une valeur unique.

CorrelId
Si `CorrelId` fait partie de l'identité de l'abonné, il s'agit de la valeur spécifiée par l'abonné lors de l'enregistrement. Sinon, il s'agit d'une valeur différente de zéro choisie par le gestionnaire de files d'attente.

Priority
`Priority` prend la valeur définie par le diffuseur de publications ou comme résolue si le diffuseur de publications a spécifié MQPRI_PRIORITY_AS_Q_DEF.

Persistance
`Persistence` prend la valeur définie par le diffuseur de publications ou comme résolue si le diffuseur de publications a spécifié MQPER_PERSISTENCE_AS_Q_DEF, sauf indication contraire dans le message `Register Subscriber` pour l'abonné auquel cette publication est envoyée.

ReplyToQ
`ReplyToQ` est mis à blanc.

ReplyToQMgr
ReplyToGestionnaire de files d'attente est défini sur le nom du gestionnaire de files d'attente.

UserIdentifier
UserIdentifier est l'identificateur utilisateur de l'abonné, tel que défini lors de l'enregistrement de l'abonné.

AccountingToken
AccountingToken est le jeton de comptabilité de l'abonné, tel que défini lors de la première inscription de l'abonné.

ApplIdentityData
Les données ApplIdentity sont les données d'identité de l'application de l'abonné, telles qu'elles sont définies lorsque l'abonné est enregistré pour la première fois.

PutApplType
PutApplType est défini sur MQAT_BROKER.

PutApplName
PutApplNom est défini sur les 28 premiers caractères du nom du gestionnaire de files d'attente.

PutDate
PutDate est la date à laquelle le message a été inséré.

PutTime
PutTime est l'heure à laquelle le message a été inséré.

ApplOriginData
ApplOriginLes données sont mises à blanc.

Paramètres MQMD dans les messages de réponse du gestionnaire de files d'attente

Un gestionnaire de files d'attente utilise ces paramètres des zones du descripteur de message (MQMD) lors de l'envoi d'une réponse à un message de publication. Toutes les autres zones du MQMD sont définies sur leurs valeurs par défaut.

Report
Rapport est défini sur des zéros.

MsgType
MsgType est défini sur MQMT_REPLY.

Format
Format est défini sur MQFMT_RF_HEADER_2

MsgId
La valeur de MsgId dépend des options Report du message de commande d'origine. Par défaut, il est défini sur MQMI_NONE, de sorte que le gestionnaire de files d'attente génère une valeur unique.

CorrelId
La valeur de CorrelId dépend des options Report du message de commande d'origine. Par défaut, cela signifie que l'ID CorrelId est défini sur la même valeur que l'ID MsgId du message de commande. Cela peut être utilisé pour corréler les commandes avec leurs réponses.

Priority
Priorityest défini sur la même valeur que dans le message de commande d'origine.

Persistance
Persistence est défini sur la valeur définie dans le message de commande d'origine.

Expiration
Expiration est définie sur la même valeur que dans le message de commande d'origine reçu par le gestionnaire de files d'attente.

PutApplType
PutApplType est défini sur MQAT_BROKER.

PutApplName
PutApplNom est défini sur les 28 premiers caractères du nom du gestionnaire de files d'attente.

D'autres zones de contexte sont définies comme si elles étaient générées avec MQPMO_PASS_IDENTITY_CONTEXT.

Codages de machine

Cette section décrit la structure de la zone *Encoding* dans le descripteur de message.

Pour un récapitulatif des zones de la structure, voir «MQMD-Descripteur de message», à la page 394 .

La zone *Encoding* est un entier 32 bits qui est divisé en quatre sous-zones distinctes ; ces sous-zones identifient:

- Codage utilisé pour les entiers binaires
- Codage utilisé pour les entiers décimaux condensés
- Codage utilisé pour les nombres à virgule flottante
- Bits réservés

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondent au sous-champ, et 0 bit ailleurs. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis:

MQENC_INTEGER_MASK
Masque pour le codage d'entier binaire.

Cette sous-zone occupe les positions de bit 28 à 31 dans la zone *Encoding* .

MQENC_DECIMAL_MASK
Masque pour le codage d'entier décimal condensé.

Cette sous-zone occupe les positions de bit 24 à 27 dans la zone *Encoding* .

MQENC_FLOAT_MASK
Masque pour le codage en virgule flottante.

Cette sous-zone occupe les positions de bit 20 à 23 dans la zone *Encoding* .

MASQUE_RÉSERVÉ_MQENC_MASQUE
Masque pour les bits réservés.

Cette sous-zone occupe les positions de bit 0 à 19 dans la zone *Encoding* .

Codage d'entiers binaires

Les valeurs suivantes sont valides pour le codage d'entier binaire:

MQENC_INTEGER_UNDEFINI
Les entiers binaires sont représentés à l'aide d'un codage non défini.

MQENC_INTEGER_NORMAL
Les entiers binaires sont représentés de manière classique:

- L'octet le moins significatif du nombre a l'adresse la plus élevée de tous les octets du nombre ; l'octet le plus significatif a l'adresse la plus faible
- Le bit de poids faible de chaque octet est adjacent à l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est adjacent à l'octet avec l'adresse inférieure suivante

MQENC_INTEGER_REVERSED
Les entiers binaires sont représentés de la même manière que MQENC_INTEGER_NORMAL, mais avec les octets disposés dans l'ordre inverse. Les bits de chaque octet sont organisés de la même manière que MQENC_INTEGER_NORMAL.

Codage d'entier décimal condensé

Les valeurs suivantes sont valides pour le codage d'entier décimal condensé:

MQENC_DECIMAL_UNDEFINI
Les entiers décimaux condensés sont représentés à l'aide d'un codage non défini.

MQENC_DECIMAL_NORMAL
Les entiers décimaux condensés sont représentés de manière conventionnelle:

- Chaque chiffre décimal dans la forme imprimable du nombre est représenté en décimal condensé par un chiffre hexadécimal unique compris entre X' 0'et X' 9'. Chaque chiffre hexadécimal occupe quatre bits, de sorte que chaque octet du nombre décimal condensé représente deux chiffres décimaux dans la forme imprimable du nombre.
- L'octet le moins significatif dans le nombre décimal condensé est l'octet qui contient le chiffre décimal le moins significatif. Dans cet octet, les quatre bits de poids fort contiennent le chiffre décimal de poids faible, et les quatre bits de poids faible contiennent le signe. Le signe est soit X'C'(positif), X 'D'(négatif), soit X'F'(non signé).
- L'octet le moins significatif du nombre a l'adresse la plus élevée de tous les octets du nombre ; l'octet le plus significatif a l'adresse la plus faible.
- Le bit de poids faible de chaque octet est adjacent à l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est adjacent à l'octet avec l'adresse inférieure suivante.

MQENC_DECIMAL_REVERSED
Les entiers décimaux condensés sont représentés de la même manière que MQENC_DECIMAL_NORMAL, mais avec les octets organisés dans l'ordre inverse. Les bits de chaque octet sont organisés de la même manière que MQENC_DECIMAL_NORMAL.

Codage à virgule flottante

Les valeurs admises pour le codage en virgule flottante sont les suivantes:

MQENC_FLOAT_NON DEFINI
Les nombres à virgule flottante sont représentés à l'aide d'un codage non défini.

MQENC_FLOAT_IEEE_NORMAL
Les nombres à virgule flottante sont représentés à l'aide de la norme IEEE[3]format à virgule flottante, avec les octets organisés comme suit:

- L'octet de poids faible de la mantisse a l'adresse la plus élevée de tous les octets du nombre ; l'octet contenant l'exposant a l'adresse la plus faible
- Le bit de poids faible de chaque octet est adjacent à l'octet avec l'adresse supérieure suivante ; le bit de poids fort de chaque octet est adjacent à l'octet avec l'adresse inférieure suivante

Les détails du codage IEEE float se trouvent dans la norme IEEE 754.

MQENC_FLOAT_IEEE_REVERSED
Les nombres à virgule flottante sont représentés de la même manière que MQENC_FLOAT_IEEE_NORMAL, mais avec les octets organisés dans l'ordre inverse. Les bits de chaque octet sont organisés de la même manière que MQENC_FLOAT_IEEE_NORMAL.

MQENC_FLOAT_S390
Les nombres à virgule flottante sont représentés à l'aide du format à virgule flottante System/390 standard ; ce format est également utilisé par System/370.

Construction de codages

Pour construire une valeur pour la zone *Encoding* dans MQMD, les constantes appropriées qui décrivent les codages requis peuvent être:

[3] L'Institut des ingénieurs électriciens et électroniciens

- Ajouté ensemble, ou
- Combiné à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit)

Quelle que soit la méthode utilisée, combinez un seul des codages MQENC_INTEGER_ * avec l'un des codages MQENC_DECIMAL_ * et l'un des codages MQENC_FLOAT_ *.

Analyse des codages

La zone *Encoding* contient des sous-zones ; pour cette raison, les applications qui doivent examiner le codage de type entier, décimal condensé ou flottant doivent utiliser l'une des techniques décrites.

Utilisation d'opérations de bit

Si le langage de programmation prend en charge les opérations de bit, procédez comme suit:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de codage requis:
   - MQENC_INTEGER_MASK pour le codage d'entier binaire
   - MQENC_DECIMAL_MASK pour le codage d'entier décimal condensé
   - MQENC_FLOAT_MASK pour le codage en virgule flottante

   Appelez la valeur A.
2. Combinez la zone *Encoding* avec A à l'aide de l'opération AND bit à bit ; appelez le résultat B.
3. B est le codage requis et peut être testé pour l'égalité avec chacune des valeurs valides pour ce type de codage.

Utilisation de l'arithmétique

Si le langage de programmation *ne prend pas* en charge les opérations bit, effectuez les étapes suivantes à l'aide de l'arithmétique entière:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de codage requis:
   - 1 pour le codage d'entier binaire
   - 16 pour le codage d'entier décimal condensé
   - 256 pour le codage à virgule flottante

   Appelez la valeur A.
2. Divisez la valeur de la zone *Encoding* par A; appelez le résultat B.
3. Divisez B par 16 ; appelez le résultat C.
4. Multipliez C par 16 et soustrayez de B; appelez le résultat D.
5. Multipliez D par A; appelez le résultat E.
6. E est le codage requis et peut être testé pour l'égalité avec chacune des valeurs valides pour ce type de codage.

Récapitulatif des codages de l'architecture de la machine

Les codages des architectures de machine sont présentés dans la Tableau 578, à la page 893.

Tableau 578. Récapitulatif des codages pour les architectures de machine

Architecture de la machine	Codage d'entier binaire	Codage d'entier décimal condensé	Codage à virgule flottante
IBM i	normale	normale	IEEE normal
Intel x86	reversed	reversed	IEEE inversé

Tableau 578. Récapitulatif des codages pour les architectures de machine (suite)			
Architecture de la machine	**Codage d'entier binaire**	**Codage d'entier décimal condensé**	**Codage à virgule flottante**
PowerPC	normale	normale	IEEE normal
System/390	normale	normale	System/390

Options de rapport et indicateurs de message

Cette section décrit les zones *Report* et *MsgFlags* qui font partie du descripteur de message MQMD spécifié dans les appels MQGET, MQPUT et MQPUT1 .

Les rubriques de cette section décrivent:

- Structure de la zone de rapport et mode de traitement du gestionnaire de files d'attente
- Comment une application analyse la zone de rapport
- Structure de la zone d'indicateurs de message

Pour plus d'informations sur le descripteur de message MQMD, voir «MQMD-Descripteur de message», à la page 394.

Structure de la zone de rapport

Ces informations décrivent la structure de la zone de rapport.

La zone *Report* est un entier 32 bits qui est divisé en trois sous-zones distinctes. Ces sous-zones identifient:

- Options de rapport rejetées si le gestionnaire de files d'attente local ne les reconnaît pas
- Options de rapport toujours acceptées, même si le gestionnaire de files d'attente local ne les reconnaît pas
- Options de rapport acceptées uniquement si certaines autres conditions sont remplies

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondent au sous-champ, et 0 bit ailleurs. Les bits d'un sous-champ ne sont pas nécessairement adjacents. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis pour identifier les sous-zones:

MQRO_REJECT_UNSUP_MASK
Ce masque identifie les positions de bit dans la zone *Report* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local entraînent l'échec de l'appel MQPUT ou MQPUT1 avec le code achèvement MQCC_FAILED et le code anomalie MQRC_REPORT_OPTIONS_ERROR.

Cette sous-zone occupe les positions de bit 3, et 11 à 13.

MQRO_ACCEPT_UNSUP_MASK
Ce masque identifie les positions de bit dans la zone *Report* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local sont néanmoins acceptées sur les appels MQPUT ou MQPUT1 . Le code achèvement MQCC_WARNING avec le code anomalie MQRC_UNKNOWN_REPORT_OPTION est renvoyé dans ce cas.

Cette sous-zone occupe les positions de bit 0 à 2, 4 à 10 et 24 à 31.

Les options de rapport suivantes sont incluses dans cette sous-zone:

- ACTIVITE MQRO
- 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_AUCUN
- MQRO_PAN
- MQRO_PASS_CORREL_ID
- MQRO_PASS_MSG_ID

MQRO_ACCEPT_UNSUP_IF_XMIT_MASK
Ce masque identifie les positions de bit dans la zone *Report* où les options de rapport qui ne sont pas prises en charge par le gestionnaire de files d'attente local sont néanmoins acceptées sur les appels MQPUT ou MQPUT1 *à condition* que les deux conditions suivantes soient remplies:

- Le message est destiné à un gestionnaire de files d'attente éloignées.
- L'application ne place pas le message directement dans une file d'attente de transmission locale (c'est-à-dire que la file d'attente identifiée par les zones *ObjectQMgrName* et *ObjectName* dans le descripteur d'objet spécifié dans l'appel MQOPEN ou MQPUT1 n'est pas une file d'attente de transmission locale).

Le code achèvement MQCC_WARNING avec le code anomalie MQRC_UNKNOWN_REPORT_OPTION est renvoyé si ces conditions sont satisfaites et MQCC_FAILED avec le code anomalie MQRC_REPORT_OPTIONS_ERROR si ce n'est pas le cas.

Cette sous-zone occupe les positions de bit 14 à 23.

Les options de rapport suivantes sont incluses dans cette sous-zone:

- MQRO_COA
- MQRO_COA_WITH_DATA
- MQRO_COA_WITH_FULL_DATA
- MQRO_COD
- MQRO_COD_WITH_DATA
- MQRO_COD_WITH_FULL_DATA

Si des options sont spécifiées dans la zone *Report* que le gestionnaire de files d'attente ne reconnaît pas, le gestionnaire de files d'attente vérifie chaque sous-zone à tour de rôle en utilisant l'opération bit à bit AND pour combiner la zone *Report* avec le masque de cette sous-zone. Si le résultat de cette opération est différent de zéro, le code achèvement et les codes raison décrits ci-dessus sont renvoyés.

Si MQCC_WARNING est renvoyé, il n'est pas défini quel code anomalie est renvoyé si d'autres conditions d'avertissement existent.

La possibilité de spécifier et d'accepter des options de rapport qui ne sont pas reconnues par le gestionnaire de files d'attente local est utile lors de l'envoi d'un message avec une option de rapport qui est reconnue et traitée par un gestionnaire de files d'attente *distant* .

Analyse de la zone de rapport

La zone *Report* contient des sous-zones. Par conséquent, les applications qui doivent vérifier si l'expéditeur du message a demandé un rapport particulier doivent utiliser l'une des techniques décrites.

Utilisation d'opérations de bit

Si le langage de programmation prend en charge les opérations de bit, procédez comme suit:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de rapport à vérifier:

   - MQRO_COA_WITH_FULL_DATA pour le rapport COA
   - MQRO_COD_WITH_FULL_DATA pour le rapport COD
   - MQRO_EXCEPTION_WITH_FULL_DATA pour le rapport d'exception
   - MQRO_EXPIRATION_WITH_FULL_DATA pour le rapport d'expiration

   Appelez la valeur A.

   Sous z/OS, utilisez les valeurs MQRO_ * _WITH_DATA à la place des valeurs MQRO_ * _WITH_FULL_DATA.
2. Combinez la zone *Report* avec A à l'aide de l'opération AND bit à bit ; appelez le résultat B.
3. Testez B pour l'égalité avec chaque valeur possible pour ce type de rapport.

   Par exemple, si A est MQRO_EXCEPTION_WITH_FULL_DATA, testez B pour l'égalité avec chacun des éléments suivants afin de déterminer ce qui a été spécifié par l'expéditeur du message:

   - MQRO_AUCUN
   - MQRO_EXCEPTION
   - MQRO_EXCEPTION_WITH_DATA
   - MQRO_EXCEPTION_WITH_FULL_DATA

   Les tests peuvent être effectués dans l'ordre qui convient le mieux à la logique de l'application.

Utilisez une méthode similaire pour tester les options MQRO_PASS_MSG_ID ou MQRO_PASS_CORREL_ID ; sélectionnez comme valeur A la valeur appropriée de ces deux constantes, puis procédez comme indiqué ci-dessus.

Utilisation de l'arithmétique

Si le langage de programmation *ne prend pas* en charge les opérations bit, effectuez les étapes suivantes à l'aide de l'arithmétique entière:

1. Sélectionnez l'une des valeurs suivantes, en fonction du type de rapport à vérifier:

   - Rapport MQRO_COA for COA
   - Rapport MQRO_COD pour COD
   - MQRO_EXCEPTION pour le rapport d'exception
   - MQRO_EXPIRATION pour le rapport d'expiration

   Appelez la valeur A.
2. Divisez la zone *Report* par A; appelez le résultat B.
3. Divisez B par 8; appelez le résultat C.
4. Multipliez C par 8 et soustrayez de B; appelez le résultat D.
5. Multipliez D par A; appelez le résultat E.
6. Testez E pour l'égalité avec chaque valeur possible pour ce type de rapport.

   Par exemple, si A est MQRO_EXCEPTION, testez E pour l'égalité avec chacun des éléments suivants afin de déterminer ce qui a été spécifié par l'expéditeur du message:

   - MQRO_AUCUN
   - MQRO_EXCEPTION
   - MQRO_EXCEPTION_WITH_DATA
   - MQRO_EXCEPTION_WITH_FULL_DATA

Les tests peuvent être effectués dans l'ordre qui convient le mieux à la logique de l'application.

Le pseudocode suivant illustre cette technique pour les messages de rapport d'exception:

```
A = MQRO_EXCEPTION
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

Utilisez une méthode similaire pour tester les options MQRO_PASS_MSG_ID ou MQRO_PASS_CORREL_ID ; sélectionnez comme valeur A celle de ces deux constantes qui est appropriée, puis procédez comme décrit ci-dessus, en remplaçant la valeur 8 dans les étapes ci-dessus par la valeur 2.

Structure de la zone d'indicateurs de message

Ces informations décrivent la structure de la zone d'indicateurs de message.

La zone *MsgFlags* est un entier 32 bits qui est divisé en trois sous-zones distinctes. Ces sous-zones identifient:

- Indicateurs de message rejetés si le gestionnaire de files d'attente local ne les reconnaît pas
- Indicateurs de message toujours acceptés, même si le gestionnaire de files d'attente local ne les reconnaît pas
- Indicateurs de message acceptés uniquement si certaines autres conditions sont remplies

Remarque : Toutes les sous-zones de *MsgFlags* sont réservées pour être utilisées par le gestionnaire de files d'attente.

Chaque sous-champ est identifié par un masque de bit qui a 1 bit dans les positions correspondent au sous-champ, et 0 bit ailleurs. Les bits sont numérotés de sorte que le bit 0 est le bit de poids fort, et le bit 31 le bit de poids faible. Les masques suivants sont définis pour identifier les sous-zones:

MQMF_REJECT_UNSUP_MASK
Ce masque identifie les positions de bit dans la zone *MsgFlags* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local entraînent l'échec de l'appel MQPUT ou MQPUT1 avec le code achèvement MQCC_FAILED et le code anomalie MQRC_MSG_FLAGS_ERROR.

Cette sous-zone occupe les positions de bit 20 à 31.

Les indicateurs de message suivants sont inclus dans cette sous-zone:

- MQMF_LAST_MSG_IN_GROUP
- SEGMENT_DERNIER_MQMF_SEGMENT
- GROUPE MQMF_MSG_IN_GROUP
- SEGMENT_MQMF
- MQMF_SEGMENTATION_ALLOWED
- MQMF_SEGMENTATION_INHIBÉE

MQMF_ACCEPT_UNSUP_MASK
Ce masque identifie les positions de bit dans la zone *MsgFlags* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local sont néanmoins acceptés sur les appels MQPUT ou MQPUT1 . Le code achèvement est MQCC_OK.

Cette sous-zone occupe les positions de bit 0 à 11.

MQMF_ACCEPT_UNSUP_IF_XMIT_MASK
Ce masque identifie les positions de bit dans la zone *MsgFlags* où les indicateurs de message qui ne sont pas pris en charge par le gestionnaire de files d'attente local sont néanmoins acceptés sur les appels MQPUT ou MQPUT1 *à condition* que les deux conditions suivantes soient satisfaites:

- Le message est destiné à un gestionnaire de files d'attente éloignées.
- L'application ne place pas le message directement dans une file d'attente de transmission locale (c'est-à-dire que la file d'attente identifiée par les zones *ObjectQMgrName* et *ObjectName* dans le descripteur d'objet spécifié dans l'appel MQOPEN ou MQPUT1 n'est pas une file d'attente de transmission locale).

Le code achèvement MQCC_OK est renvoyé si ces conditions sont satisfaites et MQCC_FAILED avec le code anomalie MQRC_MSG_FLAGS_ERROR si ce n'est pas le cas.

Cette sous-zone occupe les positions de bit 12 à 19.

Si des indicateurs sont spécifiés dans la zone *MsgFlags* que le gestionnaire de files d'attente ne reconnaît pas, le gestionnaire de files d'attente vérifie chaque sous-zone à tour de rôle en utilisant l'opération bit à bit AND pour combiner la zone *MsgFlags* avec le masque de cette sous-zone. Si le résultat de cette opération est différent de zéro, le code achèvement et les codes raison décrits ci-dessus sont renvoyés.

Exit de conversion de données

Cette collection de rubriques décrit l'interface de l'exit de conversion de données et le traitement effectué par le gestionnaire de files d'attente lorsque la conversion de données est requise.

Pour plus d'informations sur la conversion de données, voir *Data Conversion under WebSphere MQ* à l'adresse https://www.ibm.com/support/docview.wss?uid=swg27005729.

L'exit de conversion de données est appelé dans le cadre du traitement de l'appel MQGET afin de convertir les données de message d'application à la représentation requise par l'application de réception. La conversion des données de message d'application est facultative ; elle nécessite que l'option MQGMO_CONVERT soit spécifiée sur l'appel MQGET.

Les sujets suivants sont décrits:

- Traitement effectué par le gestionnaire de files d'attente en réponse à l'option MQGMO_CONVERT ; voir «Traitement de la conversion», à la page 898.
- Conventions de traitement utilisées par le gestionnaire de files d'attente lors du traitement d'un format intégré ; ces conventions sont également recommandées pour les exits écrits par l'utilisateur. Voir «Conventions de traitement», à la page 900.
- Remarques spéciales relatives à la conversion des messages de rapport ; voir «Conversion des messages de rapport», à la page 904.
- Les paramètres transmis à l'exit de conversion de données ; voir «MQ_DATA_CONV_EXIT-Exit de conversion de données», à la page 917.
- Appel pouvant être utilisé à partir de l'exit pour convertir des données de type caractères entre différentes représentations ; voir «MQXCNVC-Conversion de caractères», à la page 911.
- Paramètre de structure de données spécifique à l'exit ; voir «MQDXP-Paramètre d'exit de conversion de données», à la page 905.

Traitement de la conversion

Ces informations décrivent le traitement effectué par le gestionnaire de files d'attente en réponse à l'option MQGMO_CONVERT.

Le gestionnaire de files d'attente effectue les actions suivantes si l'option MQGMO_CONVERT est spécifiée dans l'appel MQGET et qu'un message doit être renvoyé à l'application:

1. Si une ou plusieurs des conditions suivantes sont remplies, aucune conversion n'est nécessaire:
   - Les données de message se trouvent déjà dans le jeu de caractères et le codage requis par l'application émettant l'appel MQGET. L'application doit définir les zones *CodedCharSetId* et *Encoding* dans le paramètre *MsgDesc* de l'appel MQGET sur les valeurs requises, avant d'émettre l'appel.
   - La longueur des données de message est égale à zéro.

- La longueur du paramètre *Buffer* de l'appel MQGET est égale à zéro.

Dans ces cas, le message est renvoyé sans conversion à l'application émettant l'appel MQGET ; les valeurs *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* sont définies sur les valeurs des informations de contrôle du message et l'appel se termine avec l'une des combinaisons suivantes de code achèvement et de code anomalie:

Code de fin d'exécution	Code raison
MQCC_OK	MQRC_NONE
MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED
MQCC_WARNING	MQRC_TRUNCATED_MSG_FAILED

Les étapes suivantes sont effectuées uniquement si le jeu de caractères ou le codage des données de message diffère de la valeur correspondante dans le paramètre *MsgDesc* et que des données doivent être converties:

2. Si la zone *Format* des informations de contrôle du message a la valeur MQFMT_NONE, le message est renvoyé non converti avec le code achèvement MQCC_WARNING et le code anomalie MQRC_FORMAT_ERROR.

   Dans tous les autres cas, le traitement de la conversion se poursuit.
3. Le message est supprimé de la file d'attente et placé dans une mémoire tampon temporaire de même taille que le paramètre *Buffer* . Pour les opérations de navigation, le message est copié dans la mémoire tampon temporaire au lieu d'être supprimé de la file d'attente.
4. Si le message doit être tronqué pour tenir dans la mémoire tampon, les opérations suivantes sont effectuées:
   - Si l'option MQGMO_ACCEPT_TRUNCATED_MSG n'a *pas* été spécifiée, le message est renvoyé non converti avec le code achèvement MQCC_WARNING et le code anomalie MQRC_TRUNCATED_MSG_FAILED.
   - Si l'option MQGMO_ACCEPT_TRUNCATED_MSG *a été* spécifiée, le code achèvement est défini sur MQCC_WARNING, le code anomalie est défini sur MQRC_TRUNCATED_MSG_ACCEPTED et le traitement de la conversion se poursuit.
5. Si le message peut être hébergé dans la mémoire tampon sans troncature ou si l'option MQGMO_ACCEPT_TRUNCATED_MSG a été spécifiée, les opérations suivantes sont effectuées:
   - Si le format est un format intégré, la mémoire tampon est transmise au service de conversion de données du gestionnaire de files d'attente.
   - Si le format n'est pas un format intégré, la mémoire tampon est transmise à un exit écrit par l'utilisateur portant le même nom que le format. Si l'exit est introuvable, le message est renvoyé sans conversion, avec le code achèvement MQCC_WARNING et le code anomalie MQRC_FORMAT_ERROR.

   Si aucune erreur ne se produit, la sortie du service de conversion de données ou de l'exit écrit par l'utilisateur est le message converti, ainsi que le code achèvement et le code raison à renvoyer à l'application émettant l'appel MQGET.
6. Si la conversion aboutit, le gestionnaire de files d'attente renvoie le message converti à l'application. Dans ce cas, le code achèvement et le code anomalie renvoyés par l'appel MQGET sont l'une des combinaisons suivantes:

Code de fin d'exécution	Code raison
MQCC_OK	MQRC_NONE
MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED

Toutefois, si la conversion est effectuée par un exit écrit par l'utilisateur, d'autres codes raison peuvent être renvoyés, même lorsque la conversion aboutit.

Si la conversion échoue, le gestionnaire de files d'attente renvoie le message non converti à l'application, avec les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* définies sur les valeurs des informations de contrôle du message, et avec le code achèvement MQCC_WARNING.

Conventions de traitement

Lors de la conversion d'un format intégré, le gestionnaire de files d'attente respecte les conventions de traitement décrites.

Les exits écrits par l'utilisateur doivent également respecter ces conventions, bien que cela ne soit pas appliqué par le gestionnaire de files d'attente. Les formats intégrés convertis par le gestionnaire de files d'attente sont les suivants:

- MQFMT_ADMIN
- MQFMT_CICS (z/OS uniquement)
- MQFMT_COMMAND_1
- MQFMT_COMMAND_2
- MQFMT_DEAD_LETTER_HEADER
- MQFMT_DIST_HEADER
- MQFMT_EVENT version 1
- MQFMT_EVENT version 2
- MQFMT_IMS
- MQFMT_IMS_VAR_STRING
- MQFMT_MD_EXTENSION
- MQFMT_PCF
- MQFMT_REF_MSG_HEADER
- MQFMT_RF_HEADER
- MQFMT_RF_HEADER_2
- MQFMT_STRING
- MQFMT_TRIGGER
- MQFMT_WORK_INFO_HEADER (z/OS uniquement)
- MQFMT_XMIT_Q_HEADER

1. Si le message se développe lors de la conversion et dépasse la taille du paramètre *Buffer* , procédez comme suit:
   - Si l'option MQGMO_ACCEPT_TRUNCATED_MSG n'a *pas* été spécifiée, le message est renvoyé non converti avec le code achèvement MQCC_WARNING et le code anomalie MQRC_CONVERTED_MSG_TOO_BIG.
   - Si l'option MQGMO_ACCEPT_TRUNCATED_MSG *a été* spécifiée, le message est tronqué, le code achèvement est défini sur MQCC_WARNING, le code anomalie est défini sur MQRC_TRUNCATED_MSG_ACCEPTED et le traitement de la conversion se poursuit.
2. En cas de troncature (avant ou pendant la conversion), le nombre d'octets valides renvoyés dans le paramètre *Buffer* peut être *inférieur* à la longueur de la mémoire tampon.

   Cela peut se produire, par exemple, si un entier de 4 octets ou un caractère DBCS chevauche la fin de la mémoire tampon. L'élément incomplet des informations n'est pas converti et les octets du message renvoyé ne contiennent pas d'informations valides. Cela peut également se produire si un message tronqué avant la conversion est réduit lors de la conversion.

   Si le nombre d'octets valides renvoyés est inférieur à la longueur de la mémoire tampon, les octets inutilisés à la fin de la mémoire tampon sont définis sur null.

3. Si un tableau ou une chaîne chevauche la fin de la mémoire tampon, autant de données que possible sont converties ; seul l'élément de tableau ou le caractère DBCS qui est incomplet n'est pas converti ; les éléments de tableau ou les caractères précédents sont convertis.
4. Si une troncature est effectuée (avant ou pendant la conversion), la longueur renvoyée pour le paramètre *DataLength* est la longueur du message *non converti* avant la troncature.
5. Lorsque des chaînes sont converties entre des jeux de caractères codés sur un octet (SBCS), des jeux de caractères codés sur deux octets (DBCS) ou des jeux de caractères codés sur plusieurs octets (MBCS), les chaînes peuvent se développer ou se contracter.
   - Dans les formats PCF MQFMT_ADMIN, MQFMT_EVENT et MQFMT_PCF, les chaînes des structures MQCFST et MQCFSL se développent ou se contractent selon les besoins pour s'adapter à la chaîne après la conversion.

     Pour la structure de liste de chaînes MQCFSL, les chaînes de la liste peuvent se développer ou se contracter différemment. Dans ce cas, le gestionnaire de files d'attente remplit les chaînes plus courtes avec des blancs pour qu'elles aient la même longueur que la chaîne la plus longue après la conversion.
   - Dans le format MQFMT_REF_MSG_HEADER, les chaînes adressées par les zones *SrcEnvOffset*, *SrcNameOffset*, *DestEnvOffset*et *DestNameOffset* se développent ou se contractent selon les besoins pour s'adapter aux chaînes après la conversion.
   - Dans le format MQFMT_RF_HEADER, la zone *NameValueString* se développe ou se contracte selon les besoins pour s'adapter aux paires nom-valeur après la conversion.
   - Dans les structures avec des tailles de champ fixes, le gestionnaire de files d'attente permet aux chaînes de se développer ou de se contracter dans leurs champs fixes, à condition qu'aucune information significative ne soit perdue. A cet égard, les blancs de fin et les caractères qui suivent le premier caractère nul de la zone sont considérés comme non significatifs.
     - Si la chaîne se développe, mais que seuls des caractères non significatifs doivent être supprimés pour prendre en charge la chaîne convertie dans la zone, la conversion aboutit et l'appel se termine avec MQCC_OK et le code anomalie MQRC_NONE (en supposant qu'il n'y ait pas d'autres erreurs).
     - Si la chaîne se développe, mais que la chaîne convertie nécessite que des caractères significatifs soient supprimés pour tenir dans la zone, le message est renvoyé non converti et l'appel se termine avec MQCC_WARNING et le code anomalie MQRC_CONVERTED_STRING_TOO_BIG.

       Remarque : Le code anomalie MQRC_CONVERTED_STRING_TOO_BIG indique dans ce cas si l'option MQGMO_ACCEPT_TRUNCATED_MSG a été spécifiée ou non.
     - Si la chaîne se contracte, le gestionnaire de files d'attente complète la chaîne avec des blancs à la longueur de la zone.
6. Pour les messages comportant une ou plusieurs structures d'en-tête MQ suivies de données utilisateur, une ou plusieurs structures d'en-tête peuvent être converties, alors que le reste du message ne l'est pas. Toutefois, à deux exceptions près, les zones *CodedCharSetId* et *Encoding* de chaque structure d'en-tête indiquent toujours correctement le jeu de caractères et le codage des données qui suivent la structure d'en-tête.

   Les deux exceptions sont les structures MQCIH et MQIIH, où les valeurs des zones *CodedCharSetId* et *Encoding* de ces structures ne sont pas significatives. Pour ces structures, les données qui suivent la structure se trouvent dans le même jeu de caractères et le même codage que la structure MQCIH ou MQIIH elle-même.
7. Si les zones *CodedCharSetId* ou *Encoding* dans les informations de contrôle du message en cours d'extraction ou dans le paramètre *MsgDesc* indiquent des valeurs non définies ou non prises en charge, le gestionnaire de files d'attente peut ignorer l'erreur si la valeur non définie ou non prise en charge n'a pas besoin d'être utilisée pour convertir le message.

   Par exemple, si la zone *Encoding* du message spécifie un codage à virgule flottante non pris en charge, mais que le message contient uniquement des données entières ou des données à virgule flottante qui ne nécessitent pas de conversion (car les codages à virgule flottante source et cible sont identiques), l'erreur peut ne pas être diagnostiquée.

Si l'erreur est diagnostiquée, le message est renvoyé sans conversion, avec le code achèvement MQCC_WARNING et l'un des codes anomalie MQRC_SOURCE_ * _ERROR ou MQRC_TARGET_ * _ERROR (selon le cas) ; les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* sont définies sur les valeurs des informations de contrôle du message.

Si l'erreur n'est pas diagnostiquée et que la conversion aboutit, les valeurs renvoyées dans les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* sont celles spécifiées par l'application émettrice de l'appel MQGET.

8. Dans tous les cas, si le message est renvoyé à l'application non convertie, le code achèvement est défini sur MQCC_WARNING et les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* sont définies sur les valeurs appropriées aux données non converties. Cette opération est également effectuée pour MQFMT_NONE.

   Le paramètre *Reason* est défini sur un code qui indique la raison pour laquelle la conversion n'a pas pu être effectuée, sauf si le message a également dû être tronqué ; les codes raison liés à la troncature sont prioritaires sur les codes raison liés à la conversion. (Pour déterminer si un message tronqué a été converti, vérifiez les valeurs renvoyées dans les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* .)

   Lorsqu'une erreur est diagnostiquée, un code anomalie spécifique est renvoyé ou le code anomalie général MQRC_NOT_CONVERTIE. Le code anomalie renvoyé dépend des capacités de diagnostic du service de conversion de données sous-jacent.

9. Si le code achèvement MQCC_WARNING est renvoyé et que plusieurs codes anomalie sont pertinents, l'ordre de priorité est le suivant:

   a. Les raisons suivantes prévalent sur toutes les autres ; une seule des raisons de ce groupe peut se produire:

      - MQRC_SIGNAL_REQUEST_ACCEPTED
      - MQRC_TRUNCATED_MSG_ACCEPTED

   b. L'ordre de priorité dans les codes anomalie restants n'est pas défini.

10. A la fin de l'appel MQGET:

    - Le code anomalie suivant indique que la conversion du message a abouti:
      - MQRC_NONE
    - Les codes anomalie suivants indiquent que le message *peut* avoir été converti (consultez les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* pour le savoir):
      - MQRC_MSG_MARKED_BROWSE_CO_OP
      - MQRC_TRUNCATED_MSG_ACCEPTED
    - Tous les autres codes anomalie indiquent que le message n'a pas été converti.

    Le traitement suivant est spécifique aux formats intégrés ; il ne s'applique pas aux formats définis par l'utilisateur:

11. A l'exception des formats suivants:

    - MQFMT_ADMIN
    - MQFMT_COMMAND_1
    - MQFMT_COMMAND_2
    - MQFMT_EVENT
    - MQFMT_IMS_VAR_STRING
    - MQFMT_PCF
    - MQFMT_STRING

    aucun des formats intégrés ne peut être converti depuis ou vers des jeux de caractères qui ne comportent pas de caractères SBCS pour les caractères admis dans les noms de file d'attente. Si une tentative est effectuée pour effectuer une telle conversion, le message est renvoyé non converti,

avec le code achèvement MQCC_WARNING et le code anomalie MQRC_SOURCE_CCSID_ERROR ou MQRC_TARGET_CCSID_ERROR, selon le cas.

Le jeu de caractères Unicode UCS-2 est un exemple de jeu de caractères qui ne comporte pas de caractères SBCS pour les caractères valides dans les noms de file d'attente.

12. Si les données de message pour un format intégré sont tronquées, les zones du message qui contiennent des longueurs de chaînes, ou des nombres d'éléments ou de structures, ne sont *pas* ajustées pour refléter la longueur des données effectivement renvoyées à l'application ; les valeurs renvoyées pour ces zones dans les données de message sont les valeurs applicables au message *avant troncature*.

    Lors du traitement de messages tels qu'un message MQFMT_ADMIN tronqué, assurez-vous que l'application ne tente pas d'accéder aux données au-delà de la fin des données renvoyées.

13. Si le nom de format est MQFMT_DEAD_LETTER_HEADER, les données de message commencent par une structure MQDLH, éventuellement suivie de zéro ou plusieurs octets de données de message d'application. Le format, le jeu de caractères et le codage des données de message d'application sont définis par les zones *Format*, *CodedCharSetId*et *Encoding* de la structure MQDLH au début du message. Etant donné que la structure MQDLH et les données de message d'application peuvent avoir des jeux de caractères et des codages différents, l'un ou l'autre, ou les deux, de la structure MQDLH et des données de message d'application peuvent nécessiter une conversion.

    Le gestionnaire de files d'attente convertit d'abord la structure MQDLH, si nécessaire. Si la conversion aboutit ou si la structure MQDLH ne requiert pas de conversion, le gestionnaire de files d'attente vérifie les zones *CodedCharSetId* et *Encoding* de la structure MQDLH pour déterminer si la conversion des données de message d'application est requise. Si la conversion *est* requise, le gestionnaire de files d'attente appelle l'exit écrit par l'utilisateur avec le nom donné par la zone *Format* dans la structure MQDLH ou effectue la conversion elle-même (si *Format* est le nom d'un format intégré).

    Si l'appel MQGET renvoie le code achèvement MQCC_WARNING et que le code anomalie indique que la conversion a échoué, l'une des conditions suivantes s'applique:

    - La structure MQDLH n'a pas pu être convertie. Dans ce cas, les données de message d'application n'auront pas non plus été converties.
    - La structure MQDLH a été convertie, mais les données de message d'application ne l'ont pas été.

    L'application peut examiner les valeurs renvoyées dans les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* , ainsi que celles de la structure MQDLH, afin de déterminer laquelle des valeurs ci-dessus s'applique.

14. Si le nom de format est MQFMT_XMIT_Q_HEADER, les données de message commencent par une structure MQXQH, éventuellement suivie de zéro ou plusieurs octets de données supplémentaires. Ces données supplémentaires sont généralement les données de message d'application (qui peuvent être de longueur nulle), mais il peut également y avoir une ou plusieurs autres structures d'en-tête MQ , au début des données supplémentaires.

    La structure MQXQH doit être dans le jeu de caractères et le codage du gestionnaire de files d'attente. Le format, le jeu de caractères et le codage des données qui suivent la structure MQXQH sont fournis par les zones *Format*, *CodedCharSetId*et *Encoding* de la structure MQMD contenue *dans* le MQXQH. Pour chaque structure d'en-tête MQ suivante présente, les zones *Format*, *CodedCharSetId*et *Encoding* de la structure décrivent les données qui suivent cette structure ; ces données sont soit une autre structure d'en-tête MQ , soit les données de message d'application.

    Si l'option MQGMO_CONVERT est spécifiée pour un message MQFMT_XMIT_Q_HEADER, les données de message d'application et certaines des structures d'en-tête MQ sont converties, *mais les données de la structure MQXQH ne sont pas*. En cas de retour de l'appel MQGET, par conséquent:

    - Les valeurs des zones *Format*, *CodedCharSetId*et *Encoding* du paramètre *MsgDesc* décrivent les données de la structure MQXQH et *ne décrivent pas* les données de message d'application ; par conséquent, les valeurs ne sont *pas* identiques à celles spécifiées par l'application qui a émis l'appel MQGET.

Cela a pour effet qu'une application qui extrait à plusieurs reprises des messages d'une file d'attente de transmission avec l'option MQGMO_CONVERT spécifiée doit réinitialiser les zones *CodedCharSetId* et *Encoding* du paramètre *MsgDesc* sur les valeurs requises pour les données de message d'application, avant chaque appel MQGET.

- Les valeurs des zones *Format*, *CodedCharSetId*et *Encoding* de la dernière structure d'en-tête MQ présente décrivent les données du message d'application. Si aucune autre structure d'en-tête MQ n'est présente, les données de message d'application sont décrites par ces zones dans la structure MQMD au sein de la structure MQXQH. Si la conversion aboutit, les valeurs seront identiques à celles spécifiées dans le paramètre *MsgDesc* par l'application qui a émis l'appel MQGET.

Si le message est un message de liste de distribution, la structure MQXQH est suivie d'une structure MQDH (plus ses tableaux d'enregistrements MQOR et MQPMR), qui à son tour peut être suivie de zéro ou plusieurs autres structures d'en-tête MQ et de zéro ou plusieurs octets de données de message d'application. A l'instar de la structure MQXQH, la structure MQDH doit être dans le jeu de caractères et le codage du gestionnaire de files d'attente, et elle n'est pas convertie sur l'appel MQGET, même si l'option MQGMO_CONVERT est spécifiée.

Le traitement des structures MQXQH et MQDH décrites ci-dessus est principalement destiné à être utilisé par les agents de canal de message lorsqu'ils extraient des messages des files d'attente de transmission.

Conversion des messages de rapport

En général, un message de rapport peut contenir des quantités variables de données de message d'application, en fonction des options de rapport spécifiées par l'expéditeur du message d'origine. Toutefois, un rapport d'activité peut contenir des données, mais sans l'option de rapport mentionnant * _WITH_DATA dans la constante.

En particulier, un message de rapport peut contenir:

1. Aucune donnée de message d'application
2. Certaines des données de message d'application du message d'origine

   Cela se produit lorsque l'expéditeur du message d'origine spécifie MQRO_ * _WITH_DATA et que le message a une longueur supérieure à 100 octets.
3. Toutes les données de message d'application du message d'origine

   Cela se produit lorsque l'expéditeur du message d'origine spécifie MQRO_ * _WITH_FULL_DATA ou MQRO_ * _WITH_DATA et que le message est de 100 octets ou moins.

Lorsque le gestionnaire de files d'attente ou l'agent MCA génère un message de rapport, il copie le nom de format du message d'origine dans la zone *Format* des informations de contrôle du message de rapport. Le nom de format dans le message de rapport peut donc impliquer une longueur de données différente de la longueur réellement présente dans le message de rapport (cas 1 et 2 ci-dessus).

Si l'option MQGMO_CONVERT est spécifiée lors de l'extraction du message de rapport:

- Pour le cas 1 ci-dessus, l'exit de conversion de données n'est pas appelé (car le message de rapport ne contient pas de données).
- Pour le cas 3 ci-dessus, le nom de format implique correctement la longueur des données de message.
- Mais pour le cas 2 ci-dessus, l'exit de conversion de données est appelé pour convertir un message *plus court* que la longueur impliquée par le nom de format.

  En outre, le code anomalie transmis à l'exit est généralement MQRC_NONE (c'est-à-dire que le code anomalie n'indique pas que le message a été tronqué). Cela se produit car les données de message ont été tronquées par l' *émetteur* du message de rapport et non par le gestionnaire de files d'attente du récepteur en réponse à l'appel MQGET.

En raison de ces possibilités, l'exit de conversion de données ne doit *pas* utiliser le nom de format pour déduire la longueur des données qui lui sont transmises ; à la place, l'exit doit vérifier la longueur des données fournies et être prêt à convertir *moins* de données que la longueur impliquée par le nom de

format. Si les données peuvent être converties correctement, le code achèvement MQCC_OK et le code anomalie MQRC_NONE doivent être renvoyés par l'exit. La longueur des données de message à convertir est transmise à l'exit en tant que paramètre *InBufferLength* .

Interface de programmation sensible au produit

MQDXP-Paramètre d'exit de conversion de données

La structure MQDXP est un paramètre que le gestionnaire de files d'attente transmet à l'exit de conversion de données lorsque l'exit est appelé pour convertir les données de message dans le cadre du traitement de l'appel MQGET. Pour plus de détails sur l'exit de conversion de données, voir la description de l'appel MQ_DATA_CONV_EXIT.

Les données de type caractère dans MQDXP se trouvent dans le jeu de caractères du gestionnaire de files d'attente local ; elles sont fournies par l'attribut de gestionnaire de files d'attente *CodedCharSetId* . Les données numériques dans MQDXP sont dans le codage de la machine native ; elles sont fournies par MQENC_NATIVE.

Seules les zones *DataLength*, *CompCode*, *Reason*et *ExitResponse* de MQDXP peuvent être modifiées par l'exit ; les modifications apportées aux autres zones sont ignorées. Toutefois, la zone *DataLength ne peut pas* être modifiée si le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique.

Lorsque le contrôle revient au gestionnaire de files d'attente à partir de l'exit, le gestionnaire de files d'attente vérifie les valeurs renvoyées dans MQDXP. Si les valeurs renvoyées ne sont pas valides, le gestionnaire de files d'attente poursuit le traitement comme si l'exit avait renvoyé MQXDR_CONVERSION_FAILED dans *ExitResponse*; cependant, le gestionnaire de files d'attente ignore les valeurs des zones *CompCode* et *Reason* renvoyées par l'exit dans ce cas, et utilise à la place les valeurs que ces zones avaient dans *input* pour l'exit. Les valeurs suivantes dans MQDXP entraînent ce traitement:

- Zone *ExitResponse* non MQXDR_OK et non MQXDR_CONVERSION_FAILED
- Zone *CompCode* non MQCC_OK et non MQCC_WARNING
- Zone *DataLength* inférieure à zéro ou zone *DataLength* modifiée lorsque le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique.

Le tableau suivant récapitule les zones de la structure.

Tableau 579. Zones dans MQDXP

Zone	Description	Topic
StrucId	Identificateur de structure	StrucId
Version	Numéro de version de structure	Version
AppOptions	Options d'application	AppOptions
Encoding	Codage numérique requis par l'application	Codage
CodedCharSetId	Jeu de caractères requis par l'application	CodedCharSetId
DataLength	Longueur en octets des données de message	DataLength
CompCode	Code de fin d'exécution	CompCode
Reason	Qualification du code anomalie *CompCode*	Motif
ExitResponse	Réponse de l'exit	ExitResponse

Tableau 579. Zones dans MQDXP (suite)		
Zone	**Description**	**Topic**
Hconn	Descripteur de connexion	Nom de connexion
pEntryPoints	Adresse de la structure MQIEP	pEntryPoints

Zones

La structure MQDXP contient les zones suivantes ; elles sont décrites par ordre alphabétique.

AppOptions
Type: MQLONG

Il s'agit d'une copie de la zone *Options* de la structure MQGMO spécifiée par l'application émettant l'appel MQGET. L'exit peut avoir besoin de les examiner pour déterminer si l'option MQGMO_ACCEPT_TRUNCATED_MSG a été spécifiée.

Il s'agit d'une zone d'entrée de l'exit.

CodedCharSetId
Type: MQLONG

Il s'agit de l'identificateur de jeu de caractères codés du jeu de caractères requis par l'application émettant l'appel MQGET ; voir la zone *CodedCharSetId* dans la structure MQMD pour plus de détails. Si l'application spécifie la valeur spéciale MQCCSI_Q_MGR dans l'appel MQGET, le gestionnaire de files d'attente remplace cette valeur par l'identificateur de jeu de caractères réel du jeu de caractères utilisé par le gestionnaire de files d'attente, avant d'appeler l'exit.

Si la conversion aboutit, l'exit doit le copier dans la zone *CodedCharSetId* du descripteur de message.

Il s'agit d'une zone d'entrée de l'exit.

CompCode
Type: MQLONG

Lorsque l'exit est appelé, il contient le code achèvement renvoyé à l'application qui a émis l'appel MQGET, si l'exit ne fait rien. Il s'agit toujours de MQCC_WARNING, car soit le message a été tronqué, soit le message doit être converti, ce qui n'a pas encore été fait.

Dans la sortie de l'exit, cette zone contient le code achèvement à renvoyer à l'application dans le paramètre *CompCode* de l'appel MQGET ; seuls MQCC_OK et MQCC_WARNING sont valides. Consultez la description de la zone *Reason* pour obtenir des suggestions sur la manière dont l'exit peut définir cette zone sur la sortie.

Il s'agit d'une zone d'entrée-sortie pour l'exit.

DataLength
Type: MQLONG

Lorsque l'exit est appelé, cette zone contient la longueur d'origine des données de message d'application. Si le message a été tronqué pour tenir dans la mémoire tampon fournie par l'application, la taille du message fourni à l'exit est *inférieure* à la valeur de *DataLength*. La taille du message fourni à l'exit est toujours indiquée par le paramètre *InBufferLength* de l'exit, quelle que soit la troncature qui s'est produite.

La troncature est indiquée par la zone *Reason* dont la valeur est MQRC_TRUNCATED_MSG_ACCEPTED en entrée de l'exit.

La plupart des conversions n'ont pas besoin de modifier cette longueur, mais un exit peut le faire si nécessaire ; la valeur définie par l'exit est renvoyée à l'application dans le paramètre *DataLength* de l'appel MQGET. Toutefois, cette longueur *ne peut pas* être modifiée si le message en cours de conversion est un segment qui ne contient qu'une partie d'un message logique. En effet, si

vous modifiez la longueur, les décalages des segments ultérieurs dans le message logique seront incorrects.

Notez que si l'exit souhaite modifier la longueur des données, sachez que le gestionnaire de files d'attente a déjà décidé si les données de message s'intègrent dans la mémoire tampon de l'application, en fonction de la longueur des données *non converties* . Cette décision détermine si le message est supprimé de la file d'attente (ou si le curseur de navigation a été déplacé pour une demande de navigation) et s'il n'est pas affecté par une modification de la longueur des données provoquée par la conversion. Pour cette raison, il est recommandé que les exits de conversion n'entraînent pas de modification de la longueur des données de message d'application.

Si la conversion de caractères implique un changement de longueur, une chaîne peut être convertie en une autre chaîne de même longueur en octets, en tronquant les blancs de fin ou en ajoutant des blancs si nécessaire.

L'exit n'est pas appelé si le message ne contient pas de données de message d'application ; par conséquent, *DataLength* est toujours supérieur à zéro.

Il s'agit d'une zone d'entrée-sortie pour l'exit.

Encoding
Type: MQLONG

Codage numérique requis par l'application.

Il s'agit du codage numérique requis par l'application qui émet l'appel MQGET ; voir la zone *Encoding* dans la structure MQMD pour plus de détails.

Si la conversion aboutit, l'exit le copie dans la zone *Encoding* du descripteur de message.

Il s'agit d'une zone d'entrée de l'exit.

ExitOptions
Type: MQLONG

Il s'agit d'une zone réservée ; sa valeur est `0`.

ExitResponse
Type: MQLONG

Réponse de l'exit.Ce paramètre est défini par l'exit pour indiquer la réussite ou non de la conversion. Il doit s'agir de l'une des valeurs suivantes:

MQXDR_OK
La conversion a abouti.

Si l'exit spécifie cette valeur, le gestionnaire de files d'attente renvoie la valeur suivante à l'application qui a émis l'appel MQGET:

- Valeur de la zone *CompCode* dans la sortie de l'exit
- Valeur de la zone *Reason* dans la sortie de l'exit
- Valeur de la zone *DataLength* dans la sortie de l'exit
- Contenu de la mémoire tampon de sortie de l'exit *OutBuffer*. Le nombre d'octets renvoyés correspond au paramètre *OutBufferLength* de l'exit le moins élevé et à la valeur de la zone *DataLength* dans la sortie de l'exit.

  Si les zones *Encoding* et *CodedCharSetId* du paramètre de descripteur de message de l'exit sont *les deux* inchangées, le gestionnaire de files d'attente renvoie:
- Valeur des zones *Encoding* et *CodedCharSetId* dans la structure MQDXP sur l' *entrée* de l'exit.

  Si l'une ou les deux zones *Encoding* et *CodedCharSetId* du paramètre de descripteur de message de l'exit ont été modifiées, le gestionnaire de files d'attente renvoie:
- Valeur des zones *Encoding* et *CodedCharSetId* dans le paramètre de descripteur de message de l'exit sur la sortie de l'exit

Echec de MQXDR_CONVERSION_FAILED
La conversion a échoué.

Si l'exit spécifie cette valeur, le gestionnaire de files d'attente renvoie la valeur suivante à l'application qui a émis l'appel MQGET:

- Valeur de la zone *CompCode* dans la sortie de l'exit
- Valeur de la zone *Reason* dans la sortie de l'exit
- Valeur de la zone *DataLength* dans l' *entrée* de l'exit
- Contenu de la mémoire tampon d'entrée de l'exit *InBuffer*. Le nombre d'octets renvoyés est indiqué par le paramètre *InBufferLength*

Si l'exit a modifié *InBuffer*, les résultats ne sont pas définis.

ExitResponse est une zone de sortie de l'exit.

Hconn
Type: MQHCONN

Il s'agit d'un descripteur de connexion qui peut être utilisé sur l'appel MQXCNVC. Ce descripteur n'est pas nécessairement le même que celui spécifié par l'application qui a émis l'appel MQGET.

pEntryPoints
Type: PMQIEP

Adresse d'une structure MQIEP via laquelle les appels MQI et DCI peuvent être effectués.

Reason
Type: MQLONG

Code anomalie qualifiant *CompCode*.

Lorsque l'exit est appelé, il contient le code anomalie renvoyé à l'application qui a émis l'appel MQGET, si l'exit choisit de ne rien faire. Parmi les valeurs possibles figurent MQRC_TRUNCATED_MSG_ACCEPTED, qui indique que le message a été tronqué pour tenir dans la mémoire tampon fournie par l'application, et MQRC_NOT_CONVERTIS, qui indique que le message requiert une conversion mais que cette opération n'a pas encore été effectuée.

Dans la sortie de l'exit, cette zone contient la raison du renvoi à l'application dans le paramètre *Reason* de l'appel MQGET ; les éléments suivants sont recommandés:

- Si *Reason* a la valeur MQRC_TRUNCATED_MSG_ACCEPTED en entrée de l'exit, les zones *Reason* et *CompCode* ne doivent pas être modifiées, que la conversion aboutisse ou échoue.

  (Si la zone *CompCode* n'est pas MQCC_OK, l'application qui extrait le message peut identifier un échec de conversion en comparant les valeurs *Encoding* et *CodedCharSetId* renvoyées dans le descripteur de message avec les valeurs demandées ; en revanche, l'application ne peut pas distinguer un message tronqué d'un message correspondant à la mémoire tampon. Pour cette raison, MQRC_TRUNCATED_MSG_ACCEPTED doit être renvoyé de préférence à l'une des raisons qui indiquent l'échec de la conversion.)
- Si *Reason* avait une autre valeur en entrée de l'exit:
  - Si la conversion aboutit, *CompCode* doit être défini sur MQCC_OK et *Reason* sur MQRC_NONE.
  - Si la conversion échoue, ou si le message se développe et doit être tronqué pour tenir dans la mémoire tampon, *CompCode* doit être défini sur MQCC_WARNING (ou laissé inchangé) et *Reason* sur l'une des valeurs répertoriées, pour indiquer la nature de l'échec.

    Notez que si le message après la conversion est trop volumineux pour la mémoire tampon, il doit être tronqué uniquement si l'application qui a émis l'appel MQGET a spécifié l'option MQGMO_ACCEPT_TRUNCATED_MSG:

    - S'il a spécifié cette option, la raison MQRC_TRUNCATED_MSG_ACCEPTED est renvoyée.
    - S'il n'a pas spécifié cette option, le message est renvoyé non converti, avec le code anomalie MQRC_CONVERTED_MSG_TOO_BIG.

Les codes anomalie répertoriés sont recommandés par l'exit pour indiquer la raison pour laquelle la conversion a échoué, mais l'exit peut renvoyer d'autres valeurs à partir de l'ensemble de codes MQRC_ * si cela est jugé approprié. En outre, la plage de valeurs MQRC_APPL_FIRST à MQRC_APPL_LAST est allouée pour être utilisée par l'exit pour indiquer les conditions que l'exit souhaite communiquer à l'application émettrice de l'appel MQGET.

Remarque : Si le message ne peut pas être converti correctement, l'exit *doit* renvoyer MQXDR_CONVERSION_FAILED dans la zone *ExitResponse* afin que le gestionnaire de files d'attente renvoie le message non converti. Cela est vrai quel que soit le code anomalie renvoyé dans la zone *Reason* .

MQRC_APPL_FIRST
(900, X'384') Valeur la plus faible pour le code anomalie défini par l'application.

MQRC_APPL_LAST
(999, X'3E7') Valeur la plus élevée pour le code anomalie défini par l'application.

MQRC_CONVERTED_MSG_TOO_BIG
(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

MQRC_NOT_CONVERTED
(2119, X'847') Données de message non converties.

MQRC_SOURCE_CCSID_ERROR
(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

MQRC_SOURCE_DECIMAL_ENC_ERROR
(2113, X'841') Codage décimal condensé dans le message non reconnu.

MQRC_SOURCE_FLOAT_ENC_ERROR
(2114, X'842') Codage à virgule flottante dans le message non reconnu.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840') Codage d'entier source non reconnu.

MQRC_TARGET_CCSID_ERROR
(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

MQRC_TARGET_DECIMAL_ENC_ERROR
(2117, X'845') Codage décimal condensé spécifié par le récepteur non reconnu.

MQRC_TARGET_FLOAT_ENC_ERROR
(2118, X'846') Codage à virgule flottante spécifié par le récepteur non reconnu.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844') Codage d'entier cible non reconnu.

MQRC_TRUNCATED_MSG_ACCEPTED
(2079, X'81F') Message tronqué renvoyé (traitement terminé).

Il s'agit d'une zone d'entrée-sortie pour l'exit.

StrucId
Type: MQCHAR4

Identificateur de structure.La valeur doit être:

ID_STRUC_MQDXP
Identificateur de la structure des paramètres d'exit de conversion de données.

Pour le langage de programmation C, la constante MQDXP_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQDXP_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

Il s'agit d'une zone d'entrée de l'exit.

Version
Type: MQLONG

Numéro de version de la structure.La valeur doit être:

MQDXP_VERSION_1
Numéro de version de la structure des paramètres d'exit de conversion de données.

La constante suivante indique le numéro de version de la version en cours:

MQDXP_CURRENT_VERSION
Version actuelle de la structure des paramètres d'exit de conversion de données.

Remarque : Lorsqu'une nouvelle version de cette structure est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit donc vérifier que la zone *Version* est égale ou supérieure à la version la plus basse qui contient les zones que l'exit doit utiliser.

Il s'agit d'une zone d'entrée de l'exit.

Déclaration C

```
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 */
};
```

Déclaration COBOL (IBM i uniquement)

```
**   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.
```

Déclaration assembleur System/390

```
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-Conversion de caractères

L'appel MQXCNVC convertit les caractères d'un jeu de caractères à un autre à l'aide du langage de programmation C.

Cet appel fait partie de l'interface de conversion de données (DCI) WebSphere MQ , qui est l'une des interfaces de l'infrastructure WebSphere MQ .

Remarque: L'appel peut être utilisé à partir des environnements d'exit d'application et de conversion de données.

Syntaxe

MQXCNVC (*Hconn*, *Options*, *SourceCCSID*, *SourceLength*, *SourceBuffer*, *TargetCCSID*, *TargetLength*, *TargetBuffer*, *DataLength*, *CompCode*, *Cause*)

Paramètres

Hconn
Type : MQHCONN - entrée

Ce descripteur représente la connexion au gestionnaire de files d'attente.

Dans un exit de conversion de données, *Hconn* est normalement le descripteur qui est transmis à l'exit de conversion de données dans la zone *Hconn* de la structure MQDXP ; ce descripteur n'est pas nécessairement le descripteur spécifié par l'application qui a émis l'appel MQGET.

Sous IBM i, la valeur spéciale suivante peut être spécifiée pour *Hconn*:

MQHC_DEF_HCONN
Descripteur de connexion par défaut.

Si vous exécutez une application CICS TS 3.2 ou supérieure, assurez-vous que le programme d'exit de conversion de caractères, qui appelle l'appel MQXCNVC, est défini en tant qu'OPENAPI. Cette définition empêche l'erreur 2018 MQRC_HCONN_ERROR causée par une connexion incorrecte et permet à MQGET de s'exécuter.

OPTIONS
Type : MQLONG - entrée

Options qui contrôlent l'action de MQXCNVC.

Aucune ou plusieurs des options décrites peuvent être spécifiées. Si plusieurs valeurs sont requises, les valeurs peuvent être les suivantes:

- ajoutées les unes aux autres (n'ajoutez pas la même constante plusieurs fois), ou
- Combiné à l'aide de l'opération OR bit à bit (si le langage de programmation prend en charge les opérations bit)

Option de conversion par défaut: L'option suivante contrôle l'utilisation de la conversion de caractères par défaut:

MQDCC_CONVERSION_PAR_DÉFAUT_PAR_DÉFAUT
Conversion par défaut.

Cette option indique que la conversion de caractères par défaut peut être utilisée si l'un des jeux de caractères indiqués dans l'appel, ou les deux, ne sont pas pris en charge. Cela permet au gestionnaire de files d'attente d'utiliser un jeu de caractères par défaut spécifié par l'installation qui se rapproche du jeu de caractères spécifié lors de la conversion de la chaîne.

Remarque : Le résultat de l'utilisation d'un jeu de caractères approximatif pour convertir la chaîne est que certains caractères peuvent être convertis de manière incorrecte. Cela peut être évité en utilisant dans la chaîne uniquement des caractères qui sont communs au jeu de caractères spécifié et au jeu de caractères par défaut.

Les jeux de caractères par défaut sont définis par une option de configuration lorsque le gestionnaire de files d'attente est installé ou redémarré.

Si MQDCC_DEFAULT_CONVERSION n'est pas spécifié, le gestionnaire de files d'attente utilise uniquement les jeux de caractères spécifiés pour convertir la chaîne, et l'appel échoue si l'un ou les deux jeux de caractères ne sont pas pris en charge.

Cette option est prise en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows.

Option de remplissage: L'option suivante permet au gestionnaire de files d'attente de remplir la chaîne convertie avec des blancs ou de supprimer les caractères de fin non significatifs afin que la chaîne convertie corresponde à la mémoire tampon cible:

MQDCC_FILL_TARGET_BUFFER
Remplissage de la mémoire tampon cible.

Cette option demande que la conversion soit effectuée de telle sorte que la mémoire tampon cible soit complètement remplie:

- Si la chaîne se contracte lors de sa conversion, des blancs de fin sont ajoutés afin de remplir la mémoire tampon cible.
- Si la chaîne se développe lors de sa conversion, les caractères de fin qui ne sont pas significatifs sont supprimés pour que la chaîne convertie corresponde à la mémoire tampon cible. Si cette opération peut aboutir, l'appel se termine avec MQCC_OK et le code anomalie MQRC_NONE.

  S'il y a trop peu de caractères de fin insignifiants, la chaîne la plus adaptée est placée dans la mémoire tampon cible et l'appel se termine avec MQCC_WARNING et le code anomalie MQRC_CONVERTED_MSG_TOO_BIG.

  Les caractères non significatifs sont les suivants:

  – Blancs de fin

  – Caractères suivant le premier caractère null dans la chaîne (à l'exclusion du premier caractère null lui-même)
- Si la chaîne, *TargetCCSID*et *TargetLength* sont telles que la mémoire tampon cible ne peut pas être complètement définie avec des caractères valides, l'appel échoue avec MQCC_FAILED et le code anomalie MQRC_TARGET_LENGTH_ERROR. Cela peut se produire lorsque *TargetCCSID* est un jeu de caractères DBCS pur (tel que UCS-2), mais que *TargetLength* indique une longueur qui est un nombre impair d'octets.
- *TargetLength* peut être inférieur ou supérieur à *SourceLength*. Au retour de MQXCNVC, *DataLength* a la même valeur que *TargetLength*.

Si cette option n'est pas spécifiée:

- La chaîne est autorisée à se contracter ou à se développer dans la mémoire tampon cible selon les besoins. Les caractères de fin non significatifs ne sont pas ajoutés ou supprimés.

  Si la chaîne convertie correspond à la mémoire tampon cible, l'appel se termine avec MQCC_OK et le code anomalie MQRC_NONE.

  Si la chaîne convertie est trop grande pour la mémoire tampon cible, une partie de la chaîne est placée dans la mémoire tampon cible et l'appel se termine avec MQCC_WARNING et le

code anomalie MQRC_CONVERTED_MSG_TOO_BIG. Notez qu'un nombre d'octets inférieur à *TargetLength* peut être renvoyé dans ce cas.

- *TargetLength* peut être inférieur ou supérieur à *SourceLength*. Au retour de MQXCNVC, *DataLength* est inférieur ou égal à *TargetLength*.

Cette option est prise en charge dans les environnements suivants: AIX, HP-UX, IBM i, Solaris, Linux, Windows.

Options de codage: Les options décrites peuvent être utilisées pour spécifier les codages d'entier des chaînes source et cible. Le codage approprié est utilisé *uniquement* lorsque l'identificateur de jeu de caractères correspondant indique que la représentation du jeu de caractères dans la mémoire principale dépend du codage utilisé pour les entiers binaires. Cela n'affecte que certains jeux de caractères multi-octets (par exemple, les jeux de caractères UCS-2).

Le codage est ignoré si le jeu de caractères est un jeu de caractères mono-octet (SBCS) ou un jeu de caractères multi-octets avec une représentation dans la mémoire principale qui ne dépend pas du codage d'entier.

Une seule des valeurs MQDCC_SOURCE_ * doit être spécifiée, combinée à l'une des valeurs MQDCC_TARGET_ *:

MQDCC_SOURCE_ENC_NATIVE
Le codage source est la valeur par défaut pour l'environnement et le langage de programmation.

MQDCC_SOURCE_ENC_NORMAL
Le codage source est normal.

MQDCC_SOURCE_ENC_INVERSE
Le codage source est inversé.

MQDCC_SOURCE_ENC_UNDEFINED
Le codage source n'est pas défini.

MQDCC_TARGET_ENC_NATIVE
Le codage cible est la valeur par défaut pour l'environnement et le langage de programmation.

MQDCC_CIBLE_ENC_NORMAL
Le codage cible est normal.

MQDCC_CIBLE_INVERSÉ
Le codage cible est inversé.

MQDCC_TARGET_ENC_UNDEFINED
Le codage cible n'est pas défini.

Les valeurs de codage définies précédemment peuvent être ajoutées directement à la zone *Options* . Toutefois, si le codage source ou cible est obtenu à partir de la zone *Encoding* dans le MQMD ou d'une autre structure, le traitement suivant doit être effectué:

1. Le codage d'entier doit être extrait de la zone *Encoding* en éliminant les codages à virgule flottante et décimal condensé. Pour plus de détails sur cette opération, voir «Analyse des codages», à la page 893 .
2. Le codage d'entier résultant de l'étape 1 doit être multiplié par le facteur approprié avant d'être ajouté à la zone *Options* . Ces facteurs sont les suivants:
   - MQDCC_SOURCE_ENC_FACTOR pour le codage source
   - MQDCC_TARGET_ENC_FACTOR pour le codage cible

L'exemple de code suivant illustre comment cela peut être codé dans le langage de programmation C:

```
Options = (MsgDesc.Encoding & MQENC_INTEGER_MASK)
           * MQDCC_SOURCE_ENC_FACTOR
        + (DataConvExitParms.Encoding & MQENC_INTEGER_MASK)
           * MQDCC_TARGET_ENC_FACTOR;
```

S'il n'est pas spécifié, les options de codage sont définies par défaut comme non définies (MQDCC_ * _ENC_UNDEFINED). Dans la plupart des cas, cela n'affecte pas la réussite de l'appel MQXCNVC. Toutefois, si le jeu de caractères correspondant est un jeu de caractères multi-octets dont la représentation dépend du codage (par exemple, un jeu de caractères UCS-2), l'appel échoue avec le code anomalie MQRC_SOURCE_INTEGER_ENC_ERROR ou MQRC_TARGET_INTEGER_ENC_ERROR, selon le cas.

Les options de codage sont prises en charge dans les environnements suivants: AIX, HP-UX, z/OS, IBM i, Solaris, Linux, Windows.

Option par défaut: si aucune des options décrites précédemment n'est spécifiée, l'option suivante peut être utilisée:

MQDCC_AUCUN
Aucune option spécifiée.

MQDCC_NONE est défini pour faciliter la documentation du programme. Il n'est pas prévu que cette option soit utilisée avec une autre option, mais comme sa valeur est zéro, une telle utilisation ne peut pas être détectée.

SourceCCSID
Type : MQLONG - entrée

Il s'agit de l'identificateur de jeu de caractères codés de la chaîne d'entrée dans *SourceBuffer*.

SourceLength
Type : MQLONG - entrée

Il s'agit de la longueur en octets de la chaîne d'entrée dans *SourceBuffer*; elle doit être supérieure ou égale à zéro.

SourceBuffer
Type: MQCHARxSourceLongueur-entrée

Il s'agit de la mémoire tampon contenant la chaîne à convertir d'un jeu de caractères à un autre.

TargetCCSID
Type : MQLONG - entrée

Il s'agit de l'identificateur de jeu de caractères codés du jeu de caractères dans lequel *SourceBuffer* doit être converti.

TargetLength
Type : MQLONG - entrée

Il s'agit de la longueur en octets de la mémoire tampon de sortie *TargetBuffer*; elle doit être supérieure ou égale à zéro. Il peut être inférieur ou supérieur à *SourceLength*.

TargetBuffer
Type: MQCHARxTargetLongueur-sortie

Il s'agit de la chaîne après sa conversion au jeu de caractères défini par *TargetCCSID*. La chaîne convertie peut être plus courte ou plus longue que la chaîne non convertie. Le paramètre *DataLength* indique le nombre d'octets valides renvoyés.

DataLength
Type : MQLONG - sortie

Il s'agit de la longueur de la chaîne renvoyée dans la mémoire tampon de sortie *TargetBuffer*. La chaîne convertie peut être plus courte ou plus longue que la chaîne non convertie.

CompCode
Type : MQLONG - sortie

Il peut s'agir de :

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* a pour valeur MQCC_WARNING :

MQRC_CONVERTED_MSG_TOO_BIG
(2120, X'848') Données converties trop volumineuses pour la mémoire tampon.

Si *CompCode* a pour valeur MQCC_FAILED :

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Paramètre de longueur des données non valide.

MQRC_DBCS_ERROR
(2150, X'866') Chaîne DBCS non valide.

MQRC_HCONN_ERROR
(2018, X'7E2') Descripteur de connexion non valide.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Ressources système disponibles insuffisantes.

MQRC_SOURCE_BUFFER_ERROR
(2145, X'861') Paramètre de tampon source non valide.

MQRC_SOURCE_CCSID_ERROR
(2111, X'83F') Identificateur de jeu de caractères codés source non valide.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840') Codage d'entier source non reconnu.

MQRC_SOURCE_LENGTH_ERROR
(2143, X'85F') Paramètre de longueur source non valide.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Mémoire disponible insuffisante.

MQRC_TARGET_BUFFER_ERROR
(2146, X'862') Paramètre de tampon cible non valide.

MQRC_TARGET_CCSID_ERROR
(2115, X'843') Identificateur de jeu de caractères codés cible non valide.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844') Codage d'entier cible non reconnu.

MQRC_TARGET_LENGTH_ERROR
(2144, X'860') Paramètre de longueur cible incorrect.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une erreur inattendue s'est produite.

Pour plus de détails sur ces codes, voir Codes anomalie.

Appel C

```
MQXCNVC (Hconn, Options, SourceCCSID, SourceLength, SourceBuffer,
        TargetCCSID, TargetLength, TargetBuffer, &DataLength,
        &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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 */
```

Déclaration COBOL (IBM i uniquement)

```
     CALL 'MQXCNVC' USING HCONN, OPTIONS, SOURCECCSID, SOURCELENGTH,
                          SOURCEBUFFER, TARGETCCSID, TARGETLENGTH,
                          TARGETBUFFER, DATALENGTH, COMPCODE, REASON.
```

Déclarez les paramètres comme suit :

```
**   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.
```

Déclaration d'assembleur S/390

```
         CALL MQXCNVC,(HCONN,OPTIONS,SOURCECCSID,SOURCELENGTH,         X
               SOURCEBUFFER,TARGETCCSID,TARGETLENGTH,TARGETBUFFER,     X
               DATALENGTH,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
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-Exit de conversion de données

L'appel MQ_DATA_CONV_EXIT décrit les paramètres transmis à l'exit de conversion de données.

Aucun point d'entrée appelé MQ_DATA_CONV_EXIT n'est fourni par le gestionnaire de files d'attente (voir la remarque d'utilisation 11).

Cette définition fait partie de l'interface de conversion de données (DCI) WebSphere MQ , qui est l'une des interfaces de l'infrastructure WebSphere MQ .

Syntaxe

MQ_DATA_CONV_EXIT (*DataConvExitParms*, *MsgDesc*, *InBufferLength*, *InBuffer*, *OutBufferLength*, *OutBuffer*)

Paramètres

DataConvExitParms
Type: MQDXP-entrée / sortie

Cette structure contient des informations relatives à l'appel de l'exit. L'exit définit des informations dans cette structure pour indiquer le résultat de la conversion. Pour plus de détails sur les zones de cette structure, voir «MQDXP-Paramètre d'exit de conversion de données», à la page 905 .

MsgDesc
Type : MQMD - entrée/sortie

En entrée de l'exit, il s'agit du descripteur de message associé aux données de message transmises à l'exit dans le paramètre *InBuffer* .

Remarque : Le paramètre *MsgDesc* transmis à l'exit est toujours la version la plus récente de MQMD prise en charge par le gestionnaire de files d'attente qui appelle l'exit. Si l'exit est destiné à être portable entre différents environnements, l'exit vérifie la zone *Version* dans *MsgDesc* pour vérifier que les zones auxquelles l'exit doit accéder sont présentes dans la structure.

Dans les environnements suivants, l'exit est transmis à un MQMD version-2 : AIX, HP-UX, IBM i, Solaris, Linux, Windows. Dans tous les autres environnements qui prennent en charge l'exit de conversion de données, l'exit est transmis à un MQMD version-1 .

En sortie, l'exit remplace les zones *Encoding* et *CodedCharSetId* par les valeurs demandées par l'application, si la conversion a abouti ; ces modifications sont répercutées sur l'application. Toutes les autres modifications apportées par l'exit à la structure sont ignorées ; elles ne sont pas reflétées dans l'application.

Si l'exit renvoie MQXDR_OK dans la zone *ExitResponse* de la structure MQDXP, mais qu'il ne modifie pas les zones *Encoding* ou *CodedCharSetId* du descripteur de message, le gestionnaire de files d'attente renvoie pour ces zones les valeurs que les zones correspondantes de la structure MQDXP avaient en entrée de l'exit.

LongueurInBuffer
Type : MQLONG - entrée

Longueur en octets de *InBuffer*.

Il s'agit de la longueur de la mémoire tampon d'entrée *InBuffer*, qui indique le nombre d'octets à traiter par l'exit. *InBufferLength* est la longueur la moins élevée des données de message avant la conversion et la longueur de la mémoire tampon fournie par l'application lors de l'appel MQGET.

La valeur est toujours supérieure à zéro.

InBuffer
Type: MQBYTExInBufferLength -entrée

Mémoire tampon contenant le message non converti.

Contient les données du message avant la conversion. Si l'exit ne parvient pas à convertir les données, le gestionnaire de files d'attente renvoie le contenu de cette mémoire tampon à l'application une fois l'exit terminé.

Remarque : L'exit ne doit pas modifier *InBuffer*; si ce paramètre est modifié, les résultats ne sont pas définis.

Dans le langage de programmation C, ce paramètre est défini comme un pointeur vers vide.

OutBuffer
Type : MQLONG - entrée

Longueur en octets de *OutBuffer*.

Il s'agit de la longueur de la mémoire tampon de sortie *OutBuffer*, qui est identique à la longueur de la mémoire tampon fournie par l'application sur l'appel MQGET.

La valeur est toujours supérieure à zéro.

OutBuffer
Type: MQBYTExOutBufferLength -sortie

Mémoire tampon contenant le message converti.

En sortie de l'exit, si la conversion a abouti (comme indiqué par la valeur MQXDR_OK dans la zone *ExitResponse* du paramètre *DataConvExitParms*), *OutBuffer* contient les données de message à distribuer à l'application, dans la représentation demandée. Si la conversion a échoué, les modifications apportées par l'exit à cette mémoire tampon sont ignorées.

Dans le langage de programmation C, ce paramètre est défini comme un pointeur vers vide.

Notes d'utilisation

1. Un exit de conversion de données est un exit écrit par l'utilisateur qui reçoit le contrôle lors du traitement d'un appel MQGET. La fonction exécutée par l'exit de conversion de données est définie par le fournisseur de l'exit ; cependant, l'exit doit se conformer aux règles décrites ici et dans la structure de paramètres associée MQDXP.

   Les langages de programmation pouvant être utilisés pour un exit de conversion de données sont déterminés par l'environnement.
2. L'exit est appelé uniquement si *toutes* les conditions suivantes sont remplies:
   - L'option MQGMO_CONVERT est spécifiée dans l'appel MQGET
   - La zone *Format* du descripteur de message n'est pas MQFMT_NONE
   - Le message n'est pas déjà dans la représentation requise, c'est-à-dire que l'un ou les deux éléments *CodedCharSetId* et *Encoding* du message sont différents de la valeur spécifiée par l'application dans le descripteur de message fourni dans l'appel MQGET.
   - Le gestionnaire de files d'attente n'a pas déjà effectué la conversion avec succès
   - La longueur de la mémoire tampon de l'application est supérieure à zéro
   - La longueur des données de message est supérieure à zéro
   - Le code anomalie jusqu'à présent lors de l'opération MQGET est MQRC_NONE ou MQRC_TRUNCATED_MSG_ACCEPTED

3. Lorsqu'un exit est en cours d'écriture, pensez à le coder de manière à ce qu'il puisse convertir les messages qui ont été tronqués. Les messages tronqués peuvent se produire de l'une des manières suivantes:

   - L'application de réception fournit une mémoire tampon plus petite que le message, mais spécifie l'option MQGMO_ACCEPT_TRUNCATED_MSG dans l'appel MQGET.

     Dans ce cas, la zone *Reason* du paramètre *DataConvExitParms* en entrée de l'exit a la valeur MQRC_TRUNCATED_MSG_ACCEPTED.

   - L'expéditeur du message l'a tronqué avant de l'envoyer. Cela peut se produire avec des messages de rapport, par exemple (voir «Conversion des messages de rapport», à la page 904 pour plus de détails).

     Dans ce cas, la zone *Reason* du paramètre *DataConvExitParms* en entrée de l'exit a la valeur MQRC_NONE (si l'application de réception a fourni une mémoire tampon suffisamment grande pour le message).

   Par conséquent, la valeur de la zone *Reason* en entrée de l'exit ne peut pas toujours être utilisée pour déterminer si le message a été tronqué.

   La caractéristique distinctive d'un message tronqué est que la longueur fournie à l'exit dans le paramètre *InBufferLength* est *inférieure* à la longueur impliquée par le nom de format contenu dans la zone *Format* du descripteur de message. L'exit doit donc vérifier la valeur de *InBufferLength* avant de tenter de convertir les données ; l'exit *ne doit pas* supposer que la quantité totale de données impliquée par le nom de format a été fournie.

   Si l'exit n'a *pas* été écrit pour convertir les messages tronqués et que *InBufferLength* est inférieur à la valeur attendue, l'exit renvoie MQXDR_CONVERSION_FAILED dans la zone *ExitResponse* du paramètre *DataConvExitParms* , avec les zones *CompCode* et *Reason* définies sur MQCC_WARNING et MQRC_FORMAT_ERROR.

   Si l'exit *a* été écrit pour convertir les messages tronqués, l'exit convertit autant de données que possible (voir la note d'utilisation suivante), en prenant soin de ne pas tenter d'examiner ou de convertir les données au-delà de la fin de *InBuffer*. Si la conversion aboutit, l'exit laisse la zone *Reason* dans le paramètre *DataConvExitParms* inchangée. Renvoie MQRC_TRUNCATED_MSG_ACCEPTED si le message a été tronqué par le gestionnaire de files d'attente du récepteur et MQRC_NONE si le message a été tronqué par l'expéditeur du message.

   Il est également possible pour un message de développer *lors de la conversion* , jusqu'à ce qu'il soit supérieur à *OutBuffer*. Dans ce cas, l'exit doit décider s'il faut tronquer le message ; la zone *AppOptions* dans le paramètre *DataConvExitParms* indique si l'application de réception a spécifié l'option MQGMO_ACCEPT_TRUNCATED_MSG.

4. En règle générale, toutes les données du message fourni à l'exit dans *InBuffer* sont converties, ou aucune de ces données n'est convertie. Cependant, une exception à cette règle se produit si le message est tronqué, soit avant la conversion, soit pendant la conversion ; dans ce cas, il peut y avoir un élément incomplet à la fin de la mémoire tampon (par exemple: 1 octet d'un caractère codé sur deux octets, ou 3 octets d'un entier codé sur quatre octets). Dans ce cas, envisagez d'omettre l'élément incomplet et de définir les octets inutilisés dans *OutBuffer* sur des valeurs nulles. Toutefois, les éléments ou caractères complets d'un tableau ou d'une chaîne *doivent* être convertis.
5. Lorsqu'un exit est nécessaire pour la première fois, le gestionnaire de files d'attente tente de charger un objet portant le même nom que le format (à l'exception des extensions). L'objet chargé doit contenir l'exit qui traite les messages avec ce nom de format. Pensez à rendre le nom de l'exit et le nom de l'objet contenant l'exit identiques, bien que tous les environnements n'en aient pas besoin.
6. Une nouvelle copie de l'exit est chargée lorsqu'une application tente d'extraire le premier message qui utilise cette *Format* depuis que l'application s'est connectée au gestionnaire de files d'attente. Pour les applications CICS ou IMS , cela signifie que le sous-système CICS ou IMS est connecté au gestionnaire de files d'attente. Une nouvelle copie peut également être chargée à d'autres moments, si le gestionnaire de files d'attente a supprimé une copie précédemment chargée. Pour cette raison,

un exit ne doit pas tenter d'utiliser la mémoire statique pour communiquer des informations d'un appel de l'exit au suivant-l'exit peut être déchargé entre les deux appels.

7. S'il existe un exit fourni par l'utilisateur portant le même nom que l'un des formats intégrés pris en charge par le gestionnaire de files d'attente, l'exit fourni par l'utilisateur ne remplace pas la routine de conversion intégrée. Les seules circonstances dans lesquelles un tel exit est appelé sont les suivantes:
   - Si la routine de conversion intégrée ne peut pas gérer les conversions vers ou depuis le *CodedCharSetId* ou *Encoding* impliqué, ou
   - Si la routine de conversion intégrée n'a pas réussi à convertir les données (par exemple, parce qu'une zone ou un caractère ne peut pas être converti).
8. La portée de l'exit dépend de l'environnement. Les noms *Format* doivent être choisis pour réduire le risque de conflits avec d'autres formats. Commencez par les caractères qui identifient l'application définissant le nom de format.
9. L'exit de conversion de données s'exécute dans un environnement tel que celui du programme qui a émis l'appel MQGET ; l'environnement inclut l'espace adresse et le profil utilisateur (le cas échéant). Le programme peut être un agent de canal de transmission de messages envoyant des messages à un gestionnaire de files d'attente de destination qui ne prend pas en charge la conversion de messages. L'exit ne peut pas compromettre l'intégrité du gestionnaire de files d'attente, car il ne s'exécute pas dans l'environnement du gestionnaire de files d'attente.
10. Le seul appel MQI pouvant être utilisé par l'exit est MQXCNVC ; la tentative d'utilisation d'autres appels MQI échoue avec le code anomalie MQRC_CALL_IN_PROGRESS ou d'autres erreurs imprévisibles.
11. Aucun point d'entrée appelé MQ_DATA_CONV_EXIT n'est fourni par le gestionnaire de files d'attente. Toutefois, un `typedef` est fourni pour le nom MQ_DATA_CONV_EXIT dans le langage de programmation C et peut être utilisé pour déclarer l'exit écrit par l'utilisateur afin de s'assurer que les paramètres sont corrects. Le nom de l'exit doit être identique au nom de format (nom contenu dans la zone *Format* de MQMD), bien que cela ne soit pas obligatoire dans tous les environnements.

    L'exemple suivant montre comment l'exit qui traite le format MYFORMAT peut être déclaré dans le langage de programmation C:

```
#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. Sous z/OS, si un exit de croisement d'API est également en vigueur, il est appelé après l'exit de conversion de données.

Appel C

```
exitname (&DataConvExitParms, &MsgDesc, InBufferLength,
         InBuffer, OutBufferLength, OutBuffer);
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
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 */
```

Déclaration COBOL (IBM i uniquement)

```
     CALL 'exitname' USING DATACONVEXITPARMS, MSGDESC, INBUFFERLENGTH,
                           INBUFFER, OUTBUFFERLENGTH, OUTBUFFER.
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
**   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).
```

Déclaration assembleur System/390

```
         CALL EXITNAME,(DATACONVEXITPARMS,MSGDESC,INBUFFERLENGTH,      X
               INBUFFER,OUTBUFFERLENGTH,OUTBUFFER)
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
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
```

Propriétés spécifiées en tant qu'éléments MQRFH2

Les propriétés de descripteur de non-message peuvent être spécifiées en tant qu'éléments dans les dossiers d'en-tête MQRFH2 . Présentation des éléments MQRFH2 spécifiés en tant que propriétés.

Cela permet de conserver la compatibilité avec les versions précédentes des clients WebSphere MQ JMS et XMS . Cette section explique comment spécifier des propriétés dans les en-têtes MQRFH2 .

Pour utiliser des éléments MQRFH2 en tant que propriétés, spécifiez les éléments comme décrit dans Utilisation des classes WebSphere MQ pour Java. Ces informations complètent celles décrites dans «MQRFH2 - En-tête 2 de règles et de formatage», à la page 507.

Mappage de types de données de propriété à des types de données MQRFH2

Cette rubrique fournit des informations sur les types de propriété de message mappés à leurs types de données MQRFH2 correspondants.

Tableau 580. Types de données MQRFH2 pris en charge	
Type de propriété de message	**Type de données MQRFH2**
MQBYTE [ ]	bin.hex
MQBOOL	boolean
MQINT8	i1
MQINT16	i2
MQINT32	i4
MQINT64	i8
MQFLOAT32	r4
MQFLOAT64	r8
MQCHAR [ ]	String

Tout élément sans type de données est supposé être de type "chaîne".

Un type de données MQRFH2 de `int`, c'est-à-dire un entier de taille non spécifiée, est traité comme s'il s'agissait d'un `i8`.

Une valeur nulle est indiquée par l'attribut d'élément `xsi:nil='true'`. N'utilisez pas l'attribut `xsi:nil='false'` pour les valeurs non nulles.

Par exemple, la propriété suivante a une valeur null:

```
<NullProperty xsi:nil='true'></NullProperty>
```

Une propriété d'octet ou de chaîne de caractères peut avoir une valeur vide. Il est représenté par un élément MQRFH2 avec une valeur d'élément de longueur nulle.

Par exemple, la propriété suivante a une valeur vide:

```
<EmptyProperty></EmptyProperty>
```

Dossiers MQRFH2 pris en charge

Présentation de l'utilisation des zones de descripteur de message en tant que propriétés.

Les dossiers <jms>, <mcd>, <mqext>et <usr> sont décrits dans L'en-tête MQRFH2 et JMS. Le dossier <usr> est utilisé pour transporter les propriétés définies par l'application JMS associées à un message. Les groupes ne sont pas autorisés dans le dossier <usr> .

L'en-tête MQRFH2 et JMS prennent en charge les dossiers supplémentaires suivants:

- <mq>

  Ce dossier est utilisé et réservé aux propriétés définies par MQqui sont utilisées par IBM WebSphere MQ.

- <mq_usr>

  Ce dossier peut être utilisé pour transporter des propriétés définies par l'application qui ne sont pas exposées en tant que propriétés définies par l'utilisateur JMS, car les propriétés peuvent ne pas répondre aux exigences d'une propriété JMS. Ce dossier peut contenir des groupes que le dossier <usr> ne peut pas contenir.

- Tout dossier marqué avec l'attribut `content='properties'` .

  Ce type de dossier est équivalent au dossier <mq_usr> dans le contenu.

- <mqps>

Ce dossier est utilisé pour les propriétés de publication / abonnement IBM WebSphere MQ .

IBM WebSphere MQ prend également en charge les dossiers suivants qui sont déjà utilisés par WAS/SIB:

- `<sib>`

  Ce dossier est utilisé et réservé aux propriétés de message système WAS/SIB qui ne sont pas exposées en tant que propriétés JMS ou qui sont mappées aux propriétés JMS_IBM_ *, mais qui sont exposées aux applications WAS/SIB ; ces dernières incluent les propriétés des chemins de routage en aval et en amont.

  Certains au moins ne peuvent pas être exposés en tant que propriétés JMS, car il s'agit de tableaux d'octets. Si votre application ajoute des propriétés à ce dossier, la valeur est ignorée ou supprimée.

- `<sib_usr>`

  Ce dossier est utilisé et réservé aux propriétés de message utilisateur WAS/SIB qui ne peuvent pas être exposées en tant que propriétés utilisateur JMS car elles ne sont pas de types pris en charge ; elles sont exposées aux applications WAS/SIB.

  Il s'agit de propriétés utilisateur que vous pouvez obtenir ou définir via l'interface SIMessage, mais le contenu du tableau d'octets est mappé à la valeur de propriété requise.

  Si votre application IBM WebSphere MQ écrit un élément `bin.hex` arbitraire dans le dossier, l'application reçoit probablement un `IOException`, car il n'est pas au format attendu pour la restauration. Si vous ajoutez autre chose qu'un élément `bin.hex` , vous recevez un `ClassCastException`.

  N'essayez pas de rendre les propriétés disponibles pour WAS/SIB en utilisant ce dossier ; utilisez plutôt le dossier `<usr>` à cette fin.

- `<sib_context>`

  Ce dossier est utilisé pour les propriétés de message système WAS/SIB qui ne sont pas exposées aux applications utilisateur WAS/SIB ou en tant que propriétés JMS. Il s'agit notamment des propriétés de sécurité et transactionnelles utilisées pour les services Web et similaires.

  Votre application ne doit pas ajouter de propriétés à ce dossier.

- `<mqema>`

  Ce dossier a été utilisé par WAS/SIB à la place du dossier `<mqext>` .

Les noms de dossier MQRFH2 sont sensibles à la casse.

Les dossiers suivants sont réservés, dans n'importe quel mélange de minuscules ou de majuscules:

- Tout dossier préfixé par `mq` ou `wmq`; réservé à l'utilisation par IBM WebSphere MQ.
- Tout dossier préfixé par `sib`; réservé à l'utilisation par WAS/SIB.
- Dossiers `<Root>` et `<Body>` ; réservés mais non utilisés.

Les dossiers suivants ne sont pas reconnus comme contenant des propriétés de message:

- `<psc>`

  Utilisé par WebSphere Message Broker pour transmettre des messages de commande de publication / abonnement au courtier.

- `<pscr>`

  Utilisé par WebSphere Message Broker pour contenir des informations provenant du courtier, en réponse à des messages de commande de publication / abonnement.

- Tout dossier non défini par WebSphere Message Broker, qui n'est pas marqué avec l'attribut `content='properties'` .

Ne spécifiez pas `content='properties'` dans les dossiers `<psc>` ou `<pscr>` . Dans ce cas, ces dossiers sont traités comme des propriétés et WebSphere Message Broker risque de ne plus fonctionner comme prévu.

Si votre application génère des messages avec des propriétés, dans les en-têtes MQRFH2 à reconnaître comme un en-tête MQRFH2 contenant des propriétés, l'en-tête doit figurer dans la liste des en-têtes pouvant être chaînés en tête du message.

MQRFH2 peut être précédé d'un nombre quelconque d'en-têtes standard MQH, d'un MQCIH, d'un MQDLH, d'un MQIIH, d'un MQTM, d'un MQTMC2ou d'un MQXQH. Une chaîne ou un MQCFH se termine l'analyse syntaxique car ils ne peuvent pas être chaînés.

Il est possible qu'un message contienne plusieurs en-têtes MQRFH2 contenant toutes des propriétés de message. Les dossiers portant le même nom peuvent coexister dans des en-têtes différents, sauf restriction contraire, par exemple par WAS/SIB. Les dossiers sont traités comme un seul dossier logique, s'ils sont tous dans des en-têtes significatifs.

Alors que les dossiers des en-têtes significatifs ne peuvent pas être fusionnés avec ceux des en-têtes non significatifs, les dossiers portant le même nom dans les en-têtes significatifs peuvent être fusionnés, ce qui supprime les propriétés en conflit. Vos applications ne doivent pas dépendre de la présentation des propriétés dans leur message.

Les groupes MQRFH2 sont analysés pour rechercher des propriétés dans les dossiers définis par l'utilisateur, c'est-à-dire dans les dossiers <wmq>, <jms>, <mcd>, <usr>, <mqext>, <sib>, <sib_usr>, <sib_context>et <mqema> .

Les groupes des dossiers de propriétés définis par IBM, à l'exception des dossiers <wmq> et <mq> , sont analysés pour les propriétés.

Un dossier MQRFH2 ne peut pas contenir de contenu mixte ; un dossier ou un groupe peut contenir des groupes ou des propriétés, ou une valeur, mais pas les deux.

Un segment d'un message, qu'il s'agisse du premier segment ou d'un segment suivant, ne peut pas contenir de propriétés définies par IBM WebSphere MQautres que celles du descripteur de message. Par conséquent, l'insertion d'un message contenant de telles propriétés avec MQMF_SEGMENT ou MQMF_SEGMENTATION_ALLOWED entraîne l'échec de l'insertion avec MQRC_SEGMENTATION_NOT_ALLOWED.

Toutefois, les groupes de messages peuvent contenir des propriétés définies par IBM WebSphere MQ.

Génération d'en-têtes MQRFH2

Si WebSphere MQ convertit les propriétés de message en leur représentation MQRFH2 , il doit ajouter MQRFH2 au message. Il ajoute le fichier MQRFH2 sous la forme d'un en-tête distinct ou le fusionne avec un en-tête existant.

La génération de nouveaux en-têtes MQRFH2 par WebSphere MQ peut interrompre les en-têtes existants dans un message. Les applications qui analysent une mémoire tampon de messages pour les en-têtes doivent savoir que le nombre et la position des en-têtes dans une mémoire tampon peuvent changer dans certaines circonstances. WebSphere MQ tente de réduire l'impact de l'ajout de propriétés à un message en fusionnant des propriétés de message dans un en-tête MQRFH2 existant, là où il le peut. Il tente également de minimiser l'impact en insérant un MQRFH2 généré dans une position fixe par rapport à d'autres en-têtes dans la mémoire tampon du message.

Un en-tête MQRFH2 généré est placé après le MQMD, ainsi que n'importe quel nombre d'en-têtes MQXQH, MQRFHet MQDLH , quel que soit l'ordre dans lequel ils se trouvent. L'en-tête MQRFH2 généré est placé immédiatement avant le premier en-tête qui n'est pas un en-tête MQMD, MQXQH, MQDLHou MQRFH .

Règles de fusion des MQRFH2 générées

Les règles suivantes s'appliquent à la fusion d'un MQRFH2 généré avec un MQRFH2existant. L'en-tête MQRFH2 généré est fusionné avec un en-tête MQRFH2 existant, si:

1. Le MQRFH2 existant se trouve à la même position WebSphere MQ place un MQRFH2généré ou une version antérieure dans la chaîne d'en-tête.
2. Le CCSID des propriétés générées est identique au CCSIDNameValue du MQRFH2existant.

Dans le cas contraire, l'en-tête généré est placé séparément dans la mémoire tampon, dans la position décrite précédemment.

Règles de fusion des dossiers dans un MQRFH2 existant

Si les propriétés de message sont fusionnées dans un MQRFH2existant, le MQRFH2 existant est analysé pour rechercher les dossiers qui correspondent aux propriétés de message et les fusionne. Si un dossier correspondant n'existe pas, un nouveau dossier est ajouté à la fin des dossiers existants. Si un dossier correspondant existe, la recherche est effectuée dans le dossier. Toutes les propriétés correspondantes sont écrasées. Tous les nouveaux sont ajoutés à la fin du dossier.

Restrictions du dossier MQRFH2

Présentation des restrictions de dossier dans les en-têtes MQRFH2

Les restrictions MQRFH2 s'appliquent aux dossiers suivants:

- Les noms d'élément du dossier <usr> ne doivent pas commencer par le préfixe JMS; ces noms de propriété sont réservés à l'utilisation par JMS et ne sont pas valides pour les propriétés définies par l'utilisateur.

  Un tel nom d'élément n'entraîne pas l'échec de l'analyse syntaxique de MQRFH2 , mais il n'est pas accessible aux API de propriété de message WebSphere MQ .
- Les noms d'élément du dossier <usr> ne doivent pas être, dans une combinaison de minuscules ou de majuscules, NULL, TRUE, FALSE, NOT, AND, OR, BETWEEN, LIKE, IN, IS et ESCAPE. Ces noms correspondent à des mots clés SQL et rendent les sélecteurs d'analyse syntaxique plus difficiles, car <usr> est le dossier par défaut utilisé lorsqu'aucun dossier n'est spécifié pour une propriété particulière dans un sélecteur.

  Un tel nom d'élément n'entraîne pas l'échec de l'analyse syntaxique de MQRFH2 , mais il n'est pas accessible aux API de propriété de message WebSphere MQ .
- Les noms d'élément d'un dossier considéré comme contenant des propriétés de message ne doivent pas contenir de point (.) (caractère Unicode U+002E), car il est utilisé dans les noms de propriété pour indiquer la hiérarchie.

  Un tel nom d'élément n'entraîne pas l'échec de l'analyse syntaxique de MQRFH2 , mais il n'est pas accessible aux API de propriété de message WebSphere MQ .

En général, les en-têtes MQRFH2 qui contiennent des données de style XML valides peuvent être analysés par WebSphere MQ sans échec, bien que certains éléments de MQRFH2 ne soient pas accessibles via les API de propriété de message WebSphere MQ .

Conflits de noms d'élément MQRFH2

Présentation des conflits dans les noms d'élément MQRFH2 .

Une seule valeur peut être associée à une propriété de message. Si une tentative d'accès à une propriété entraîne un conflit de valeurs, l'une d'elles est choisie de préférence à l'autre.

La syntaxe WebSphere MQ permettant d'accéder aux éléments MQRFH2 permet l'identification unique d'un élément, si un dossier ne contient aucun élément portant le même nom. Si un dossier contient plusieurs éléments portant le même nom, la valeur de la propriété utilisée est celle la plus proche de la tête du message.

Cela s'applique si plusieurs dossiers du même nom sont contenus dans des en-têtes MQRFH2 significatifs différents dans le même message.

Un conflit peut se produire lorsque l'appel MQGET est traité après qu'une propriété de descripteur de non-message a été définie deux fois: via un appel MQSETMP et directement dans l'en-tête MQRFH2 brut.

Si cela se produit, la propriété associée au message par un appel API prend la préférence sur une dans les données de message, c'est-à-dire sur celle de l'en-tête MQRFH2 brut. Si un conflit se produit, il est considéré comme venant logiquement avant les données du message.

Mappage des noms de propriété vers les noms de dossier et d'élément MQRFH2

Présentation des différences entre les noms de propriété et les noms d'élément dans l'en-tête MQRFH2 .

Lorsque vous utilisez l'une des API définies qui génèrent en fin de compte des en-têtes MQRFH2 , afin de spécifier les propriétés de message (par exemple, MQ JMS), le nom de propriété n'est pas nécessairement le nom d'élément dans le dossier MQRFH2 .

Par conséquent, un mappage est effectué entre le nom de la propriété et l'élément MQRFH2 , et inversement, en prenant en compte à la fois le nom du dossier qui contient l'élément et le nom de l'élément. Certains exemples de IBM WebSphere MQ classes for JMS sont déjà documentés dans Utilisation de Java.

Nom de la propriété	Nom de dossier MQRFH2	Nom d'élément MQRFH2
JMSDestination	jms	Dst
JMSType	mcd	Type, Set, Fmt
xxx (défini par l'utilisateur, où xxx ne commence pas par JMS)	usr	xxx

Par conséquent, lorsqu'une application JMS accède à la propriété JMSDestination , elle est mappée à l'élément Dst dans le dossier <jms> .

Lorsque vous spécifiez des propriétés en tant qu'éléments MQRFH2 , IBM WebSphere MQ définit ses éléments comme suit:

Nom de la propriété	Nom de dossier MQRFH2	Nom de groupe MQRFH2	Nom d'élément MQRFH2
<Property>	<usr>	Non applicable	<Property>
<folder>.<Property>	<folder>	Non applicable	<Property>
<folder>.<group>.<Property>	<folder>	<group>	<Property>

Par exemple, lorsqu'une application JMS IBM WebSphere MQ tente d'accéder à la propriété Property1 , elle est mappée à l'élément Property1 dans le dossier <usr> . La propriété wmq . Property2 est mappée à la propriété Property2 dans le dossier <wmq> .

Si le nom de la propriété en contient plusieurs. , le nom d'élément MQRFH2 utilisé est celui qui suit la valeur finale. et les groupes MQRFH2 sont utilisés pour former une hiérarchie ; les groupes MQRFH2 imbriqués sont autorisés.

L'en-tête JMS et les propriétés spécifiques au fournisseur qui sont contenues dans un MQRFH2 dans les dossiers <mcd>, <jms>et <mqext> sont accessibles par une application IBM WebSphere MQ à l'aide des noms abrégés définis dans Utilisation des classes WebSphere MQ pour Java.

Les propriétés JMS définies par l'utilisateur sont accessibles à partir du dossier <usr> . Une application IBM WebSphere MQ peut utiliser le dossier <usr> pour ses propriétés d'application s'il est acceptable que la propriété apparaisse aux applications JMS comme l'une de ses propriétés définies par l'utilisateur.

S'il n'est pas acceptable, choisissez un autre dossier ; le dossier <wmq_usr> est fourni comme emplacement standard pour ces propriétés non JMS.

Vos applications peuvent spécifier et utiliser n'importe quel dossier MQRFH2 avec une utilisation bien définie, non documentée dans «Propriétés spécifiées en tant qu'éléments MQRFH2», à la page 921 si vous notez ce qui suit:

1. Il se peut que le dossier soit déjà utilisé ou qu'il soit utilisé ultérieurement par une autre application fournissant un accès non défini aux propriétés qu'il contient. Voir Noms de propriété pour connaître la convention de dénomination suggérée pour les noms de propriété.

2. Les propriétés ne sont pas accessibles aux versions précédentes du client IBM WebSphere MQ classes for JMS ou XMS qui peuvent uniquement accéder au dossier <usr> pour les propriétés définies par l'utilisateur.
3. Le dossier doit être marqué avec l'attribut `content` avec la valeur définie sur `properties`, par exemple, `content='properties'`.

   «MQSETMP-Définition de la propriété de message», à la page 769 ajoute automatiquement cet attribut si nécessaire. Cet attribut ne doit être ajouté à aucun des dossiers définis par IBM, par exemple, <jms> et <usr>. Ce faisant, le message est rejeté par le client IBM WebSphere MQ classes for JMS avant la version 7.0avec un `MessageFormatException`.

Etant donné que le dossier <usr> est l'emplacement par défaut des propriétés de la syntaxe <Property> , une application IBM WebSphere MQ et une application JMS permettent d'accéder à la même valeur de propriété définie par l'utilisateur à l'aide du même nom.

Noms de dossier réservés

Il existe plusieurs noms de dossier réservés. Vous ne pouvez pas utiliser de tels noms comme préfixes de dossier ; par exemple, `Root.Property1` n'accède pas à une propriété valide car `Root` est réservé. La liste suivante contient les noms de dossiers réservés:

- Racine
- Corps
- Propriétés
- Environnement
- LocalEnvironment
- DestinationList
- ExceptionList
- InputBody
- InputRoot
- InputProperties
- Environnement InputLocal
- Liste InputDestination
- Liste InputException
- OutputRoot
- Environnement OutputLocal
- Liste OutputDestination
- Liste OutputException

Mappage des zones de descripteur de propriété dans les en-têtes MQRFH2

Lorsqu'une propriété est convertie en élément MQRFH2 , les attributs d'élément suivants sont utilisés pour spécifier les zones significatives du descripteur de propriété: Cette section décrit comment les zones MQPD sont converties en attributs d'élément MQRFH2 .

Support

La zone de descripteur de propriété de support est divisée en trois attributs d'élément

- L'attribut d'élément **sr** spécifie des valeurs dans le masque de bit MQPD_REJECT_UNSUP_MASK.
- L'attribut d'élément **sa** spécifie des valeurs dans le masque de bits MQPD_ACCEPT_UNSUP_MASK.
- L'attribut d'élément **sx** spécifie des valeurs dans le masque de bit MQPD_ACCEPT_UNSUP_IF_XMIT_MASK.

Ces attributs d'élément ne sont valides que dans le dossier < mq> et sont ignorés s'ils sont définis sur des éléments des autres dossiers contenant des propriétés.

Valeur de prise en charge	Attribut d'élément MQRFH2	Valeur de l'attribut MQRFH2
MQPD_SUPPORT_XX_ENCODE_CASE_ONE facultatif	sa	facultatif Il s'agit de la valeur par défaut.
MQPD_SUPPORT_REQUIS	sr	obligatoire
MQPD_SUPPORT_REQUIRED_IF_LOCAL	sx	locales

Contexte

Utilisez l'attribut d'élément **context** pour indiquer le contexte de message auquel appartient une propriété. Utilisez une seule valeur. Cet attribut d'élément est valide sur une propriété dans tout dossier contenant des propriétés.

Valeur du contexte	Valeur de l'attribut MQRFH2
MQPD_NO_CONTEXT	aucun Il s'agit de la valeur par défaut.
MQPD_USER _CONTEXT	utilisateur

CopyOptions

Utilisez l'attribut d'élément **copy** pour indiquer les messages dans lesquels une propriété doit être copiée. Plusieurs valeurs sont admises ; séparez les valeurs par une virgule. Par exemple, **copy='reply'** et **copy='publish,report'** sont valides. Cet attribut d'élément est valide sur une propriété dans tout dossier contenant des propriétés.

Remarque : Dans la définition d'attribut, les apostrophes ou les guillemets sont valides, par exemple **copy='reply'** ou **copy="report"**

Valeur CopyOption	Valeur de l'attribut MQRFH2
MQPD_COPY_FORWARD	transmission
MQPD_RÉPONSE_COPIE	répondre
RAPPORT MQPD_COPY_REPORT	rapport
MQPD_COPY_PUBLISH	Publication
MQPD_COPY_ALL	Tous Ne spécifiez pas cette valeur avec une autre valeur. Lorsqu'elle est utilisée avec une autre valeur, elle est prioritaire sur toute valeur à l'exception de **none**.
MQPD_COPY_DEFAULT	par défaut Il s'agit de la valeur par défaut. Cela revient à spécifier les trois valeurs MQCOPY_FORWARD, MQCOPY_REPORT et MQCOPY_PUBLISH. Ne spécifiez pas cette valeur avec une autre valeur.

Valeur CopyOption	Valeur de l'attribut MQRFH2
MQPD_COPY_NONE	aucun Ne spécifiez pas cette valeur avec une autre valeur. Lorsqu'elle est utilisée avec une autre valeur, elle est prioritaire.

Restrictions du dossier < mq> MQRFH2

Lorsqu'un message est inséré dans une file d'attente, il est recherché dans un dossier < mq> afin que le message puisse être traité en fonction de ses propriétés définies par MQ. Pour permettre une analyse syntaxique efficace des propriétés définies par MQ, les restrictions suivantes s'appliquent au dossier:

- Seules les propriétés du premier dossier < mq> significatif du message sont traitées par MQ; les propriétés de tout autre dossier < mq> du message sont ignorées.
- Si le dossier est en UTF-8, seuls les caractères UTF-8 mono-octet sont autorisés dans le dossier. Un caractère multi-octet dans le dossier peut entraîner l'échec de l'analyse syntaxique et le rejet du message.
- N'incluez pas les groupes MQRFH2 dans le dossier < mq>. La présence du caractère Unicode U+003C dans une valeur de propriété entraîne le rejet du message.
- N'utilisez pas de chaînes d'échappement dans le dossier. Une chaîne d'échappement est traitée comme la valeur réelle de l'élément.
- Seul le caractère Unicode U+0020 est traité comme un espace dans le dossier. Tous les autres caractères sont traités comme significatifs et peuvent entraîner l'échec de l'analyse syntaxique du dossier et le rejet du message.

Si l'analyse du dossier < mq> échoue ou si le dossier ne respecte pas ces restrictions, le message est rejeté avec CompCode **MQCC_FAILED** et Reason **MQRC_RFH_RESTRICTED_FORMAT_ERR**.

En-têtes MQRFH2 non valides

Lors d'un appel MQPUT, MQPUT1ou MQGET, une analyse partielle des en-têtes MQRFH2 du message peut être effectuée pour vérifier quels dossiers sont inclus et pour déterminer si les dossiers contiennent des propriétés. Présentation des en-têtes MQRFH2 non valides.

Si l'analyse partielle du message ne peut pas aboutir car la structure n'est pas valide, par exemple, la zone `StrucLength` est trop petite, alors:

- L'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC_RFH_ERROR, s'il est possible de déterminer que l'application inclut une option WebSphere MQ Version 7, de sorte que les applications existantes n'échouent pas.
- L'appel MQGET est renvoyé avec succès et l'en-tête MQRFH2 contenant l'erreur est renvoyé dans la mémoire tampon que vous avez fournie.

Si l'analyse syntaxique partielle échoue car il est impossible de détecter si un dossier particulier contient des propriétés ou non, par exemple, le dossier commence par `<<jms`. L'analyse syntaxique échoue donc avant que le nom du dossier ne soit déterminé.

- L'appel MQPUT ou MQPUT1 échoue avec le code anomalie MQRC_RFH_FORMAT_ERROR, s'il est possible de déterminer que l'application inclut une option WebSphere MQ Version 7, de sorte que les applications existantes n'échouent pas.
- L'appel MQGET est renvoyé avec succès et l'en-tête MQRFH2 contenant l'erreur est renvoyé dans la mémoire tampon que vous avez fournie.
- En interne dans le gestionnaire de files d'attente, le message n'est pas rejeté en raison du dossier mal formaté, mais le dossier est toujours traité comme s'il ne contenait aucune propriété.

Un message peut circuler dans le réseau du gestionnaire de files d'attente avec un dossier contenant une telle erreur de syntaxe, mais il n'est jamais analysé et détecté, alors qu'un ou plusieurs dossiers du message sont:

- Valide
- L'analyse syntaxique a abouti
- Utilisé dans le traitement du message

Par conséquent, la détection n'est pas garantie.

Si l'une de vos applications utilise «MQSETMP-Définition de la propriété de message», à la page 769ou MQINQMP pour accéder à une propriété, ce qui entraîne l'analyse complète d'un dossier MQRFH2 et la détection d'une erreur telle que l'analyse ne peut pas aboutir, cela est indiqué par un code retour approprié à l'appel d'API. Aucune propriété du dossier n'est mise à la disposition de l'application.

Si une tentative d'analyse complète d'un dossier MQRFH2 est effectuée et que l'analyseur syntaxique trouve des attributs d'élément non reconnus ou un type de données non reconnu, l'analyse se poursuit et aboutit sans qu'aucun avertissement ne soit émis ; cela ne constitue pas une erreur d'analyse.

Conversion de page de code

Cette section décrit les noms de jeu de codes et les CCSID, la langue nationale, la conversion z/OS , la conversion IBM i et la prise en charge de la conversion Unicode.

Chaque section de la langue nationale contient les informations suivantes:

- Les CCSID natifs pris en charge
- Conversions de page de codes **non** prises en charge

Les termes suivants sont utilisés dans les informations:

-8
Indique, pour HP-UX , que le CCSID est défini pour le jeu de codes HP-UX *roman8*

AIX
Indique WebSphere MQ for AIX

HP-UX
Indique WebSphere MQ for HP-UX

Linux
Indique WebSphere MQ for Linux for Intel et WebSphere MQ for Linux for zSeries

HP Integrity NonStop Server
Indique WebSphere MQ for HP Integrity NonStop Server

OS/400
Indique WebSphere MQ for IBM i

Solaris
Indique WebSphere MQ for Solaris

Windows
Indique WebSphere MQ for Windows

z/OS
Indique WebSphere MQ for z/OS

La valeur par défaut pour la conversion de données est la conversion à effectuer sur le système cible (récepteur).

Si le produit source prend en charge la conversion, un canal peut être configuré et les données échangées en définissant l'attribut de canal CONVERT sur YES au niveau de la source.

Remarque :

1. La conversion des informations client WebSphere MQ MQI s'effectue sur le serveur, de sorte que le serveur doit prendre en charge la conversion du CCSID client vers le CCSID serveur.

2. La conversion peut inclure la prise en charge ajoutée par CSD/PTF à la version la plus récente de WebSphere MQ. Vérifiez le contenu du dernier niveau de service pour savoir si vous devez installer une CSD/PTF pour activer cette conversion.

Voir Tableau 581, à la page 931 pour une référence croisée entre certains numéros de CCSID et certains noms de jeu de codes de l'industrie.

Noms de jeu de codes et CCSID

WebSphere MQ for z/OS fournit davantage de conversions que celles répertoriées dans les tables spécifiques à la langue.

Tableau 581. Noms de jeu de codes et CCSID

Noms de jeu de codes	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 (euro)	923
big5	950
eucJP	954 5050 33722
eucKR	970
eucTW	964
eucCN	1383
PCK	943
GBK	1386
koi8-r	878

Langues nationales

Ces informations contiennent les langues prises en charge par WebSphere MQ.

Les langues prises en charge par WebSphere MQ sont les suivantes:

- Anglais (Etats-Unis)-voir la rubrique «Anglais (Etats-Unis)», à la page 932
- Allemand-voir la rubrique «allemand», à la page 933
- Danois et norvégien-voir la rubrique «Danois et norvégien», à la page 933
- Finnois et suédois-voir la rubrique «Finnois et suédois», à la page 934
- Italien-voir la rubrique «italien», à la page 935
- Espagnol-voir la rubrique «espagnol», à la page 935

- Anglais / gaélique du Royaume-Uni-voir la rubrique «Anglais / gaélique du Royaume-Uni», à la page 936
- Français-voir la rubrique «français», à la page 937
- Multilingue-voir la rubrique «Multilingue», à la page 937
- Portugais-voir la rubrique «Portugais», à la page 938
- Islandais-voir la rubrique «Islandais», à la page 938
- Langues d'Europe orientale-voir la rubrique «Langues d'Europe orientale», à la page 939
- Cyrillique-voir la rubrique «Cyrillique», à la page 940
- Estonien-voir la rubrique «Estonien», à la page 941
- Letton et lituanien-voir la rubrique «Letton et lituanien», à la page 942
- Ukrainien-voir la rubrique «Ukrainien», à la page 943
- Grec-voir la rubrique «Grec», à la page 944
- Turc-voir la rubrique «Turc», à la page 944
- Hébreu-voir la rubrique «Hébreu», à la page 945
- Farsi-voir la rubrique «Persan», à la page 947
- Urdu-voir la rubrique «Ourdou», à la page 947
- Thaï-voir la rubrique «Thaï», à la page 947
- Lao-voir la rubrique «Laotien», à la page 948
- Vietnamien-voir la rubrique «Vietnamien», à la page 948
- Japonais latin SBCS-voir la rubrique «Japonais latin SBCS», à la page 948
- Japonais Katakana SBCS-voir la rubrique «Japonais Katakana SBCS», à la page 950
- Japonais Kanji / Latin Mixed-voir la rubrique «Japonais Kanji / Latin Mixte», à la page 951
- Japonais Kanji / Katakana Mixte-voir la rubrique «Japonais Kanji / Katakana Mixte», à la page 953
- Coréen-voir la rubrique «coréen», à la page 954
- Chinois simplifié-voir la rubrique «Chinois simplifié», à la page 954
- Chinois traditionnel-voir la rubrique «chinois traditionnel», à la page 955

Anglais (Etats-Unis)

Détails des CCSID et de la conversion de CCSID pour l'anglais américain.

Le tableau suivant présente les CCSID natifs pour l'anglais américain sur les plateformes prises en charge:

Plateforme	CCSID natifs
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
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

37
Ne convertit pas en pages de codes 923, 858

924
Ne convertit pas en pages de codes 437, 858, 1051, 1140, 1252, 1275, 5348

1140
Ne convertit pas en pages de codes 924, 1051, 1275

allemand

Détails des CCSID et de la conversion de CCSID pour l'allemand.

Le tableau suivant présente les CCSID natifs pour l'allemand sur les plateformes prises en charge:

Plateforme	CCSID natifs
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
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

273
Ne convertit pas en pages de codes 858, 923, 924, 1275

924
Ne convertit pas en pages de codes 273, 437, 858, 1051, 1141, 1252, 1275, 5348

1141
Ne convertit pas en pages de codes 924, 1051, 1275

Danois et norvégien

Détails des CCSID et de la conversion de CCSID pour le danois et le norvégien.

Le tableau suivant présente les CCSID natifs pour le danois et le norvégien sur les plateformes prises en charge:

Plateforme	CCSID natifs
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
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

277
Ne convertit pas en pages de codes 858, 923, 924, 1275

924
Ne convertit pas en pages de codes 277, 858, 865, 1051, 1142, 1252, 1275, 5348

1142
Ne convertit pas en pages de codes 924, 865, 1051, 1275

AIX

Page de codes :

819
Ne convertit pas en page de codes 865

HP-UX

Page de codes :

1051
Ne convertit pas en page de codes 865

Windows

Page de codes :

865
Ne convertit pas en pages de codes 1051, 1275

Finnois et suédois

Détails des CCSID et de la conversion de CCSID pour le finnois et le suédois.

Le tableau suivant présente les CCSID natifs pour le finnois et le suédois sur les plateformes prises en charge:

Plateforme	CCSID natifs
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
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

278
Ne convertit pas en pages de codes 858, 923, 924, 1275

924
Ne convertit pas en pages de codes 278, 437, 858, 865, 1051, 1143, 1252, 1275, 5348

1143
Ne convertit pas en pages de codes 865, 924, 1051, 1275

AIX

Page de codes :

819
Ne convertit pas en page de codes 865

850
Ne convertit pas en page de codes 865

HP-UX

Page de codes :

1051
Ne convertit pas en page de codes 865

Windows

Page de codes :

865
Ne convertit pas en pages de codes 1051, 1275

italien

Détails des CCSID et de la conversion de CCSID pour l'italien.

Le tableau suivant présente les CCSID natifs pour l'italien sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	280, 924, 1144
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

280
Ne convertit pas en pages de codes 858, 923, 924, 1275

924
Ne convertit pas en pages de codes 280, 437, 858, 1051, 1144, 1252, 1275, 5348

1144
Ne convertit pas en pages de codes 924, 1051, 1275

espagnol

Détails des CCSID et de la conversion de CCSID pour l'espagnol.

Le tableau suivant présente les CCSID natifs pour l'espagnol sur les plateformes prises en charge:

Plateforme	CCSID natifs
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
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

284
Ne convertit pas en pages de codes 858, 923, 924, 1275

924
Ne convertit pas en pages de codes 284, 437, 858, 1051, 1145, 1252, 1275, 5348

1145
Ne convertit pas en pages de codes 924, 1051, 1275

Anglais / gaélique du Royaume-Uni

Détails des CCSID et de la conversion de CCSID pour l'anglais / gaélique britannique.

Le tableau suivant présente les CCSID natifs pour l'anglais / gaélique du Royaume-Uni sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	285, 924, 1146
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

285
Ne convertit pas en pages de codes 858, 923, 924, 1275

924
Ne convertit pas en pages de codes 285, 437, 858, 1051, 1146, 1252, 1275, 5348

1146
Ne convertit pas en pages de codes 924, 1051, 1275

français

Détails des CCSID et de la conversion de CCSID pour le français.

Le tableau suivant présente les CCSID natifs pour le français sur les plateformes prises en charge:

Plateforme	CCSID natifs
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
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

297
Ne convertit pas en pages de codes 858, 923, 924, 1275, 5348

924
Ne convertit pas en pages de codes 297, 437, 858, 1051, 1147, 1252, 1275, 5348

1147
Ne convertit pas en pages de codes 924, 1051, 1275

Multilingue

Détails des CCSID et de la conversion de CCSID pour Multilingue.

Le tableau suivant présente les CCSID natifs pour la conversion multilingue sur les plateformes prises en charge:

Plateforme	CCSID natifs
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
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

500
Ne convertit pas en pages de codes 858, 923

924
Ne convertit pas en pages de codes 437, 858, 1051, 1148, 1252, 1275, 5348

1148
Ne convertit pas en pages de codes 924, 1051, 1275

Portugais

Détails des CCSID et de la conversion de CCSID pour le portugais.

Le tableau suivant présente les CCSID natifs pour le portugais sur les plateformes prises en charge:

Plateforme	CCSID natifs
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
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

37
Ne convertit pas en pages de codes 858, 923, 1275

500
Ne convertit pas en pages de codes 858, 923, 1275

924
Ne convertit pas en pages de codes 858, 860, 1051, 1140, 1252, 1275, 5348

1140
Ne convertit pas en pages de codes 860, 924, 1051, 1275

HP-UX

Page de codes :

1051
Ne convertit pas en page de codes 860

Windows

Page de codes :

860
Ne convertit pas en pages de codes 1051, 1275

Islandais

Détails des CCSID et de la conversion de CCSID pour l'islandais.

Le tableau suivant présente les CCSID natifs pour l'islandais sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	871, 924, 1149

Plateforme	CCSID natifs
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	850, 858, 861, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
Client Apple	1275

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

871
Ne convertit pas en pages de codes 858, 923, 924, 1275, 5348

924
Ne convertit pas en pages de codes 858, 861, 871, 1051, 1149, 1252, 1275, 5348

1149
Ne convertit pas en pages de codes 924, 1051, 1275

HP-UX

Page de codes :

1051
Ne convertit pas en page de codes 861

Windows

Page de codes :

861
Ne convertit pas en pages de codes 1051, 1275

Langues d'Europe orientale

Détails des CCSID et de la conversion de CCSID pour les langues d'Europe de l'Est. Les langues typiques utilisant ces CCSID sont l'albanais, le croate, le tchèque, le hongrois, le polonais, le roumain, le serbe, le slovaque et le slovène.

Le tableau suivant présente les CCSID natifs pour les langues d'Europe de l'Est sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	870, 1153
Windows	852, 1250, 5346, 9044
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux	912
Client Apple d'Europe de l'Est	1282
Client Apple roumain	1285
Client Apple croate	1284

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

870
Ne convertit pas en pages de codes 1284, 1285

1153
Ne convertit pas en pages de codes 1250, 1284, 1285

IBM i

Page de codes :

870
Ne convertit pas en pages de codes 1284, 1285, 5346, 9044

1153
Ne convertit pas en pages de codes 1282, 1284, 1285, 5346, 9044

HP-UX, Solaris, Linux

Page de codes :

912
Ne convertit pas en pages de codes 1284, 1285

HP Integrity NonStop Server

Page de codes :

912
Ne convertit pas en pages de codes 1153, 1284, 1285, 9044

Windows

Page de codes :

852
Ne convertit pas en pages de codes 1284, 1285

1250
Ne convertit pas en pages de codes 1284, 1285

9044
Ne convertit pas en pages de codes 912, 1282, 1284, 1285

Cyrillique

Détails des CCSID et de la conversion de CCSID pour Cyrillique. Les langues typiques utilisant ces CCSID sont le biélorusse, le bulgare, le macédonien, le russe et le serbe.

Le tableau suivant présente les CCSID natifs pour Cyrillique sur les plateformes prises en charge:

Plateforme	CCSID natifs
z/OS	1025
IBM i	880, 1025
Windows	855, 866, 1131, 1251, 5347
Solaris	878, 915

Plateforme	CCSID natifs
AIX, HP-UX, Linux, HP Integrity NonStop Server	915
Client Apple	1283

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

880
Ne convertit pas en pages de codes 855, 866, 878, 1131, 5347

1025
Ne convertit pas en pages de codes 878, 5347

Windows

Page de codes :

855
Ne convertit pas en page de codes 1131

866
Ne convertit pas en page de codes 1131

1131
Ne convertit pas en pages de codes 855, 866, 880, 1283

Estonien

Détails des CCSID et de la conversion de CCSID pour l'estonien.

Le tableau suivant présente les CCSID natifs pour l'estonien sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1122, 1157
Windows	902, 922, 1257, 5353, 9449
AIX, HP-UX, Solaris, Linux	902, 922
HP Integrity NonStop Server	922

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

1122
Ne convertit pas en pages de codes 902, 1157, 9449

1157
Ne convertit pas en pages de codes 922, 1122, 1257, 9449

IBM i

Page de codes :

1122
Ne convertit pas en pages de codes 902, 5353, 9449

1157
Ne convertit pas en pages de codes 922, 5353, 9449

HP-UX, Solaris, Linux

Page de codes :

902
Ne convertit pas en pages de codes 922, 1122, 9449

922
Ne convertit pas en pages de codes 902, 1157, 9449

Windows

Page de codes :

5353
Ne convertit pas en page de codes 9449

9449
Ne convertit pas en pages de codes 902, 922, 1122, 1157, 1257, 5353

902
Ne convertit pas en pages de codes 922, 1122, 9449

HP Integrity NonStop Server

Page de codes :

922
Ne convertit pas en pages de codes 902, 1157, 9449

Letton et lituanien

Détails des CCSID et de la conversion de CCSID pour le letton et le lituanien.

Le tableau suivant présente les CCSID natifs pour le letton et le lituanien sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1112, 1156
Windows	901, 921, 1257, 5353, 9449
AIX, HP-UX, Solaris, Linux	901, 921
HP Integrity NonStop Server	921

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

1112
Ne convertit pas en pages de codes 901, 1156, 9449

1156
Ne convertit pas en pages de codes 901, 1156, 9449

IBM i

Page de codes :

1112
Ne convertit pas en page de codes 5353

1153
Ne convertit pas en pages de codes 921, 5353, 9449

HP-UX, Solaris, Linux

Page de codes :

902
Ne convertit pas en pages de codes 921, 1112, 1257, 9449

921
Ne convertit pas en pages de codes 901, 1156, 9449

Windows

Page de codes :

901
Ne convertit pas en pages de codes 921, 1112, 1257, 9449

5355
Ne convertit pas en page de codes 9449

9449
Ne convertit pas en pages de codes 901, 921, 1112, 1156, 1257

HP Integrity NonStop Server

Page de codes :

921
Ne convertit pas en pages de codes 901, 1156, 9449

Ukrainien

Détails des CCSID et de la conversion de CCSID pour l'ukrainien.

Le tableau suivant présente les CCSID natifs pour l'ukrainien sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1123
Windows	1124, 1125, 1251, 5347
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux	1124

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

1123
Ne convertit pas en page de codes 5347

HP-UX

Page de codes :

1124
Ne convertit pas en page de codes 5347

Windows

Page de codes :

1125
Ne convertit pas en page de codes 1123

Grec

Détails des CCSID et de la conversion de CCSID pour le grec.

Le tableau suivant présente les CCSID natifs pour le grec sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	875
HP-UX	813 (voir note)
Windows	869, 1253, 5349
AIX, NCR, HP Integrity NonStop Server, Solaris, Linux	813
Client Apple	1280
Client DOS	737
Remarque : Seul le jeu de codes ISO est pris en charge sur HP-UX. Le jeu de codes HP-UX propriétaire greek8 n'a pas de CCSID enregistré et n'est pas pris en charge.	

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs, les CCSID natifs des autres plateformes avec les exceptions suivantes.

IBM i

Page de codes :

875
Ne convertit pas en page de codes 5349

Windows

Page de codes :

1253
Ne convertit pas en page de codes 737

5349
Ne convertit pas en page de codes 737

Turc

Détails des CCSID et de la conversion de CCSID pour le turc.

Le tableau suivant présente les CCSID natifs pour le turc sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1026
HP-UX	920 (voir remarque)
Windows	857, 1254, 5350

Plateforme	CCSID natifs
AIX, HP Integrity NonStop Server, Solaris, Linux	920
Client Apple	1281
Remarque : Seul le jeu de codes ISO est pris en charge sur HP-UX. Le jeu de codes HP-UX propriétaire turkish8 n'a pas de CCSID enregistré et n'est pas pris en charge.	

Toutes les plateformes non-client prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

1026
Ne convertit pas en page de codes 5350

Hébreu

Détails des CCSID et de la conversion de CCSID pour l'hébreu.

Le tableau suivant présente les CCSID natifs pour l'hébreu sur les plateformes prises en charge:

Plateforme	CCSID natifs
z/OS	424, 803, 4899, 12712
IBM i	424
AIX	916, 9048
HP-UX	916 (voir note)
Windows	1255, 5351
HP Integrity NonStop Server, Solaris, Linux	916
Remarque : Seul le jeu de codes ISO est pris en charge sur HP-UX. Le jeu de codes HP-UX propriétaire greek8 n'a pas de CCSID enregistré et n'est pas pris en charge.	

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

424
Ne convertit pas en pages de codes 867, 4899, 9048, 12712

803
Ne convertit pas en pages de codes 867, 4899, 5351, 9048, 12712

4899
Ne convertit pas en pages de codes 424, 803, 856, 862, 916, 1255

12712
Ne convertit pas en pages de codes 424, 803, 856, 916, 1255

IBM i

Page de codes :

424
Ne convertit pas en pages de codes 803, 867, 4899, 5351, 9048, 12712

La page de codes 424 est également convertie vers et depuis le CCSID 4952, qui est une variante de 856.

AIX

Page de codes :

916
Ne convertit pas en pages de codes 867, 4899, 9048, 12712

9048
Ne convertit pas en pages de codes 424, 803, 856, 862, 916, 1255

Windows

Page de codes :

1255
Ne convertit pas en pages de codes 867, 4899, 9048, 12712

5351
Ne convertit pas en page de codes 803

Arabe

Détails des CCSID et de la conversion de CCSID pour l'arabe

Le tableau suivant présente les CCSID natifs pour l'arabe sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	420
AIX	1046, 1089
HP-UX	1089 (voir note)
Windows	720, 864, 1256, 5352
HP Integrity NonStop Server, Solaris, Linux	1089
Remarque : Seul le jeu de codes ISO est pris en charge sur HP-UX. Le jeu de codes HP-UX propriétaire arabic8 n'a pas de CCSID enregistré et n'est pas pris en charge.	

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

420
Ne convertit pas en page de codes 5352

HP-UX, Solaris, Linux, HP Integrity NonStop Server, Tru64

Page de codes :

1089
Ne convertit pas en page de codes 720

Windows

Page de codes :

720
Ne convertit pas en pages de codes 1089, 5352

5352
Ne convertit pas en page de codes 720

Persan

Détails des CCSID et de la conversion de CCSID pour Farsi.

Le tableau suivant présente les CCSID natifs pour Farsi sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1097
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux Windows	1098 (voir note)
Remarque : Le CCSID natif de ces plateformes n'a pas été normalisé et peut être modifié.	

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes.

Ourdou

Détails des CCSID et de la conversion de CCSID pour l'ourdou.

Le tableau suivant présente les CCSID natifs pour l'ourdou sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	918
Windows	868
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux	1006

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

918
Ne convertit pas en page de codes 1006

Thaï

Détails des CCSID et de la conversion de CCSID pour le thaï.

Le tableau suivant présente les CCSID natifs pour le thaï sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	838
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux Windows	874 (voir note)
Remarque : Le CCSID natif de ces plateformes n'a pas été normalisé et peut être modifié.	

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes.

Laotien

Détails des CCSID et de la conversion de CCSID pour le laotien.

Le tableau suivant présente les CCSID natifs pour le laotien sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1132
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux Windows	1133

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes.

Vietnamien

Détails des CCSID et de la conversion de CCSID pour le vietnamien.

Le tableau suivant présente les CCSID natifs pour le vietnamien sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1130
Windows	1258, 5354
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux	1129

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

IBM i

Page de codes :

1130
Ne convertit pas en pages de codes 1129, 5354

Japonais latin SBCS

Détails des CCSID et de la conversion de CCSID pour les SBCS latins japonais.

Le tableau suivant présente les CCSID natifs pour le japonais latin SBCS sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1027
AIX	932, 5050, 33722 (voir la note 1)
Windows	932, 943 (voir notes 2 et 3)
Linux, HP Integrity NonStop Server, Solaris	943, 5050
HP-UX	Inconnu

Plateforme	CCSID natifs
Remarque : 1. 5050 et 33722 sont des CCSID liés à la page de codes de base 954 sous AIX. Le CCSID signalé par le système d'exploitation est 33722. 2. Windows NT utilise la page de codes 932, mais celle-ci est mieux représentée par le CCSID 943. Toutefois, toutes les plateformes de WebSphere MQ ne prennent pas en charge ce CCSID. Sous WebSphere MQ for Windows , le CCSID 932 est utilisé pour représenter la page de codes 932, mais une modification du fichier `../conv/table/ccsid.tbl` peut être effectuée, qui modifie le CCSID utilisé en 943. 3. WebSphere MQ ne prend pas en charge les pages de codes basées sur la norme JIS X 0213 (JIS2004).	

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

1027
Non converti en pages de codes 932, 942, 943, 954, 5050, 33722

IBM i

Page de codes :

1027
Ne convertit pas en page de codes 932

AIX

Page de codes :

932
Ne convertit pas en page de codes 1027

5050
Ne convertit pas en page de codes 1027

33722
Ne convertit pas en page de codes 1027

Linux

Page de codes :

943
Ne convertit pas en page de codes 1027

5050
Ne convertit pas en page de codes 1027

Solaris

Page de codes :

943
Ne convertit pas en page de codes 1027

5050
Ne convertit pas en page de codes 1027

HP Integrity NonStop Server

Page de codes :

943
Ne convertit pas en page de codes 1027

5050
Ne convertit pas en page de codes 1027

Japonais Katakana SBCS

Détails des CCSID et de la conversion de CCSID pour le japonais Katakana SBCS.

Le tableau suivant présente les CCSID natifs pour le japonais Katakana SBCS sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	290
HP-UX	897
AIX	932, 5050, 33722 (voir la note 1)
Windows	932, 943 (voir notes 2 et 3)
Linux, HP Integrity NonStop Server, Solaris	943, 5050

Remarque :

1. 5050 et 33722 sont des CCSID liés à la page de codes de base 954 sous AIX. Le CCSID signalé par le système d'exploitation est 33722.
2. Windows NT utilise la page de codes 932, mais celle-ci est mieux représentée par le CCSID 943. Toutefois, toutes les plateformes de WebSphere MQ ne prennent pas en charge ce CCSID.

   Sous WebSphere MQ for Windows , le CCSID 932 est utilisé pour représenter la page de codes 932, mais une modification du fichier `../conv/table/ccsid.tbl` peut être effectuée, qui modifie le CCSID utilisé en 943.
3. WebSphere MQ ne prend pas en charge les pages de codes basées sur la norme JIS X 0213 (JIS2004).
4. Outre les conversions ci-dessus, les produits WebSphere MQ sous AIX, HP-UX, Solaris, Linux et Tru64 prennent en charge la conversion du CCSID 897 en CCSID 37, 273, 277, 278, 280, 284, 285, 290, 297, 437, 500, 819, 850, 1027 et 1252.

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

290
Ne convertit pas en pages de codes 932, 943, 954, 5050, 33722

IBM i

Page de codes :

290
Ne convertit pas en page de codes 932

AIX

Page de codes :

932
Ne convertit pas en pages de codes 290, 897

5050
Ne convertit pas en pages de codes 290, 897

33722
Ne convertit pas en pages de codes 290, 897

HP-UX

Page de codes :

897
Ne convertit pas en pages de codes 932, 943, 954, 5050, 33722

Linux

Page de codes :

943
Ne convertit pas en pages de codes 290, 897

5050
Ne convertit pas en pages de codes 290, 897

Solaris

Page de codes :

943
Ne convertit pas en pages de codes 290, 897

5050
Ne convertit pas en pages de codes 290, 897

HP Integrity NonStop Server

Page de codes :

943
Ne convertit pas en pages de codes 290, 897

5050
Ne convertit pas en pages de codes 290, 897

Japonais Kanji / Latin Mixte

Détails des CCSID et de la conversion de CCSID pour le japonais Kanji / Latin Mixed.

Le tableau suivant présente les CCSID natifs pour le japonais Kanji / latin Mixed sur les plateformes prises en charge:

Plateforme	CCSID natifs
IBM i, z/OS	1399, 5035 (voir la note 1)
AIX	932, 5050, 33722 (voir la remarque 2)
HP-UX	932, 954, 5039 (voir note 3)
Windows	932, 943 (voir notes 4 et 5)

Plateforme	CCSID natifs
Linux, HP Integrity NonStop Server, Solaris	943, 5050

Remarque :

1. 5035 est un CCSID associé à la page de codes 939
2. 5050 et 33722 sont des CCSID liés à la page de codes de base 954 sous AIX. Le CCSID signalé par le système d'exploitation est 33722.
3. Les jeux de codes `japan15` et SJIS sous HP-UX sont représentés par le CCSID 932. Certains caractères DBCS ayant des représentations différentes dans SJIS, 932 peut être converti de manière incorrecte si la conversion n'est pas effectuée sur un système HP-UX . WebSphere MQ for HP-UX prend en charge 5039, le CCSID correct pour HP SJIS. Une modification du fichier `/var/mqm/conv/ccsid.tbl` peut être effectuée pour modifier le CCSID utilisé de 932 à 5039.
4. Windows NT utilise la page de codes 932, mais celle-ci est mieux représentée par le CCSID 943. Toutefois, toutes les plateformes de WebSphere MQ ne prennent pas en charge ce CCSID.

   Sous WebSphere MQ for Windows , le CCSID 932 est utilisé pour représenter la page de codes 932, mais une modification du fichier `../conv/table/ccsid.tbl` peut être effectuée, qui modifie le CCSID utilisé en 943.
5. WebSphere MQ ne prend pas en charge les pages de codes basées sur la norme JIS X 0213 (JIS2004).

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

1399
Ne convertit pas en pages de codes 954, 5035, 5050, 33722

5035
Ne convertit pas en pages de codes 954, 1399, 5050, 33722

IBM i

Page de codes :

1399
Ne convertit pas en page de codes 5039

5035
Ne convertit pas en page de codes 5039

HP-UX

Page de codes :

932
Ne convertit pas en pages de codes 942, 943, 1399

954
Ne convertit pas en pages de codes 942, 943, 1399

5039
Ne convertit pas en pages de codes 942, 943, 1399

HP Integrity NonStop Server

Page de codes :

943
Ne convertit pas en page de codes 1399

5050
Ne convertit pas en page de codes 1399

Japonais Kanji / Katakana Mixte

Détails des CCSID et de la conversion de CCSID pour le japonais Kanji / Katakana Mixed.

Tableau 582. CCSID natifs pour Kanji / Katakana japonais Mixte sur les plateformes prises en charge

Plateforme	CCSID natifs
z/OS	1390, 5026 (voir la note 1)
IBM i	5026 (voir note 1)
AIX	932, 5050, 33722 (voir la remarque 2)
HP-UX	932, 954, 5039 (voir note 3)
Windows	932, 943 (voir notes 4 et 5)
Linux, HP Integrity NonStop Server, Solaris	943, 5050

Remarque :

1. Le CCSID 1390 n'accepte pas les caractères minuscules. 5026 est un CCSID associé à la page de codes 930. CCSID 5026 est le CCSID signalé sur IBM i lorsque la fonction japonaise Katakana (DBCS) est sélectionnée.
2. 5050 et 33722 sont des CCSID liés à la page de codes de base 954 sous AIX. Le CCSID signalé par le système d'exploitation est 33722.
3. Les jeux de codes `japan15` et SJIS sous HP-UX sont représentés par le CCSID 932. Certains caractères DBCS ayant des représentations différentes dans SJIS, 932 peut être converti de manière incorrecte si la conversion n'est pas effectuée sur un système HP-UX . WebSphere MQ for HP-UX prend en charge 5039, le CCSID correct pour HP SJIS. Une modification du fichier `/var/mqm/conv/ccsid.tbl` peut être effectuée pour modifier le CCSID utilisé de 932 à 5039.
4. Windows NT utilise la page de codes 932, mais celle-ci est mieux représentée par le CCSID 943. Toutefois, toutes les plateformes de WebSphere MQ ne prennent pas en charge ce CCSID.

   Sous WebSphere MQ for Windows, le CCSID 932 est utilisé pour représenter la page de codes 932, mais une modification du fichier `../conv/table/ccsid.tbl` peut être effectuée qui modifie le CCSID utilisé en 943.
5. WebSphere MQ ne prend pas en charge les pages de codes basées sur la norme JIS X 0213 (JIS2004).

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

1390
Ne convertit pas en pages de codes 954, 5026, 5050, 33722

N'accepte pas les caractères minuscules.

5026
Ne convertit pas en pages de codes 954, 1390, 5050, 33722

IBM i

Page de codes :

5026
Ne convertit pas en pages de codes 1390, 5039

HP-UX

Page de codes :

932
Ne se convertit pas en pages de codes 942, 943, 1390

954
Ne se convertit pas en pages de codes 942, 943, 1390

5039
Ne se convertit pas en pages de codes 942, 943, 1390

HP Integrity NonStop Server

Page de codes :

943
Ne convertit pas en page de codes 1390

5050
Ne convertit pas en page de codes 1390

coréen

Détails des CCSID et de la conversion de CCSID pour le coréen.

Le tableau suivant présente les CCSID natifs pour le coréen sur les plateformes prises en charge:

Plateforme	CCSID natifs
z/OS, IBM i	933, 1364
AIX, HP-UX, Linux, HP Integrity NonStop Server, Solaris	970
Windows	949, 1363

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

933
Ne convertit pas en page de codes 970

1364
Ne convertit pas en page de codes 970

HP-UX

Page de codes :

970
Ne se convertit pas en pages de codes 949, 1363, 1364

Chinois simplifié

Détails des CCSID et de la conversion de CCSID pour le chinois simplifié.

Le tableau suivant présente les CCSID natifs pour le chinois simplifié sur les plateformes prises en charge:

Plateforme	CCSID natifs
z/OS	935, 1388
IBM i	935, 1388
AIX	1383, 1386
HP-UX	1381 (voir note 1)
Windows	1381, 1386 (voir note 2)
Linux, HP Integrity NonStop Server, Solaris	1383

Remarque :

1. Les jeux de codes `prc15` et `hp15CN` sous HP-UX sont représentés par le CCSID 1381.
2. Windows utilise la page de codes 936, mais celle-ci est mieux représentée par le CCSID 1386. Toutefois, toutes les plateformes de WebSphere MQ ne prennent pas en charge ce CCSID.

   Sous WebSphere MQ for Windows , le CCSID 1381 est utilisé pour représenter la page de codes 936, mais une modification du fichier `../conv/table/ccsid.tbl` peut être effectuée, qui modifie le CCSID utilisé en 1386.
3. WebSphere MQ prend en charge la première phase de la norme GB18030 en chinois.

   Sous z/OS, Linux, Windowset Solaris, la prise en charge de la conversion est assurée entre Unicode (UTF-8 et UCS-2) et CCSID 1388 (EBCDIC avec les extensions GB18030), Unicode (UTF-8 et UCS-2) et CCSID 5488 (GB18030 phase un), et entre CCSID 1388 et CCSID 5488.

   Remarque :

   Sous IBM i, le système d'exploitation prend en charge la conversion entre Unicode (UTF-8 et UCS-2) et CCSID 1388 (EBCDIC avec les extensions GB18030).

   Sous HP-UX , aucune prise en charge n'est actuellement disponible sur le système d'exploitation HP11 pour GB18030. Sous HP11i, le correctif PHCO_26456 fournit une prise en charge de la conversion entre GB18030 (CCSID 5488) et Unicode. La prise en charge n'est pas fournie pour la conversion entre GB18030 et 1388 (EBCDIC).

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

935
Ne convertit pas en page de codes 1383

1388
Ne convertit pas en page de codes 1383

HP-UX

Page de codes :

1381
Ne convertit pas en pages de codes 1383, 1386, 1388

chinois traditionnel

Détails des CCSID et de la conversion de CCSID pour le chinois traditionnel.

Le tableau suivant présente les CCSID natifs pour le chinois traditionnel sur les plateformes prises en charge:

Plateforme	CCSID natifs
z/OS, IBM i	937
HP-UX	938, 950, 964 (voir remarque)
Windows	950
AIX, HP Integrity NonStop Server, Solaris, Linux	950, 964
Remarque : Le jeu de codes `roc15` sous HP-UX est représenté par le CCSID 938.	

Toutes les plateformes prennent en charge la conversion entre leurs CCSID natifs et les CCSID natifs des autres plateformes, avec les exceptions suivantes.

z/OS

Page de codes :

937
Ne convertit pas en page de codes 964

1388
Ne convertit pas en page de codes 1383

HP-UX

Page de codes :

938
Ne convertit pas en page de codes 948

950
Ne convertit pas en page de codes 948

964
Ne convertit pas en page de codes 948

Linux, Solaris

Page de codes :

964
Ne convertit pas en page de codes 938

Prise en charge de la conversion Unicode

Certaines plateformes prennent en charge la conversion des données utilisateur vers ou depuis le codage Unicode. Les deux formes de codage Unicode prises en charge sont UCS-2 (CCSID 1200, 13488 et 17584) et UTF-8 (CCSID 1208).

Le terme *UCS-2* est souvent utilisé de manière interchangeable, mais de manière incorrecte avec *UTF-16*. UCS-2 est un codage à largeur fixe dans lequel chaque caractère occupe 2 octets. UTF-16 est un codage à largeur variable qui est un sur-ensemble de UCS-2. Outre les caractères UCS-2 à 2 octets, UTF-16 contient des caractères, appelés paires de substitution, d'une longueur de 4 octets. WebSphere MQ ne prend pas en charge les paires de substitution. La prise en charge des caractères UTF-16 et UTF-8 dans WebSphere MQ est donc limitée aux caractères Unicode qui peuvent être codés dans UCS-2.

Remarque : WebSphere MQ ne prend pas en charge les CCSID du gestionnaire de files d'attente UCS-2 . Par conséquent, les données d'en-tête de message ne peuvent pas être codées dans UCS-2.

WebSphere MQ AIX pour Unicode

Sous WebSphere MQ for AIX , la conversion vers et depuis les CCSID Unicode est prise en charge pour les CCSID du tableau suivant.

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	1 200
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 pour Unicode

Sur WebSphere MQ for HP-UX , la conversion vers et depuis les CCSID Unicode est prise en charge pour les CCSID répertoriés dans le tableau suivant.

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
1 200	1208	1250	1251	1252	1253
1254	1255	1256	1257	1258	1381
5050	5488	13488	33722		

WebSphere MQ for Windows, Solaris et Linux pour Unicode

Sous WebSphere MQ for Windows, WebSphere MQ for Solaris et WebSphere MQ for Linux , la conversion vers et depuis les CCSID Unicode est prise en charge pour les CCSID du tableau suivant.

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	1 200	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)		

Remarques :

1. 931 utilise 939 pour la conversion.
2. 932 utilise 942 pour la conversion.
3. 938 utilise 948 pour la conversion.
4. 954 et 33722 utilisent 5050 pour la conversion.
5. Sous Windows, Linuxet Solaris uniquement.

Prise en charge d'Unicode par IBM i

Pour plus de détails sur la prise en charge d'UNICODE, reportez-vous à la publication IBM i appropriée relative à votre système d'exploitation.

Prise en charge d' WebSphere MQ for z/OS pour Unicode

Sur WebSphere MQ for z/OS , la conversion vers et depuis les CCSID Unicode est prise en charge pour les CCSID suivants:

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	1 012	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	1 200	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			

Normes de codage sur les plateformes 64 bits

Utilisez ces informations pour en savoir plus sur les normes de codage sur les plateformes 64 bits et les types de données préférés.

Types de données préférés

Ces types ne changent jamais de taille et sont disponibles sur les plateformes WebSphere MQ 32 bits et 64 bits:

Nom	Longueur
MQLONG	4 octets
MQULONG	4 octets
MQINT32	4 octets
MQUINT32	4 octets
MQINT64	8 octets
MQUINT64	8 octets

Types de données standard

Découvrez les types de données standard sur les applications UNIX32 bits, UNIX64 bits et Windows 64 bits.

Applications UNIX 32 bits

Cette section est incluse à des fins de comparaison et est basée sur Solaris. Toute différence avec les autres plateformes UNIX est notée:

Nom	Longueur
Caractère	1 octet
court	2 octets
int	4 octets
long	4 octets
float	4 octets
double	8 octets
long double	16 octets Notez que sous AIX et Linux PPC, un double long est de 8 octets.
pointeur	4 octets
ptrdiff_t	4 octets
taille_t	4 octets
time_t	4 octets
clock_t	4 octets

Nom	Longueur
wchar_t	4 octets
	Notez que sous AIX , la valeur de wchar_t est de 2 octets.

Applications UNIX 64 bits

Cette section est basée sur Solaris. Toute différence avec les autres plateformes UNIX est notée:

Nom	Longueur
Caractère	1 octet
court	2 octets
int	4 octets
long	8 octets
float	4 octets
double	8 octets
long double	16 octets
	Notez que sous AIX et Linux PPC, un double long est de 8 octets.
pointeur	8 octets
ptrdiff_t	8 octets
taille_t	8 octets
time_t	8 octets
clock_t	8 octets
	Notez que sur les autres plateformes UNIX , clock_t est de 4 octets.
wchar_t	4 octets
	Notez que sous AIX , la valeur de wchar_t est de 2 octets.

Applications Windows 64 bits

Nom	Longueur
Caractère	1 octet
court	2 octets
int	4 octets
long	4 octets
float	4 octets
double	8 octets
long double	8 octets
pointeur	8 octets
	Notez que tous les pointeurs sont de 8 octets.
ptrdiff_t	8 octets
taille_t	8 octets

Nom	Longueur
time_t	8 octets
clock_t	4 octets
wchar_t	2 octets
Word	2 octets
DWORD	4 octets
HANDLE	8 octets
HFILE	4 octets

Remarques sur le codage sous Windows

DESCRIPTEUR hf ;

Utiliser

```
hf = CreateFile((LPCTSTR) FileName,
                Access,
                ShareMode,
                xihSecAttsNTRestrict,
                Create,
                AttrAndFlags,
                NULL);
```

Ne pas utiliser

```
HFILE hf;
hf = (HFILE) CreateFile((LPCTSTR) FileName,
                        Access,
                        ShareMode,
                        xihSecAttsNTRestrict,
                        Create,
                        AttrAndFlags,
                        NULL);
```

car cela génère une erreur.

size_t len fgets

Utiliser

```
size_t len
while (fgets(string1, (int) len, fp) != NULL)
len = strlen(buffer);
```

Ne pas utiliser

```
int len;

while (fgets(string1, len, fp) != NULL)
len = strlen(buffer);
```

printf

Utiliser

```
printf("My struc pointer: %p", pMyStruc);
```

Ne pas utiliser

```
printf("My struc pointer: %x", pMyStruc);
```

Si vous avez besoin d'une sortie hexadécimale, vous devez imprimer les 4 octets supérieurs et inférieurs séparément.

char * ptr

Utiliser

```
char * ptr1;
char * ptr2;
size_t bufLen;

bufLen = ptr2 - ptr1;
```

Ne pas utiliser

```
char *ptr1;
char *ptr2;
UINT32 bufLen;

bufLen = ptr2 - ptr1;
```

alignBytes

Utiliser

```
alignBytes = (unsigned short) ((size_t) address % 16);
```

Ne pas utiliser

```
void *address;
unsigned short alignBytes;

alignBytes = (unsigned short) ((UINT32) address % 16);
```

len

Utiliser

```
len = (UINT32) ((char *) address2 - (char *) address1);
```

Ne pas utiliser

```
void *address1;
void *address2;
UINT32 len;

len = (UINT32) ((char *) address2 - (char *) address1);
```

sscanf

Utiliser

```
MQLONG SBCSprt;

sscanf(line, "%d", &SBCSprt);
```

Ne pas utiliser

```
MQLONG SBCSprt;

sscanf(line, "%1d", &SBCSprt);
```

%ld tente d'insérer un type à 8 octets dans un type à 4 octets ; n'utilisez %l que si vous utilisez un type de données long réel. MQLONG, UINT32 et INT32 sont définis sur quatre octets, identiques à un int sur toutes les plateformes WebSphere MQ :

Référence SOAP

Transport WebSphere MQ pour les informations de référence SOAP classées par ordre alphabétique.

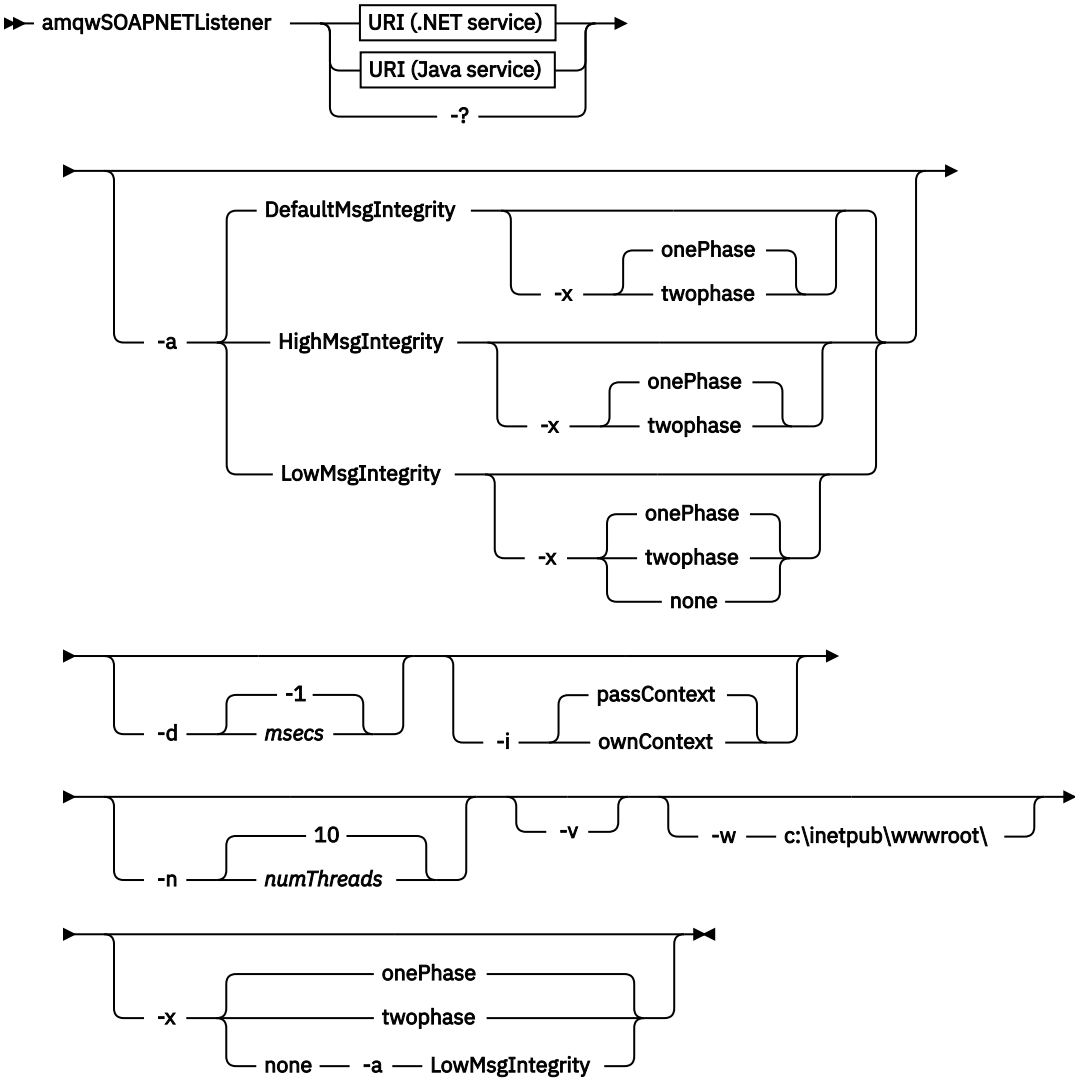
amqwSOAPNETListener: programme d'écoute SOAP IBM WebSphere MQ pour .NET Framework 1 ou 2

Syntaxe et paramètres du programme d'écoute SOAP WebSphere MQ pour .NET Framework 1 ou 2.

Objet

Démarre le programme d'écoute SOAP IBM WebSphere MQ pour .NET Framework 1 ou 2.

.NET



Paramètres obligatoires

URI ***plateforme***
Voir «Syntaxe d'URI et paramètres pour le déploiement de service Web», à la page 1004.

-?
Imprimez le texte d'aide décrivant la façon dont la commande est utilisée.

Paramètres optionnels

-a ***integrityOption***
integrityOption indique le comportement des programmes d'écoute SOAP WebSphere MQ s'il n'est pas possible d'insérer un message de demande ayant échoué dans la file d'attente des messages non livrés. *integrityOption* peut prendre l'une des valeurs suivantes:

DefaultMsgIntegrity
Pour les messages non persistants, le programme d'écoute affiche un message d'avertissement et continue de s'exécuter avec le message d'origine en cours de suppression. Pour les messages persistants, il affiche un message d'erreur, annule le message de demande pour qu'il reste dans la file d'attente des demandes et se ferme. DefaultMsgIntegrity s'applique si l'option -a est omise ou si l'option *integrityOption* n'est pas spécifiée.

LowMsgIntegrity
Pour les messages persistants et non persistants, le programme d'écoute affiche un avertissement et continue de s'exécuter, en supprimant le message.

HighMsgIntegrity
Pour les messages persistants et non persistants, le programme d'écoute affiche un message d'erreur, annule le message de demande afin qu'il reste dans la file d'attente des demandes et se ferme.

L'utilitaire de déploiement vérifie la compatibilité des indicateurs -x et -a . Si -x none est spécifié, -a LowMsgIntegrity doit être spécifié. Si les indicateurs sont incompatibles, l'utilitaire de déploiement se ferme avec un message d'erreur et aucune étape de déploiement n'a été effectuée.

-d ***ms***
msecs indique le nombre de millisecondes pendant lequel le programme d'écoute SOAP WebSphere MQ doit rester actif si des messages de demande ont été reçus sur une unité d'exécution. Si *msecs* est défini sur -1, le programme d'écoute reste actif indéfiniment.

-i ***Contexte***
Contexte indique si les programmes d'écoute transmettent le contexte d'identité. *Context* prend les valeurs suivantes:

passContext
Définissez le contexte d'identité du message de demande d'origine dans le message de réponse. Le programme d'écoute SOAP vérifie qu'il est autorisé à sauvegarder le contexte à partir de la file d'attente des demandes et à le transmettre à la file d'attente des réponses. Il effectue les vérifications lors de l'exécution lors de l'ouverture de la file d'attente de demandes pour sauvegarder le contexte et de la file d'attente de réponses pour transmettre le contexte. S'il ne dispose pas des droits requis, ou si l'appel MQOPEN échoue et que le message de réponse n'est pas traité. Le message de réponse est inséré dans la file d'attente des messages non livrés avec l'en-tête de message non livrés contenant le code retour du MQOPENayant échoué. Le programme d'écoute continue ensuite à traiter les messages entrants suivants de manière normale.

ownContext
Le programme d'écoute SOAP ne transmet pas de contexte. Le contexte renvoyé reflète l'ID utilisateur sous lequel le programme d'écoute s'exécute plutôt que l'ID utilisateur qui a créé le message de demande d'origine.

Les zones du contexte d'origine sont définies par le gestionnaire de files d'attente et non par le programme d'écoute SOAP.

-n ***numThreads***
numThreads indique le nombre d'unités d'exécution dans les scripts de démarrage générés pour le programme d'écoute SOAP WebSphere MQ . La valeur par défaut est 10. Envisagez d'augmenter ce nombre si vous disposez d'un débit de messages élevé.

-v
-v définit la sortie prolixe des commandes externes. Les messages d'erreur sont toujours affichés. Utilisez -v pour générer des commandes que vous pouvez personnaliser pour créer des scripts de déploiement personnalisés.

-w ***serviceDirectory***
serviceDirectory est le répertoire contenant le service Web.

-x ***transactionnalité***
transactionality indique le type de contrôle transactionnel pour le programme d'écoute. *transactionality* peut être définie sur l'une des valeurs suivantes:

onePhase
La prise en charge en une phase de IBM WebSphere MQ est utilisée. Si le système échoue lors du traitement, le message de demande est redistribué à l'application. Les transactions WebSphere MQ garantissent que les messages de réponse sont écrits une seule fois.

twoPhase
La prise en charge en deux phases est utilisée. Si le service est écrit de manière appropriée, le message est distribué une seule fois, en coordination avec d'autres ressources, au sein d'une seule exécution validée du service. Cette option s'applique uniquement aux connexions de liaisons de serveur.

none
Pas de prise en charge transactionnelle. Si le système échoue lors du traitement, le message de demande peut être perdu, même s'il est persistant. Le service peut avoir ou non été exécuté et les messages de réponse, de rapport ou de rebut peuvent ou non être écrits.

L'utilitaire de déploiement vérifie la compatibilité des indicateurs -x et -a . Pour plus d'informations, voir la description de l'indicateur -a .

Exemple .NET

```
amqwSOAPNETlistener
-u "jms:/queue?destination=myQ&connectionFactory=()
&targetService=myService&initialContextFactory=com.ibm.mq.jms.Nojndi"
-w C:/wmqsoap/demos
-n 20
```

amqswsdl: générez le service WSDL pour .NET Framework 1 ou 2

amqswsdl prend un service Web écrit pour .NET Framework 1 ou 2 et génère le WSDL pour la classe, en insérant l'URI que vous fournissez pour le transport WebSphere MQ pour SOAP dans le WSDL généré.

Objet

Utilisez **amqswsdl** pour générer un fichier WSDL contenant l'URI du service déployé dans WebSphere MQ. Utilisez le WSDL pour générer des proxys client.

Paramètres

escapedUri **(Entrée)**
URI du service, avec tous les "&" échappés dans "&amp.". Exemple :

```
"jms:/queue?destination=REQUESTDOTNET
&amp.initialContextFactory=com.ibm.mq.jms.Nojndi
&amp.connectionFactory=(connectQueueManager(QM1)binding(server))
&amp.targetService=Quote.asmx"
```

className**.asmx (Entrée)**
Classe de service.

***className*.wsdl (Sortie)**
WSDL du service.

Description

Si la classe est implémentée à l'aide du modèle de programmation code-behind, vous devez générer *className*.dll et le stocker dans ./bin.

amqwclientconfig: création d'un descripteur de déploiement de client de services Web Axis 1.4 pour WebSphere MQ transport for SOAP

amqwclientconfig crée le fichier de descripteur de déploiement du client client-config.wsdd Axis 1.4 .

Objet

Il ajoute le transport jms:/ au descripteur et enregistre java:com.ibm.mq.soap.transport.jms.WMQSender en tant que classe pour gérer les demandes SOAP pour le transport jms: .

Syntaxe

Description

amqwclientconfig appelle **amqwsetcp** pour définir CLASSPATH et exécute la commande suivante:

```
java org.apache.axis.utils.Admin client "%WMQSOAP_HOME%\bin\amqwclientTransport.wsdd"
```

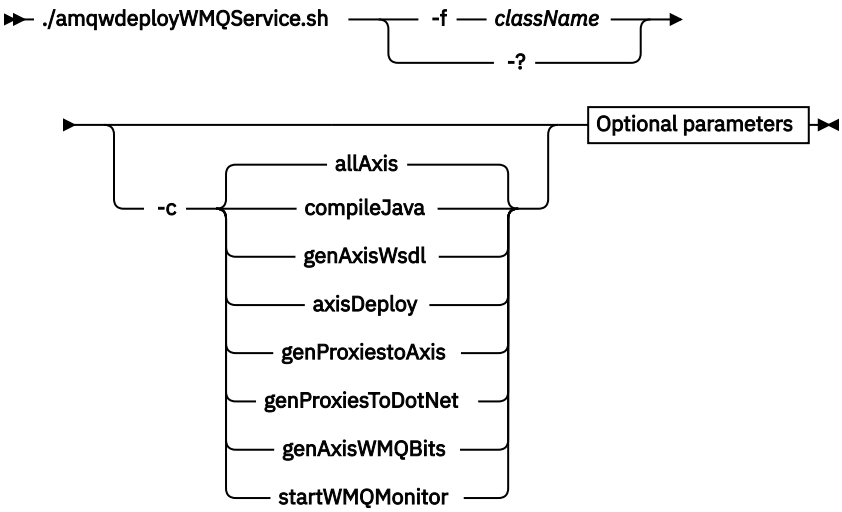
amqwdeployWMQService: utilitaire de déploiement de service Web

L'utilitaire de déploiement prépare une classe de service à utiliser comme service Web en utilisant WebSphere MQ comme transport.

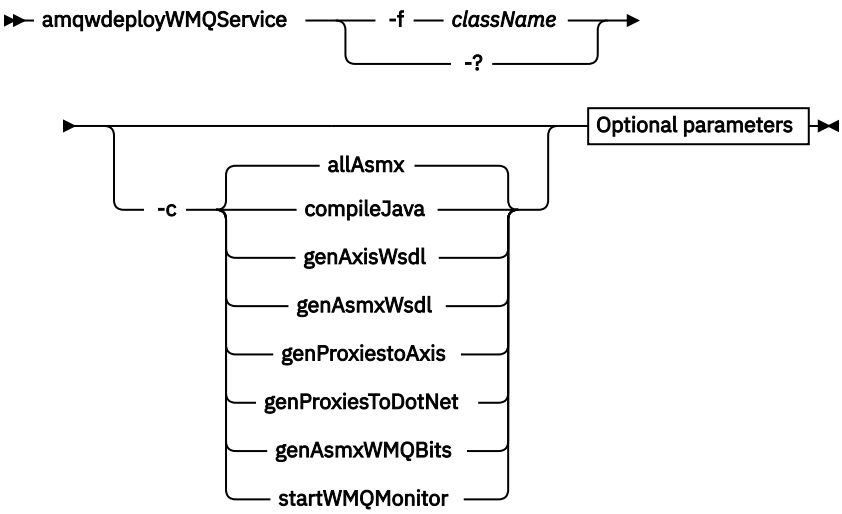
Objet

Utilisez l'utilitaire de déploiement pour générer les fichiers nécessaires au déploiement d'un service Axis 1.4, .NET Framework 1 ou .NET Framework 2. Utilisez les fichiers pour déployer un service appelé par IBM WebSphere MQ. Les fichiers générés par **amqwdeployWMQService** sont présentés dans le «Fichiers de sortie de amqwdeployWMQService», à la page 973.

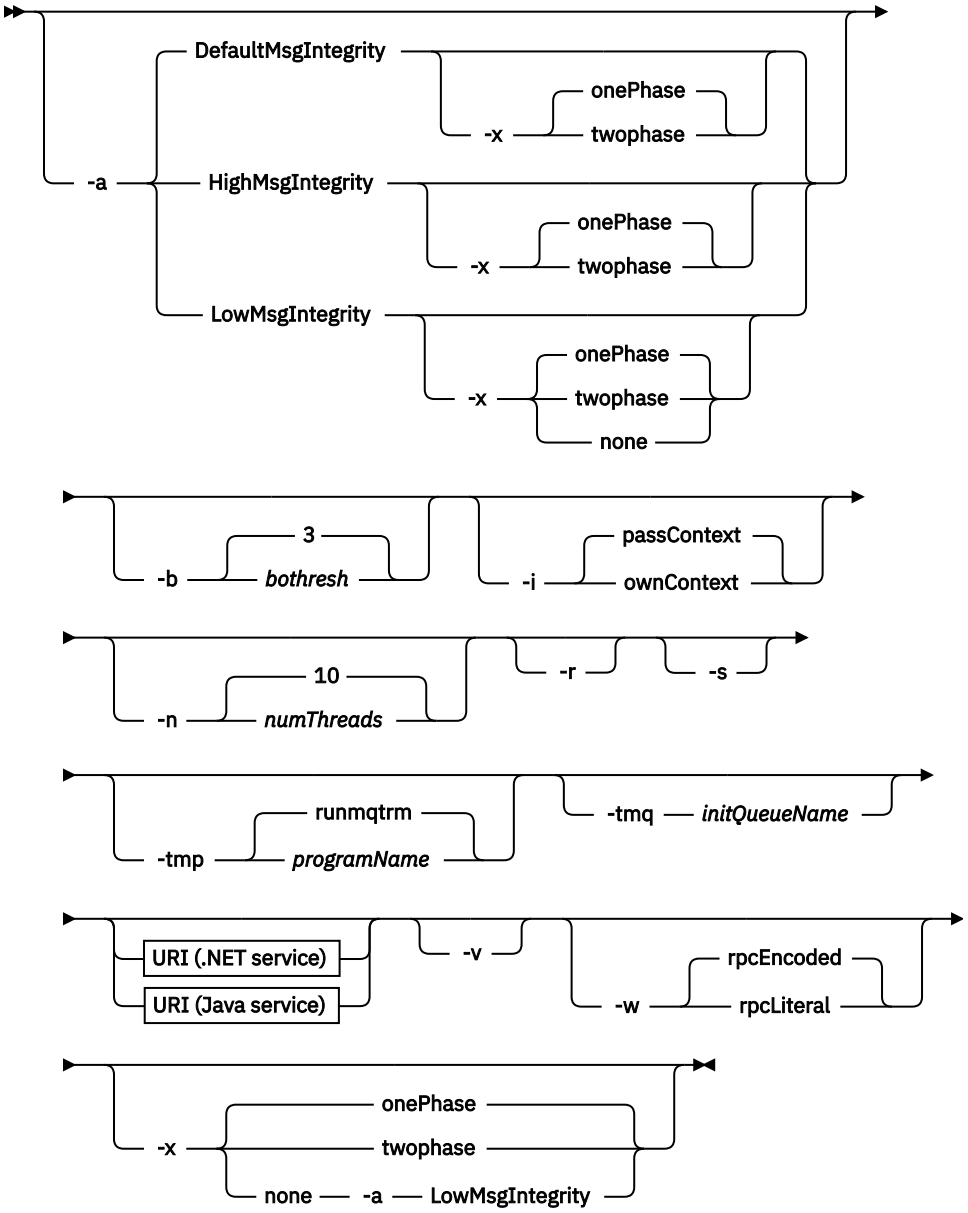
Syntax diagram
UNIX and Linux systems



Windows



Optional parameters



Paramètres obligatoires

-f ***className***
className est le nom de la classe à déployer. Pour les services Axis *className* est le fichier source Java et pour les services .NET, le fichier `.asmx` . La Figure 11, à la page 969 illustre le déploiement d'un service Axis et de Figure 12, à la page 969 d'un service .NET.

```
amqwdeployWMQService -f javaDemos/service/StockQuoteAxis.java
```

Figure 11. Exemple de déploiement du service Axis

```
amqwdeployWMQService -f StockQuoteDotNet.asmx
```

Figure 12. Exemple de déploiement du service .NET

Pour Java, *className* doit être qualifié complet par le nom du package. Il peut être spécifié en tant que nom de chemin avec des séparateurs de répertoire ou en tant que nom de classe avec des séparateurs de point. La classe générée se trouve dans `./generated/client/remote/`*`path`*

name. Pour un service .NET, bien que le répertoire puisse être spécifié, les proxys Java générés se trouvent toujours dans `./generated/client/remote/dotNetService`.

Si vous spécifiez un URI avec l'option -u et que l'URI indique *targetService*, l'utilitaire de déploiement vérifie *className*. *className* doit correspondre à *targetService*. Si la classe et le service ne correspondent pas, l'utilitaire de déploiement affiche un message d'erreur et se ferme.

-?
Imprimez le texte d'aide décrivant la façon dont la commande est utilisée.

Paramètres optionnels

-a ***integrityOption***
integrityOption indique le comportement des programmes d'écoute SOAP WebSphere MQ s'il n'est pas possible d'insérer un message de demande ayant échoué dans la file d'attente des messages non livrés. *integrityOption* peut prendre l'une des valeurs suivantes:

DefaultMsgIntegrity
Pour les messages non persistants, le programme d'écoute affiche un message d'avertissement et continue de s'exécuter avec le message d'origine en cours de suppression. Pour les messages persistants, il affiche un message d'erreur, annule le message de demande pour qu'il reste dans la file d'attente des demandes et se ferme. DefaultMsgIntegrity s'applique si l'option -a est omise ou si l'option *integrityOption* n'est pas spécifiée.

LowMsgIntegrity
Pour les messages persistants et non persistants, le programme d'écoute affiche un avertissement et continue de s'exécuter, en supprimant le message.

HighMsgIntegrity
Pour les messages persistants et non persistants, le programme d'écoute affiche un message d'erreur, annule le message de demande afin qu'il reste dans la file d'attente des demandes et se ferme.

L'utilitaire de déploiement vérifie la compatibilité des indicateurs -x et -a . Si -x none est spécifié, -a LowMsgIntegrity doit être spécifié. Si les indicateurs sont incompatibles, l'utilitaire de déploiement se ferme avec un message d'erreur et aucune étape de déploiement n'a été effectuée.

-b ***seuil***
bothresh indique le paramètre de seuil d'annulation pour la file d'attente des demandes. La valeur par défaut est 3.

-c ***opération***
operation indique la partie du processus de déploiement à exécuter. *operation* est l'une des options suivantes:

allAxis
Effectuer toutes les étapes de compilation et de configuration pour un service Axis ou Java[4].

compileJava
Compilez le service Java: `.java` dans `.class`.

genAxisWsdl
Générez le WSDL: `.class` dans `.wsdl`.

axisDeploy
Déployez le fichier de classe: `.wsdl` dans `.wsdd`, appliquez `.wsdd`.

genProxiestoAxis
Générez des proxys: `.wsdl` pour `.java` et `.class`.

genAxisWMQBits
Configurez des files d'attente IBM WebSphere MQ , des programmes d'écoute SOAP IBM WebSphere MQ et des déclencheurs pour un service Axis.

[4] Valeur par défaut si *className* a une extension .java

allAsmx
Effectuer toutes les étapes de configuration pour un service .NET[5].

genAsmxWsdl
Générez le WSDL: `.asmx` dans `.wsdl`.

genProxiesToDotNet
Générez des proxys: `.wsdl` pour `.java`, `.class`, `.cs` et `.vb`.

genAsmxWMQBits
Configuration des files d'attente IBM WebSphere MQ , des programmes d'écoute SOAP IBM WebSphere MQ et des déclencheurs

startWMQMonitor
Démarrez le moniteur de déclenchement pour les services SOAP WebSphere MQ .

Remarque : runmqtrm s'exécute sous l'ID utilisateur `mqm` . Si la sécurité est un problème, vous devez vous assurer que les programmes d'écoute sont démarrés avec les ID utilisateur appropriés.

-i *Contexte*
Contexte indique si les programmes d'écoute transmettent le contexte d'identité. *Context* prend les valeurs suivantes:

passContext
Définissez le contexte d'identité du message de demande d'origine dans le message de réponse. Le programme d'écoute SOAP vérifie qu'il est autorisé à sauvegarder le contexte à partir de la file d'attente des demandes et à le transmettre à la file d'attente des réponses. Il effectue les vérifications lors de l'exécution lors de l'ouverture de la file d'attente de demandes pour sauvegarder le contexte et de la file d'attente de réponses pour transmettre le contexte. S'il ne dispose pas des droits requis, ou si l'appel `MQOPEN` échoue et que le message de réponse n'est pas traité. Le message de réponse est inséré dans la file d'attente des messages non livrés avec l'en-tête de message non livrés contenant le code retour du `MQOPEN`ayant échoué. Le programme d'écoute continue ensuite à traiter les messages entrants suivants de manière normale.

ownContext
Le programme d'écoute SOAP ne transmet pas de contexte. Le contexte renvoyé reflète l'ID utilisateur sous lequel le programme d'écoute s'exécute plutôt que l'ID utilisateur qui a créé le message de demande d'origine.

Les zones du contexte d'origine sont définies par le gestionnaire de files d'attente et non par le programme d'écoute SOAP.

-n *numThreads*
numThreads indique le nombre d'unités d'exécution dans les scripts de démarrage générés pour le programme d'écoute SOAP WebSphere MQ . La valeur par défaut est 10. Envisagez d'augmenter ce nombre si vous disposez d'un débit de messages élevé.

-r
-r indique que les définitions de file d'attente de moniteur de demande ou de déclencheur existantes sont remplacées. Les files d'attente du moniteur de déclenchement sont remplacées uniquement si -tmq est également spécifié. Les files d'attente sont recréées avec des attributs par défaut standard et les messages existants dans les files d'attente sont supprimés. Si l'option -r n'est pas utilisée, les définitions de file d'attente existantes ne sont pas modifiées et les messages existants ne sont pas supprimés. En ne spécifiant pas -r, vous vous assurez que tous les attributs de file d'attente personnalisés sont conservés.

-s
Configurez le programme d'écoute pour qu'il s'exécute en tant que service WebSphere MQ . Si -s et -tmq sont spécifiés, l'utilitaire de déploiement affiche un message d'erreur et se ferme.

[5] Valeur par défaut si *className* possède une extension .asmx.

-tmp ***programName***
programName indique le nom d'un programme de moniteur de déclenchement. Utilisez `-tmp` *programName* dans un environnement UNIX ou Linux comme alternative à l'utilisation de **runmqtrm**. Les programmes qu'il lance sont exécutés sous le droit `mqm` .

Exemple :

```
amqwdeployWMQService -f javaDemos/service/StockQuoteAxis.java
-tmq trigger.monitor.queue -tmp trigmon
```

-tmq ***queueName***
queueName indique un nom de file d'attente de moniteur de déclenchement. Les définitions de processus IBM WebSphere MQ sont créées pour configurer le déclenchement automatique des programmes d'écoute SOAP WebSphere MQ avec le nom de file d'attente du moniteur de déclenchement associé. Si l'option n'est pas spécifiée, aucune configuration de déclenchement n'est définie par l'utilitaire de déploiement. Si -s et -tmq sont spécifiés, l'utilitaire de déploiement affiche un message d'erreur et se ferme.

URI ***plateforme***
Voir «Syntaxe d'URI et paramètres pour le déploiement de service Web», à la page 1004.

-v
-v définit la sortie prolixe des commandes externes. Les messages d'erreur sont toujours affichés. Utilisez -v pour générer des commandes que vous pouvez personnaliser pour créer des scripts de déploiement personnalisés.

-w
-w contrôle le style de WSDL à générer. La valeur par défaut est rpcEncloded, pour la compatibilité avec les versions précédentes de WebSphere MQ transport for SOAP. Utilisez rpcLiteral pour créer un WSDL compatible avec la génération de proxy client Axis2 . rpcEncoded n'est pas compatible avec les recommandations WS-I.

-x ***transactionnalité***
transactionality indique le type de contrôle transactionnel pour le programme d'écoute. *transactionality* peut être définie sur l'une des valeurs suivantes:

onePhase
La prise en charge en une phase de IBM WebSphere MQ est utilisée. Si le système échoue lors du traitement, le message de demande est redistribué à l'application. Les transactions WebSphere MQ garantissent que les messages de réponse sont écrits une seule fois.

twoPhase
La prise en charge en deux phases est utilisée. Si le service est écrit de manière appropriée, le message est distribué une seule fois, en coordination avec d'autres ressources, au sein d'une seule exécution validée du service. Cette option s'applique uniquement aux connexions de liaisons de serveur.

none
Pas de prise en charge transactionnelle. Si le système échoue lors du traitement, le message de demande peut être perdu, même s'il est persistant. Le service peut avoir ou non été exécuté et les messages de réponse, de rapport ou de rebut peuvent ou non être écrits.

L'utilitaire de déploiement vérifie la compatibilité des indicateurs -x et -a . Pour plus d'informations, voir la description de l'indicateur -a .

Erreurs

Sous Windows, si des erreurs sont signalées par **amqswsdl**, essayez d'exécuter la commande suivante pour enregistrer les fichiers `.asmx` en tant que services.

```
%windir%/Microsoft.NET/Framework/version number/aspnet_regiis.exe -ir
```

Le problème se produit généralement sur les systèmes où IIS n'a pas été installé, ou IIS a été installé après NET. Le problème se produit lorsque **amqswsdl** génère les fichiers `.wsdl` .

Remarque : Les clés de registre sont également requises pour permettre au programme d'écoute d'appeler les services. Si vous utilisez vos propres procédures de déploiement personnalisées, vous risquez de ne pas rencontrer le problème avant l'exécution.

Fichiers de sortie de amqwdeployWMQService

Liste des répertoires et des fichiers de sortie de **amqwdeployWMQService**

*Tableau 583. Fichiers de sortie de **amqwdeployWMQService***

Sorties	Description	Répertoire de sortie	Nom de fichier
.class	Fichier source Java compilé	./generated/server/*server package*	*classname*.class
.wsdl	Description du service	./generated	*classNameAxis*_Wmq.wsdl *classNameDotNet*_Wmq.wsdl
.wsdd	Fichiers de déploiement de client et de service Axis	./	client-config.wsdd server-config.wsdd
		./generated/server/*server package*	*className*_deploy.wsdd *className*_undeploy.wsdd
Source du client (.vb, .cs, .java)	Modules de remplacement de client .Net vers le service Axis	./generated/client	*classname*AxisService.cs *classname*AxisService.vb
	.Net client stubs to. Net service	./generated/client	*classname*DotNet.cs *classname*DotNet.vb

*Tableau 583. Fichiers de sortie de **amqwdeployWMQService** (suite)*

Sorties	Description	Répertoire de sortie	Nom de fichier
Auxiliaire client (.java et .class)	Proxys du client Java vers le service. Net	./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
	Proxys du client Java vers le service Axis	./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
Scripts (.cmd et .sh)	Scripts de programme d'écoute	/generated/server	startWMQJListener.cmd startWMQJListener.sh startWMQNListener.cmd endWMQJListener.cmd endWMQJListener.sh endWMQNListener.cmd

Remarques sur l'utilisation de amqwdeployWMQService

Décrit les tâches effectuées par **amqwdeployWMQService**.

L'utilitaire de déploiement effectue les actions suivantes.

1. Vérifie les chemins d'accès aux fichiers suivants:
   - axis.jar.
   - *WMQSOAP_HOME*/java/lib/com.ibm.mq.soap.jar.
   - Sous Windows, csc.exe
2. Sous Windows, utilise %SystemRoot%\Microsoft.NET\Framework\v1.1.432 ou, si le compilateur C# est installé, le chemin d'accès à csc.exe comme chemin d'accès à .NET Framework.

   Remarque : Si Microsoft Visual Studio 2008 est installé (version 9), wsdl.exe ne se trouve pas dans le chemin d'accès à csc.exe. Vous devez ajouter le chemin d'accès à .NET Framework à votre variable Path, par exemple:

   ```
   Set Path=C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727;%Path%
   ```
3. Crée le répertoire ./generated et les sous-répertoires requis, s'ils n'existent pas.
4. Pour les services Java, compile la source dans *className*.class.
5. Génère un fichier WSDL.

6. Pour les services Java, crée les fichiers de descripteur de déploiement *className*_deploy.wsdd et *className*_undeploy.wsdd
7. Pour les services Java, crée ou met à jour le fichier de descripteur de déploiement Axis, server-config.wsdd.
8. Génère les proxys client pour Java, C# et Visual Basic à partir du WSDL.

   Remarque : Sous Windows, l'utilitaire de déploiement génère des proxys pour Visual Basic et C#, quel que soit le langage dans lequel le service est écrit. Le WSDL et les proxys générés à partir de celui-ci incluent l'URI approprié pour appeler le service:

   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
   ```

   Figure 13. Exemple d'URI dans le client .NET généré pour appeler le service .NET

   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
   ```

   Figure 14. Exemple d'URI dans le client .NET généré pour appeler le service Axis 1
9. Compile les proxys Java.
10. Crée une file d'attente WebSphere MQ , *requestQueue* pour contenir les demandes du service. Le nom de file d'attente par défaut est au format SOAPJ.*directory*, ou vous pouvez spécifier *requestQueue* dans l'option d'URI -u .
11. Crée des fichiers de script de commande et de shell pour démarrer les programmes d'écoute SOAP WebSphere MQ qui traitent la file d'attente des demandes.
12. Si l'option -tmq a été utilisée, l'utilitaire de déploiement crée des définitions WebSphere MQ pour déclencher automatiquement les processus du programme d'écoute SOAP WebSphere MQ .
    - L'utilitaire de déploiement utilise l'attribut APPLICID de la commande **runmqsc** DEFINE PROCESS pour contenir une commande permettant de démarrer le programme d'écoute. La commande contient le nom du répertoire de déploiement. La zone APPLICID a une longueur maximale de 256, ce qui limite la longueur maximale du répertoire de déploiement. La limite de répertoire pour les services Java est la suivante:
      – Systèmes UNIX and Linux : 218
      – Windows: 197 moins la longueur du nom de la file d'attente des demandes.

      Pour les services .NET, la limite de répertoire est la suivante:
      – Windows: 209 moins la longueur du nom de service, moins l'extension .asmx .
    - L'utilitaire de déploiement vérifie si la limite pour APPLICID est dépassée. Si la limite est dépassée, l'utilitaire ne tente pas de définir le processus de déclenchement. Il affiche un message d'erreur et le processus de déploiement échoue sans effectuer d'étapes de déploiement.

    Les exemples suivants montrent les commandes de configuration et de démarrage générées par l'utilitaire de déploiement pour démarrer un programme d'écoute SOAP WebSphere MQ .

    ```
    DEFINE PROCESS(requestQueue) APPLICID(applicIDStr) REPLACE
    ALTER  QLOCAL (requestQueue) TRIGTYPE(FIRST) TRIGGER
           PROCESS(requestQueue) INITQ(initQueueName) TRIGMPRI(0)
    ```

    Figure 15. Commandes de configuration WebSphere MQ pour déclencher un programme d'écoute SOAP.

```
applicIDStr = start "Java WMQSoapListener -requestQueue"
                    /min .\generated\server\startWMQJListener.cmd;
```

Figure 16. Démarrage du programme d'écoute SOAP Axis sous Windows

```
applicIDStr = start "WMQAsmxListener -className\
                    /min .\generated\server\startWMQNListener.cmd;
```

Figure 17. Démarrage du programme d'écoute SOAP .NET sous Windows

```
applicIDStr = xterm -iconic -T \"Java WMQSoapListener_requestQueue\"
                   -e ./generated/server/startWMQJListener.sh & #
```

Figure 18. Démarrage du programme d'écoute SOAP Axis sur les systèmes UNIX and Linux

amqwRegisterdotNet: enregistrement du transport IBM WebSphere MQ pour SOAP vers .NET

Enregistrez le transport IBM WebSphere MQ pour SOAP dans le cache d'assemblage global sur .NET.

Objet

amqwRegisterdotNet enregistre l'émetteur SOAP, le programme d'écoute SOAP et le processeur WSDL de WebSphere MQ avec .NET Framework 1 ou 2.

Syntaxe

Description

amqwRegisterdotNet est exécuté automatiquement lors de l'installation. Vous n'avez pas besoin de l'exécuter à nouveau si .NET Framework que vous utilisez a été installé avant WebSphere MQ transport for SOAP. Vous pouvez l'exécuter autant de fois que vous le souhaitez. Utilisez-le pour réenregistrer le transport WebSphere MQ pour SOAP avec différentes versions de .NET Framework.

Remarque : Sous Windows 2003 Server, vous devez également exécuter l'utilitaire **aspnet_regiis** , même si vous ne déployez pas sur Internet Information Server (IIS). L'emplacement de l'utilitaire `aspnet_regiis.exe` peut varier selon les versions de Microsoft .NET Framework, mais il se trouve généralement dans: `%SystemRoot%/Microsoft.NET/Framework/version number/aspnet_regiis`. Si plusieurs versions sont installées, utilisez **aspnet_regiis** pour la version de .NET Framework que vous utilisez.

Licence logicielle Apache

Licence Apache Version 2.0 (janvier 2004) http://www.apache.org/licenses/

http://www.apache.org/licenses/

Licence Apache
Version 2.0, janvier 2004
http://www.apache.org/licenses/

CONDITIONS D'UTILISATION, DE REPRODUCTION ET DE DISTRIBUTION

1. Définitions.

Par "licence", on entend les conditions d'utilisation, de reproduction,
et la distribution telles que définies par les sections 1 à 9 du présent document.

"donneur de licence": le titulaire du droit d'auteur ou l'entité autorisée par
le titulaire du droit d'auteur qui accorde la licence.

Par "entité juridique", on entend l'union de l'entité qui agit et de tous les
autres entités qui contrôlent, sont contrôlées par ou sont sous contrôle commun
avec cette entité. Au sens de la présente définition, on entend par:
"contrôle": i) le pouvoir, direct ou indirect, de
la direction ou la gestion de cette entité, qu'il s'agisse d'un contrat ou
ou (ii) la propriété d'au moins cinquante pour cent (50%) des
les actions en circulation, ou iii) la propriété effective de cette entité.

"Vous" (ou "Votre") signifie un individu ou une entité juridique
exercer les droits accordés par cette licence.

On entend par forme "source" la forme privilégiée pour apporter des modifications,
y compris, mais sans s'y limiter, le code source du logiciel, la documentation
et des fichiers de configuration.

Par forme "objet", on entend toute forme résultant de
transformation ou traduction d'une forme source, y compris
non limité au code objet compilé, à la documentation générée,
et les conversions vers d'autres types de support.

Par "oeuvre", on entend l'oeuvre de la paternité, qu'elle soit à la source ou à la source.
Formulaire d'objet, mis à disposition sous la licence, comme indiqué par un
Avis de droit d'auteur inclus dans ou joint à l'oeuvre
(un exemple est fourni à l'annexe).

Par "oeuvres dérivées", on entend toute oeuvre, qu'elle soit de source ou d'objet.
qui est basée sur (ou dérivée de) l'œuvre et pour laquelle le
révisions éditoriales, annotations, élaborations ou autres modifications
représenter, dans son ensemble, une oeuvre originale de la paternité. Pour les besoins
de cette licence, les oeuvres dérivées ne doivent pas inclure les oeuvres qui restent
séparables des interfaces de, ou simplement liées (ou liées par leur nom) aux interfaces de,
l'Œuvre et les Œuvres qui en sont dérivées.

Par "contribution", on entend toute oeuvre d'auteur, y compris:
la version originale de l'Œuvre et toute modification ou ajout
à cette œuvre ou à ses oeuvres dérivées, qui est intentionnellement
soumis au concédant pour inclusion dans l'œuvre par le titulaire du droit d'auteur
ou par une personne physique ou morale autorisée à soumettre au nom de:
le titulaire du droit d'auteur. Aux fins de la présente définition, on entend par "présenté":
signifie toute forme de communication électronique, verbale ou écrite envoyée
au concédant ou à ses représentants, y compris, mais sans s'y limiter,
communication sur les listes de diffusion électroniques, systèmes de contrôle des codes source,
et des systèmes de suivi des problèmes qui sont gérés par ou pour le compte du
Donneur de licence dans le but de discuter et d'améliorer le travail, mais
à l'exclusion de la communication qui est marquée de manière visible ou autrement
désigné par écrit par le titulaire du droit d'auteur comme "Pas une contribution".

"contributeur": le concédant et toute personne physique ou morale
pour le compte de qui une contribution a été reçue par le concédant et
par la suite incorporé dans les travaux.

2. Octroi d'une licence de droit d'auteur. Sous réserve des termes et conditions de
présente Licence, chaque contributeur vous accorde par la présente un perpétuel,
mondial, non exclusif, gratuit, libre de droits, irrévocable
licence de droit d'auteur pour reproduire, préparer des oeuvres dérivées de,

affichage public, exécution publique, octroi de sous-licence et distribution de la
Travaux et oeuvres dérivées sous forme de source ou d'objet.

3. Octroi d'une licence de brevet. Sous réserve des termes et conditions de
présente Licence, chaque contributeur vous accorde par la présente un perpétuel,
mondial, non exclusif, gratuit, libre de droits, irrévocable
(sauf dans les cas indiqués dans le présent article) licence de brevet pour faire, avoir fait,
utiliser, proposer de vendre, de vendre, d'importer et de transférer d'une autre manière l'Oeuvre,
lorsque cette licence ne s'applique qu'aux revendications de brevet pouvant faire l'objet d'une licence
par un tel contributeur qui sont nécessairement enfreintes par son
Contribution (s) seule (s) ou par combinaison de leur (s) contribution (s)
avec les travaux auxquels ces contributions ont été soumises. Si vous
La Cour de justice de l'
dans une action en justice) alléguant que l'oeuvre
ou une contribution incorporée dans l'Œuvre constitue une contribution directe
ou une contrefaçon de brevet contributive, puis toute licence de brevet
accordée à Vous au titre de la présente Licence pour que cette Travail soit résiliée
à compter de la date de dépôt de ce litige.

4. Redistribution. Vous pouvez reproduire et distribuer des copies de la
Oeuvres ou oeuvres dérivées de celles-ci, quel que soit leur support, avec ou sans
modifications, et dans la forme Source ou Objet, à condition que Vous
remplissent les conditions suivantes:

(a) Vous devez donner à tout autre destinataire de l'oeuvre ou
Une copie de la présente Licence ; et

(b) Vous devez faire en sorte que tous les fichiers modifiés portent des avis bien en vue
indiquant que vous avez modifié les fichiers ; et

(c) Vous devez conserver, sous la forme Source de toute Oeuvres Dérivées
que vous distribuez, tous les droits d'auteur, les brevets, les marques et
des avis d'attribution de la forme Source de l'Œuvre,
à l'exclusion des avis qui ne concernent aucune partie de
les Oeuvres Dérivées ; et

d) Si l'oeuvre comporte un fichier texte "AVIS" faisant partie de son
distribution, toutes les Oeuvres Dérivées que Vous distribuez doivent
inclure une copie lisible des avis d'attribution contenus
dans ce fichier, à l'exclusion des avis qui ne
appartiennent à n'importe quelle partie des Oeuvres Dérivées, dans au moins une
des emplacements suivants: dans un fichier texte de type AVIS distribué
dans le cadre des Oeuvres Dérivées ; dans le formulaire Source ou
la documentation, si elle est fournie avec les Oeuvres Dérivées ; ou,
dans un affichage généré par les Dérivées Works, si et
dans tous les cas où de tels avis de tiers apparaissent normalement. Le contenu
du fichier AVIS sont à titre d'information uniquement et
ne modifiez pas la licence. Vous pouvez ajouter votre propre attribution
des avis dans les Oeuvres Dérivées que Vous distribuez, aux côtés
ou en tant qu'addendum au texte de l'AVIS provenant des travaux, fourni
que ces avis d'attribution supplémentaires ne peuvent pas être interprétés
en tant que modification de la licence.

Vous pouvez ajouter Votre propre déclaration de droits d'auteur à Vos modifications et
peut fournir des dispositions de licence supplémentaires ou différentes
pour l'utilisation, la reproduction ou la distribution de Vos modifications, ou
pour l'ensemble de ces Oeuvres Dérivées, sous réserve de Votre utilisation,
La reproduction et la distribution de l'Œuvre sont conformes aux

les conditions énoncées dans la présente licence.

5. Présentation des contributions. A moins que vous n'indiquiez explicitement le contraire,
toute Contribution intentionnellement soumise pour inclusion dans l'Œuvre
par Vous au Concédant de licence sera en vertu des termes et conditions de
cette Licence, sans dispositions supplémentaires.
Nonobstant ce qui précède, rien dans les présentes ne remplace ou ne modifie
les dispositions de tout contrat de licence distinct que vous avez éventuellement exécuté
avec le Concédant de licence concernant ces Contributions.

6. Marques. Cette licence n'accorde pas l'autorisation d'utiliser le commerce
noms, marques, marques de service ou noms de produit du concédant,
sauf dans les cas requis pour une utilisation raisonnable et coutumière dans la description
l'origine de l'œuvre et la reproduction du contenu du fichier AVIS.

7. Exclusion de garantie. Sauf si la loi applicable l'exige ou
accepté par écrit, le Concédant de licence fournit l'Œuvre (et chaque
Le contributeur fournit ses contributions) sur un BASIS "EN L'ETAT",
SANS GARANTIES OU CONDITIONS DE QUELQUE NATURE QUE CE SOIT, expresse ou
implicite, y compris, sans s'y limiter, toute garantie ou condition
TITRE, NON-CONTREFAÇON, QUALITÉ MARCHANDE ou APTITUDE À L'EXÉCUTION D'UN TRAVAIL
UN USAGE PARTICULIER. Vous êtes seul responsable de la détermination de la
l'opportunité d'utiliser ou de redistribuer l'œuvre et d'assumer toute
risques associés à l'exercice de Vos droits d'utilisation dans le cadre de cette licence.

8. Limitation de responsabilité. En aucun cas et en l'absence de théorie juridique,
en responsabilité délictuelle (y compris la négligence), contractuelle ou autre,
à moins que la loi applicable ne l'exige (par exemple, délibérément et grossièrement)
négligences) ou convenues par écrit, tout contributeur sera-t-il
Responsabilité envers vous pour les dommages, y compris tout dommage direct, indirect, spécial,
les dommages accidentels ou consécutifs de tout caractère résultant d'une
résultat de cette licence ou hors de l'utilisation ou de l'impossibilité d'utiliser la
Travaux (y compris, mais sans s'y limiter, les dommages pour perte de clientèle,
arrêt de travail, défaillance ou dysfonctionnement de l'ordinateur, ou tout
autres dommages ou pertes commerciaux), même si ce contributeur
a été informé de la possibilité de tels dommages.

9. Acceptation de la garantie ou de la responsabilité supplémentaire. Lors de la redistribution
l'Œuvre ou ses Œuvres dérivées, Vous pouvez choisir d'offrir,
et facturer des frais pour, l'acceptation du soutien, la garantie, l'indemnisation,
ou d'autres obligations de responsabilité et / ou droits compatibles avec le présent
Licence. Toutefois, en acceptant de telles obligations, Vous ne pouvez agir que
en votre nom propre et sous votre seule responsabilité, et non en votre nom
de tout autre contributeur, et uniquement si Vous acceptez d'indemniser,
défendre et protéger chaque contributeur contre toute responsabilité
a été engagée par un contributeur ou a fait l'objet d'une réclamation contre ce contributeur par une raison
de votre acceptation d'une telle garantie ou d'une responsabilité supplémentaire.

FIN DES CONDITIONS ET CONDITIONS

ANNEXE: Comment appliquer la licence Apache à votre travail.

Pour appliquer la licence Apache à votre travail, joignez les informations suivantes:
Avis de boilerplate, avec les champs entre crochets "[ ]"
remplacé par vos propres informations d'identification. (Ne pas inclure
les crochets !) Le texte doit être inclus dans le texte approprié.
syntaxe de commentaire pour le format de fichier. Nous recommandons également qu'un

le nom du fichier ou de la classe et la description de l'objectif doivent être inclus dans la même "page imprimée" que la notice de copyright pour plus facile l'identification dans les archives de tiers.

Copyright [yyyy] [nom du titulaire du droit d'auteur]

Sous licence Apache License, Version 2.0 (la "Licence") ;
vous ne pouvez pas utiliser ce fichier sauf en conformité avec la licence.
Vous pouvez obtenir une copie de la licence à l'adresse suivante:

http://www.apache.org/licenses/LICENSE-2.0

Sauf si la loi applicable l'exige ou s'il est convenu par écrit, les logiciels
distribué sous la Licence est distribué sur un BASIS "EN L'ETAT",
SANS GARANTIE NI CONDITION D'AUCUNE SORTE, expresse ou implicite.
Voir la licence pour la langue spécifique régissant les droits et
des limitations de la licence.

Paramètres SOAP MQMD

L'émetteur SOAP IBM WebSphere MQ et le programme d'écoute SOAP IBM WebSphere MQ créent un descripteur de message (**MQMD**). La rubrique décrit les zones que vous devez définir dans le MQMD si vous créez votre propre émetteur ou programme d'écoute SOAP.

Objet

Les valeurs définies dans **MQMD** contrôlent l'échange de messages entre l'expéditeur SOAP IBM WebSphere MQ , le programme d'écoute SOAP IBM WebSphere MQ et le programme client SOAP. Si vous créez votre propre émetteur ou programme d'écoute SOAP, suivez les règles de la rubrique Tableau 584, à la page 981.

Description

Tableau 584, à la page 981 décrit comment les zones **MQMD** sont définies par l'expéditeur SOAP IBM WebSphere MQ et le programme d'écoute SOAP IBM WebSphere MQ . Si vous écrivez votre propre expéditeur ou programme d'écoute, vous devez définir ces zones conformément aux règles d'échange de messages. Le programme d'écoute SOAP IBM WebSphere MQ est conforme aux protocoles d'échange de messages IBM WebSphere MQ classiques. Si vous écrivez votre propre expéditeur pour qu'il fonctionne avec les programmes d'écoute SOAP IBM WebSphere MQ , vous pouvez définir des valeurs **MQMD** différentes.

Dans Tableau 584, à la page 981, les valeurs de la colonne Paramètre sont organisées comme suit:

Demande, sens unique
Paramètres définis par l'expéditeur SOAP IBM WebSphere MQ .

Réponse, rapport
Paramètres définis par le programme d'écoute SOAP IBM WebSphere MQ en réponse à la demande de l'expéditeur SOAP IBM WebSphere MQ .

TOUT
Paramètres définis par l'émetteur SOAP IBM WebSphere MQ et le programme d'écoute SOAP IBM WebSphere MQ .

Expéditeur personnalisé
Vous pouvez écrire votre propre expéditeur. En règle générale, un expéditeur personnalisé remplace les options de rapport standard.

Tableau 584. Paramètres SOAP MQMD		
Nom de zone	**Paramètre**	**Valeurs**
StrucId	**TOUT** MQMD_STRUC_ID	'MD¬¬'[1]
Version	**TOUT** MQMD_VERSION_2	2
Report	**TOUT** MQRO_NONE + MQRO_NEW_MSG_ID + MQRO_COPY_MSG_ID_TO_CORREL_ID + MQRO_EXCEPTION + MQRO_EXPIRY + MQRO_DISCARD **Expéditeur personnalisé** Voir «Options de rapport personnalisé», à la page 985	52428800
MsgType	**Demande** MQMT_REQUEST **Réponse** MQMT_REPLY **Rapport** MQMT_REPORT **Unidirectionnel** MQMT_DATAGRAM	**MQMT_REQUEST** 1 **MQMT_REPLY** 2 **MQMT_REPORT** 4 **MQMT_DATAGRAM** 8
Expiry	**Demande, sens unique** Spécifié par l'option Expiration dans l'URI. La valeur par défaut est MQEI_UNLIMITED. **Réponse** Valeur de Expiry dans le message de demande **Rapport** MQEI_UNLIMITED	**MQEI_UNLIMITED** -1

Tableau 584. Paramètres SOAP MQMD (suite)		
Nom de zone	**Paramètre**	**Valeurs**
Feedback	**Demande, réponse, sens unique** MQFB_NONE. **Rapport** • Généré par le gestionnaire de files d'attente-valeur définie conformément aux règles normales. • Généré par le programme d'écoute SOAP IBM WebSphere MQ : **MQRC_BACKOUT_THRESHOLD_REACHED** Seuil d'annulation dépassé pour les tentatives multiples. **MQRCCF_MD_FORMAT_ERROR** Le message n'est pas reconnu comme ayant un en-tête MQRFH2 . **MQRC_RFH_PARM_MISSING** Un paramètre obligatoire, par exemple, SoapAction, dans **MQRFH2** est manquant. **MQRC_RFH_FORMAT_ERROR** Une vérification d'intégrité de base de **MQRFH2** a échoué. Par exemple, les longueurs internes sont endommagées. **MQRC_RFH_ERROR** Le **MQRFH2** a passé une vérification d'intégrité, mais le corps du message n'est pas défini sur 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	**TOUT** MQENC_NATIVE	Dépend de l'environnement
CodedCharSetId	**TOUT** Défini sur UTF-8	1208
Format	**Demande, réponse, sens unique** MQFMT_RF_HEADER_2 **Rapport** **Rapports du gestionnaire de files d'attente** Suit les règles IBM WebSphere MQ **Rapports du programme d'écoute SOAP IBM WebSphere MQ** Format du message de demande d'origine.	**MQFMT_RF_HEADER_2** "MQRFH2 "

Tableau 584. Paramètres SOAP MQMD (suite)		
Nom de zone	**Paramètre**	**Valeurs**
Priority	**Demande, sens unique** Spécifié par l'option Priorité dans l'URI. La valeur par défaut est MQPRI_PRIORITY_AS_Q_DEF. **Réponse, rapport** Valeur de Priorité dans le message de demande.	**MQPRI_PRIORITY_AS_Q_DEF** -1
Persistence	**Demande, sens unique** MQPER_PERSISTENCE_AS_Q_DEF. **Réponse, rapport** Valeur de Persistence dans le message de demande.	**MQPER_PERSISTENCE_AS_Q_DEF** 2
MsgId	**Demande, sens unique** Généré par le gestionnaire de files d'attente. **Réponse, rapport** Les IBM WebSphere MQ ensembles d'expéditeurs SOAP MQRO_NEW_MSG_ID et *MsgId* sont générés	**Généré** Valeur unique générée par le gestionnaire de files d'attente
CorrelId	**Demande, sens unique, Rapport** MQCI_NONE **Réponse, rapport** L' IBM WebSphere MQ émetteur SOAP définit MQRO_COPY_MSG_ID_TO_CORREL_ID et le programme d'écoute copie *MsgId* à partir du message de demande.	**MQCI_NONE** 0
BackoutCount	**TOUT** Inutilisé	0
ReplyToQ	**Demande** Spécifié par l'option replyDestination dans l'URI. La valeur par défaut est SYSTEM.SOAP.RESPONSE.QUEUE. **Réponse, sens unique, rapport** Vide à gauche	
ReplyToQMgr	**TOUT** Zone laissée vide	Généré par le gestionnaire de files d'attente ; voir File d'attente de réponse et gestionnaire de files d'attente .

Tableau 584. Paramètres SOAP MQMD (suite)		
Nom de zone	**Paramètre**	**Valeurs**
UserIdentifier	**Demande, sens unique, Rapport** Vide à gauche **Réponse** Dépend de l'option -i *passContext* fournie au programme d'écoute et des droits d'accès sous lequel le programme d'écoute s'exécute.	**Demande, sens unique, Rapport** Généré par le gestionnaire de files d'attente ; voir «UserIdentifier (MQCHAR12)», à la page 441 **Réponse** *Variable*
AccountingToken	**TOUT** MQACT_NONE	**MQACT_NONE** Chaîne nulle ou blancs Défini par le gestionnaire de files d'attente ; voir «AccountingToken (MQBYTE32)», à la page 398
ApplIdentityData	**TOUT** Aucun	Chaîne nulle ou blancs[2]
PutApplType	**TOUT** MQAT_NO_CONTEXT	**MQAT_NO_CONTEXT** 0 Valeur générée par le gestionnaire de files d'attente ; voir «Type PutAppl(MQLONG)», à la page 427.
PutApplName	**TOUT** Aucun	Valeur générée par le gestionnaire de files d'attente ; voir «PutApplNom (MQCHAR28)», à la page 426.
PutDate	**TOUT** Aucun	Valeur générée par le gestionnaire de files d'attente ; voir «PutDate (MQCHAR8)», à la page 429.
PutTime	**TOUT** Aucun	Valeur générée par le gestionnaire de files d'attente ; voir «PutTime (MQCHAR8)», à la page 429.
ApplOriginData	**TOUT** Aucun	Chaîne nulle ou blancs[2]
GroupId	**Demande, sens unique, Rapport** MQGI_NONE **Réponse** La zone est copiée à partir du message de demande	Valeurs NULL

Tableau 584. Paramètres SOAP MQMD (suite)		
Nom de zone	**Paramètre**	**Valeurs**
MsgSeqNumber	**Demande, sens unique, Rapport** Inutilisé **Réponse** La zone est copiée à partir du message de demande	Généré par le gestionnaire de files d'attente ; voir Ordre physique d'une file d'attente.
Offset	**Demande, sens unique, Rapport** Inutilisé **Réponse** La zone est copiée à partir du message de demande	0
MsgFlags	**Demande, sens unique, Rapport** MQMF_NONE **Réponse** La zone est copiée à partir du message de demande	**MQMF_NONE** 0 Voir «MsgFlags (MQLONG)», à la page 415
OriginalLength	**Demande, sens unique, Réponse** MQOL_UNDEFINED **Rapport** Longueur du message de demande d'origine	**MQOL_UNDEFINED** -1
Remarques : 1. Le symbole ¬ représente un caractère blanc unique. 2. La valeur Chaîne nulle ou blancs indique la chaîne nulle en C et les caractères blancs dans les autres langages de programmation.		

Options de rapport personnalisé

Vous pouvez écrire votre propre expéditeur SOAP et l'utiliser avec les programmes d'écoute fournis. En règle générale, vous pouvez écrire un expéditeur pour modifier le choix des options de rapport. Les programmes d'écoute SOAP IBM WebSphere MQ prennent en charge la plupart des combinaisons d'options de rapport, comme décrit dans les listes suivantes.

- Options de rapport prises en charge par les programmes d'écoute SOAP IBM WebSphere MQ :
  - 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
- Options de rapport prises en charge par le gestionnaire de files d'attente:

  - 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
- Les options de rapport suivantes ne sont pas prises en charge par les programmes d'écoute SOAP IBM WebSphere MQ .
  - MQRO_PAN
  - MQRO_NAN

Le comportement des programmes d'écoute SOAP IBM WebSphere MQ en réponse à des combinaisons de MQRO_EXCEPTION_* et MQRO_DISCARD est décrit dans Tableau 585, à la page 986.

La notation MQRO_EXCEPTION_* indique l'utilisation de MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA ou MQRO_EXCEPTION_WITH_FULL_DATA.

Tableau 585. Comportement du programme d'écoute résultant des paramètres MQRO_EXCEPTION_* et MQRO_DISCARD

	MQRO_DISCARD SOCKS	**MQRO_DISCARD Non activé**
MQRO_EXCEPTION_* SOCKS	Comportement par défaut. Les messages de rapport sont générés automatiquement si nécessaire et la demande d'origine est supprimée. Si un message de rapport n'a pas pu être renvoyé à la file d'attente des réponses, le message de rapport est envoyé à la file d'attente des messages non livrés.	Les messages de rapport sont générés automatiquement si nécessaire et le message d'origine est envoyé à la file d'attente de rebut. Si le message de rapport n'a pas pu être renvoyé à la file d'attente de réponses, il est également envoyé à la file d'attente de rebut. Dans ce cas, il existe donc deux entrées de file d'attente de rebut pour la demande ayant échoué.
MQRO_EXCEPTION_* Non activé	Les messages de rapport ne sont pas générés automatiquement lorsque le format entrant n'est pas reconnu ou que le nombre de tentatives d'annulation est dépassé. Le message n'est pas envoyé à la file d'attente de rebut. Aucune notification n'est renvoyée que le client peut inspecter et le message de demande d'origine est perdu.	Les messages de rapport ne sont pas générés automatiquement lorsque le format entrant n'est pas reconnu ou que le nombre de tentatives d'annulation est dépassé. Le message de demande d'origine est cependant écrit dans la file d'attente des messages non livrés lorsqu'un rapport aurait sinon été généré.

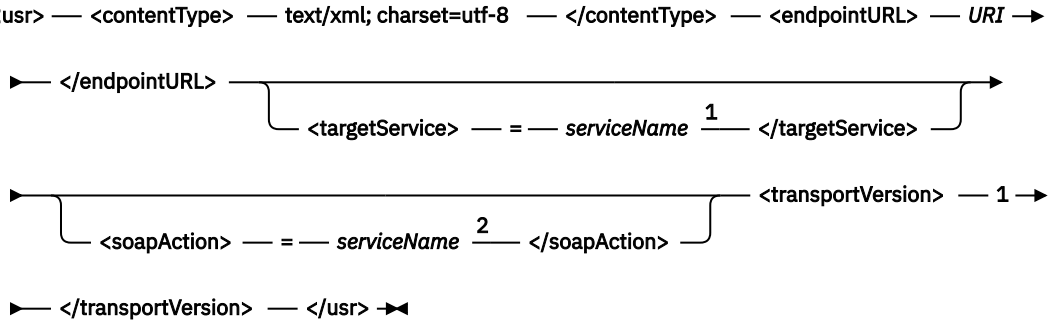
Paramètres SOAP MQRFH2

Les émetteurs et les programmes d'écoute SOAP IBM WebSphere MQ créent ou s'attendent à recevoir un MQRFH2 avec les paramètres suivants.

Objet

Les émetteurs SOAP WebSphere MQ ajoutent des propriétés au dossier <usr> créé par WebSphere MQ JMS. Les propriétés contiennent les informations requises par le conteneur SOAP dans l'environnement cible. Le «Syntaxe de la propriété», à la page 987 décrit la syntaxe des propriétés lorsqu'elles sont ajoutées à un MQRFH2. Pour la description d'un en-tête MQRFH2 , voir MQRFH2 -Règles et formatage de l'en-tête 2.

Syntaxe de la propriété



Remarques :

[1] targetService est requis pour .NET Framework 1 ou 2 et n'est pas utilisé sur Axis 1.4.
[2] soapAction est facultatif pour .NET Framework 1 ou 2 et n'est pas utilisé sur Axis 1.4.

Paramètres

contentType
contentType contient toujours la chaîne `text/xml; charset=utf-8`.

endpointURL
Voir «Syntaxe d'URI et paramètres pour le déploiement de service Web», à la page 1004.

targetService
[6]Sur l'axe, *serviceName* est le nom qualifié complet d'un service Java , par exemple: `targetService=javaDemos.service.StockQuoteAxis`. Si targetService n'est pas spécifié, un service est chargé à l'aide du mécanisme Axis par défaut.

[7]Sur .NET, *serviceName* est le nom d'un service .NET situé dans le répertoire de déploiement, par exemple: `targetService=myService.asmx`. Dans l'environnement .NET, le paramètre targetService permet à un seul programme d'écoute SOAP WebSphere MQ de pouvoir traiter des demandes pour plusieurs services. Ces services doivent être déployés à partir du même répertoire.

soapAction

transportVersion
transportVersion est toujours défini sur 1.

Exemple

L'exemple illustre un MQRFH2 et le message SOAP suivant. Les longueurs des dossiers sont affichées sous forme décimale.

Remarque : & dans l'URI est codé en tant que &

```
52464820 00000002 000002B0 00000001 RFH 0002 1208 0001
000004B8 20202020 20202020 00000000 1208 0000
000004B8                           1208
32  <mcd>
        <Msd>jms_bytes</Msd>
```

[6] Service Java uniquement
[7] Service .NET uniquement

```
    </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 transport for SOAP installation verification test

Une suite de tests de vérification d'installation (IVT) est fournie avec le transport IBM WebSphere MQ pour SOAP. **runivt** exécute un certain nombre d'applications de démonstration et s'assure que l'environnement est correctement configuré après l'installation.

Objet

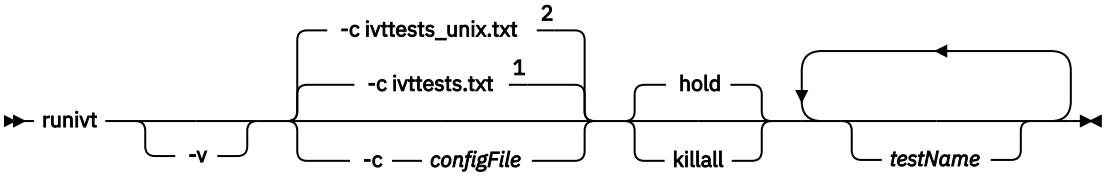
La commande **runivt** utilise les exemples de programmes fournis avec le transport WebSphere MQ for SOAP pour envoyer des demandes de service Web des clients aux services. Il exécute des tests pour Axis 1.4, .NET Framework 1 et .NET Framework 2. Les tests sont configurés dans un fichier script de test. Le fichier script de test par défaut pour Windows exécute une combinaison de tests entre les clients et les services Java et .NET.

Description

runivt doit être exécuté à partir de son propre répertoire.

La commande démarre les programmes d'écoute dans une autre fenêtre de commande. Pour cette raison, vous devez exécuter la commande à partir d'une session X Window System sur les systèmes UNIX and Linux .

runivt syntax



Remarques :

[1] Default on Windows

[2] Default on UNIX and Linux systems

Paramètres de `runivt`

`-v`
Mode prolixe. Ecrivez des messages d'erreur plus complets sur la console.

`-c` ***configFile***
Fichier de configuration définissant les tests à exécuter. Le fichier de configuration par défaut fourni avec les systèmes Windows, UNIX ou Linux est utilisé par défaut.

`hold`
Laisser le programme d'écoute en cours d'exécution une fois les tests terminés

`killall`
Arrêter le programme d'écoute une fois les tests terminés

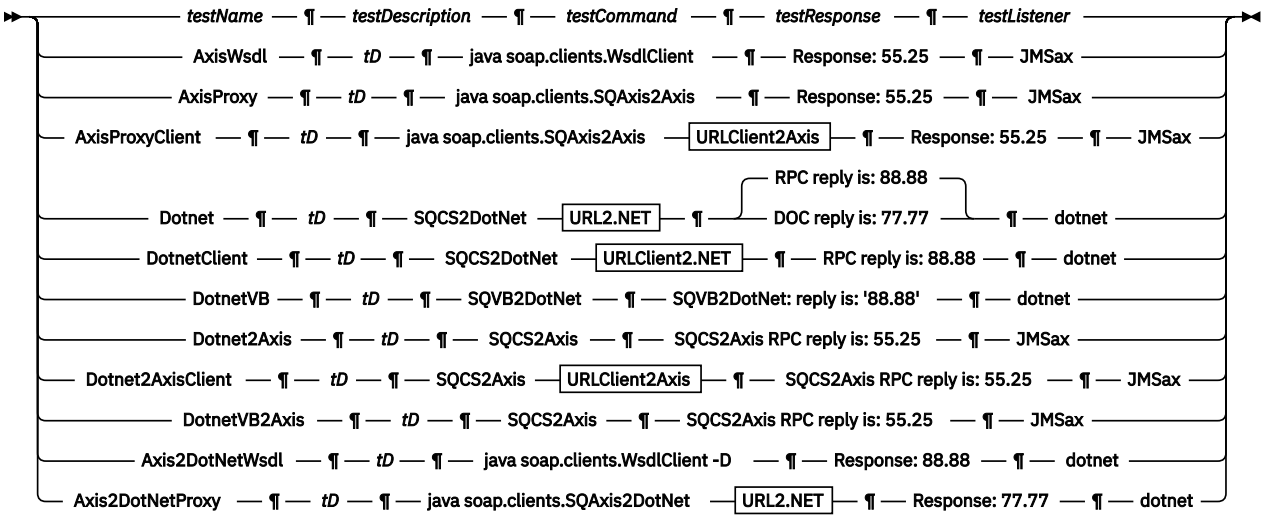
testName
Liste séparée par des espaces des tests à exécuter. Les noms de test sont sélectionnés dans le fichier de configuration. Si aucun nom n'est spécifié, tous les tests du fichier de configuration sont exécutés.

Configuration file

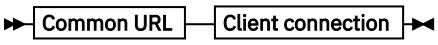
Each configuration file parameter is a separate line of the file. Leave a blank line between each group of parameters.

The parameters in the `ivttests.txt` parameter file are listed.

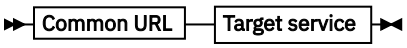
configFile **syntax**



URLClient2Axis

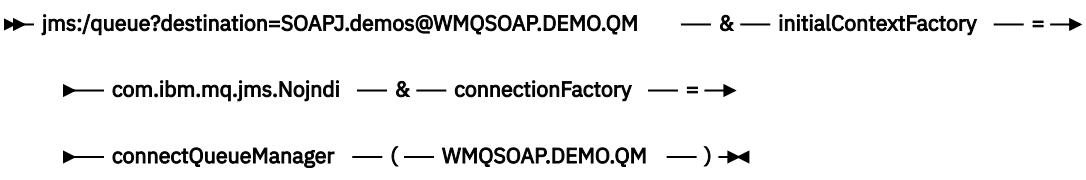


URL2.NET

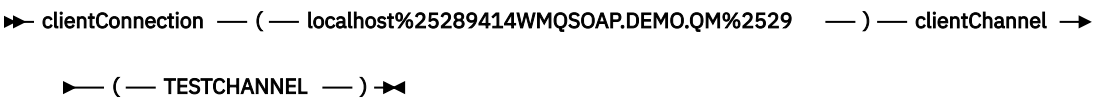


URLClient2.NET

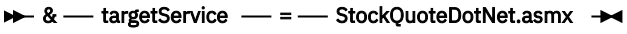
Common URL



Client connection



Target service



Paramètres de *configFile*

testName
Nom du test. Utilisez *testName* dans la commande **runivt**

testDescription
Documentation sur le test

testCommand
Commande exécutée par la commande **runivt** pour effectuer la demande du client.

testResponse
Chaîne de réponse exacte renvoyée par la demande du client à la console. Pour que le test réussisse, *testResponse* doit correspondre à la réponse réelle.

testListener
Nom du programme d'écoute SOAP WebSphere MQ démarré par **runivt** pour traiter la demande SOAP. dotnet et JMSax sont synonymes des programmes d'écoute fournis, **amqwSOAPNETlistener** et **SimpleJavaListener**.

Exemples

```
runivt
```

Figure 19. exécuter tous les tests par défaut

```
runivt dotnet
```

Figure 20. exécuter un test spécifique à partir des tests par défaut

```
runivt -c mytests.txt
```

Figure 21. exécuter un ensemble de tests personnalisés

Information associée
Vérification du transport WebSphere MQ pour SOAP

Sécurisation des services Web via le transport IBM WebSphere MQ pour SOAP

Vous pouvez sécuriser les services Web qui utilisent le transport IBM WebSphere MQ pour SOAP de deux manières. Créez un canal SSL entre le client et le serveur ou utilisez la sécurité des services Web.

SSL et le transport WebSphere MQ pour SOAP

Le transport WebSphere MQ pour SOAP fournit plusieurs options SSL qui peuvent être spécifiées pour être utilisées avec le canal client configuré pour s'exécuter en mode SSL. Les options diffèrent entre les environnements .NET et Java. Les émetteurs et les programmes d'écoute SOAP WebSphere MQ ne traitent que les options SSL applicables à leur environnement particulier. Ils ignorent les options qui ne sont pas applicables.

La présence ou l'absence de l'option `sslCipherSpec` pour les clients .NET et de l'option `sslCipherSuite` pour les clients Java détermine si SSL est utilisé ou non. Si l'option n'est pas spécifiée dans l'URI, SSL par défaut n'est pas utilisé et toutes les autres options SSL sont ignorées. Toutes les options SSL sont facultatives, sauf indication contraire.

Pour les clients WebSphere MQ , définissez les attributs SSL dans l'URI ou la table de définition de canal. Sur le serveur, définissez les attributs à l'aide des fonctions de WebSphere MQ.

Par défaut, l'option SSL WebSphere MQ standard, `SSLCAUTH`, est définie lors de l'activation de SSL sur le canal. Les clients doivent s'authentifier pour que la communication SSL puisse commencer. Si `SSLCAUTH` n'est pas défini, les communications SSL sont établies sans authentification client.

Pour s'authentifier, les clients doivent avoir un certificat affecté dans leur référentiel de clés qui est acceptable pour le gestionnaire de files d'attente. Pour plus de sécurité, les canaux WebSphere MQ peuvent être configurés pour accepter uniquement les certificats d'une liste restreinte. La liste est restreinte en vérifiant le nom distinctif du certificat par rapport à l'attribut de nom d'homologue du canal.

Si vous utilisez Java, la première connexion SSL à partir d'un client SOAP WebSphere MQ entraîne la correction des paramètres SSL suivants. Les mêmes valeurs sont utilisées dans les connexions suivantes à l'aide du même processus client:

- `MagasinsslKey`
- `sslKeyStorePassword`
- `magasinsslTrust`
- `sslTrustStorePassword`
- `SSLFIPSREQUIRED`
- `sslLDAPCRLservers`

L'effet de la modification de ces paramètres sur les connexions ultérieures à partir de ce client n'est pas défini.

Si vous utilisez .NET, la première connexion SSL à partir d'un client SOAP WebSphere MQ entraîne la correction des paramètres SSL suivants. Les mêmes valeurs sont utilisées dans les connexions suivantes à l'aide du même processus client:

- `SSLKeyRepository`
- `SSLCryptoHardware`
- `SSLFIPSREQUIRED`
- `sslLDAPCRLservers`

L'effet de la modification de ces paramètres sur les connexions ultérieures à partir de ce client n'est pas défini. Ces paramètres sont réinitialisés si toutes les connexions SSL deviennent inactives et qu'une nouvelle connexion SSL est établie.

Les propriétés suivantes peuvent également être spécifiées en tant que propriétés système:

- `MagasinsslKey`
- `sslKeyStorePassword`
- `magasinsslTrust`
- `sslTrustStorePassword`

S'ils sont spécifiés à la fois en tant que propriétés système et dans l'URI, et que les valeurs sont différentes, l'utilitaire de déploiement affiche un avertissement. Les valeurs d'URI sont prioritaires.

Tâches associées

Comment indiquer que seuls les CipherSpecs certifiés FIPS sont utilisés lors de l'exécution sur MQI Client

Référence associée

Paramètres de la fabrique de connexions SSL dans l'URI des services Web WebSphere MQ

Ajoutez des options SSL à la liste des options de fabrique de connexions dans l'URI des services Web IBM WebSphere MQ .

Paramètres de la fabrique de connexions SSL dans l'URI des services Web WebSphere MQ

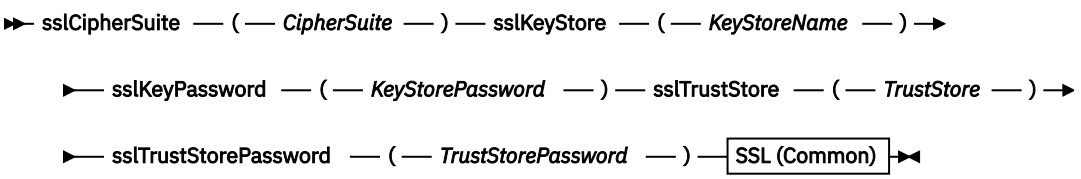
Ajoutez des options SSL à la liste des options de fabrique de connexions dans l'URI des services Web IBM WebSphere MQ .

Objet

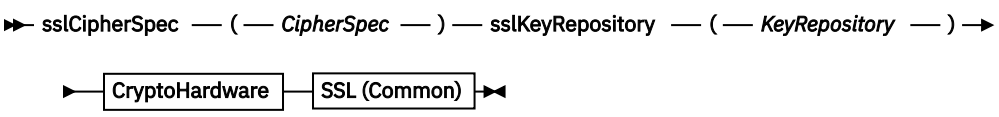
Vous pouvez utiliser une connexion sécurisée entre un client de services Web IBM WebSphere MQ et le gestionnaire de files d'attente qui héberge le service Web. Les options SSL contrôlent la façon dont SSL est configuré sur la connexion de canal client-serveur IBM WebSphere MQ MQI.

Syntax diagram

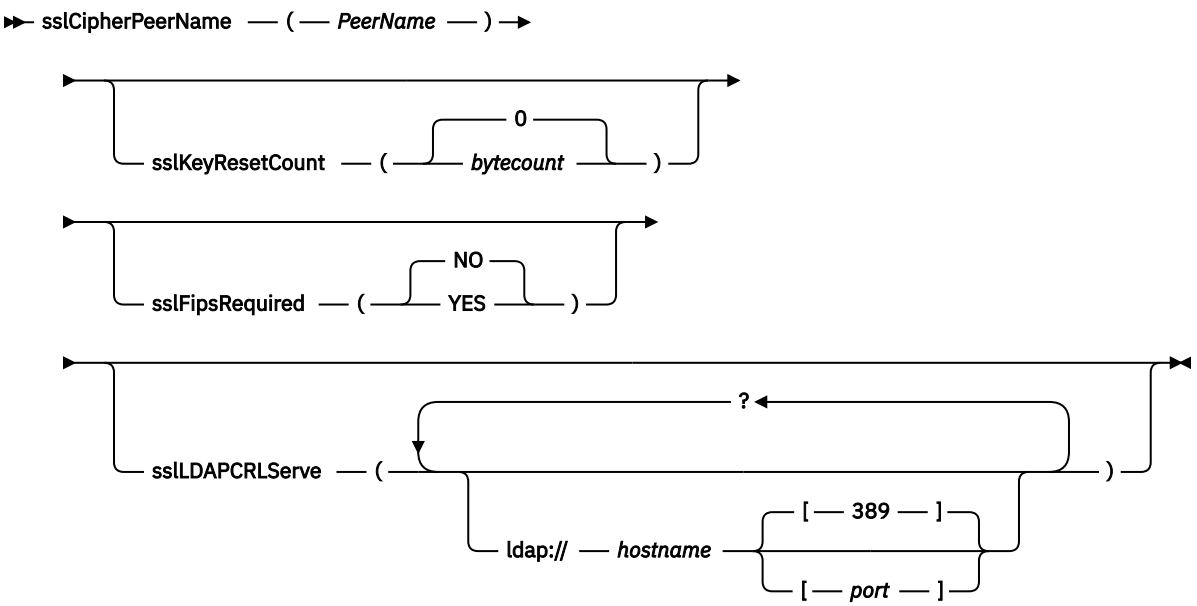
SSL (Java)



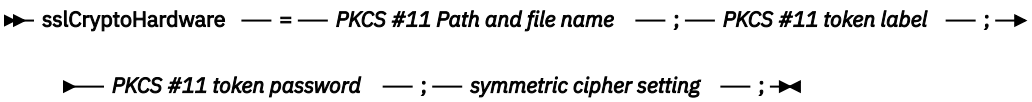
SSL (.NET)



SSL (Common)



CryptoHardware



Paramètres SSL requis (communs)

sslPeerName(*peerName*)
peerName indique le sslPeerNom utilisé sur le canal.

Paramètres SSL requis (Java)

sslCipherSuite(*CipherSuite*)
CipherSuite indique la suite sslCipher utilisée sur le canal. La suite de chiffrement CipherSuite spécifiée par le client doit correspondre à la suite de chiffrement CipherSuite spécifiée sur le canal de connexion serveur.

sslKeyStore(*KeyStoreNom*)
KeyStoreName indique la sslKeyStoreName utilisée sur le canal. Le magasin de clés contient la clé privée du client utilisée pour authentifier le client auprès du serveur. Le magasin de clés est facultatif si la connexion SSL est configurée pour accepter les connexions client anonymes.

sslKeyStorePassword(*KeyStoreMot de passe*)
KeyStorePassword indique le sslKeyStorePassword utilisé sur le canal.

sslTrustStore(*TrustStoreNom*)
TrustStore indique le sslTrustStoreName utilisé sur le canal. Le magasin de clés de confiance contient le certificat public du serveur, ou sa chaîne de clés, pour authentifier le serveur auprès du client. Le magasin de clés de confiance est facultatif si le certificat racine d'une autorité de certification est utilisé pour authentifier le serveur. Dans Java, les certificats racine sont stockés dans le magasin de certificats JRE, `cacerts`.

sslTrustStorePassword(*TrustStoreMot de passe*)
TrustStorePassword indique le sslTrustStorePassword utilisé sur le canal.

Paramètres SSL requis (.NET)

sslCipherSpec(*CipherSpec*)
CipherSpec indique la spécification sslCipherSpec utilisée sur le canal. Si l'option est spécifiée, SSL est utilisé sur le canal client.

sslKeyRepository(*KeyRepository*)
KeyRepository spécifie la spécificationsslCipher utilisée sur le canal où les clés SSL et les certificats sont stockés. *KeyRepository* est spécifié au format de radical, c'est-à-dire un chemin d'accès complet avec le nom de fichier mais avec l'extension de fichier omise. La définition du référentielsslKey a le même effet que la définition de la zone `KeyRepository` dans la structure **MQSCO** sur un appel `MQCONNX` .

Paramètres SSL facultatifs (.NET)

sslCryptoHardware(*CryptoHardware*)
CryptoHardware spécifie le matérielsslCrypto utilisé sur le canal. Les valeurs possibles pour cette zone et l'effet de sa définition sont les mêmes que pour la zone `CryptoHardware` de la structure **MQSCO** sur un `MQCONNX`.

Paramètres SSL facultatifs (communs)

sslKeyResetCount(*bytecount*)
bytecount indique le nombre d'octets transmis via un canal SSL avant que la clé secrète SSL ne doive être renégociée. Pour désactiver la renégociation des clés SSL, omettez la zone ou définissez-la sur zéro. Zéro est la seule valeur prise en charge dans certains environnements. Voir Renégociation de la clé secrète dans WebSphere MQ classes for Java. La définition de sslKeyResetCount a le même effet que la définition de la zone `KeyResetCount` dans la structure **MQSCO** sur un appel `MQCONNX` .

sslFipsRequired(*fipsCertified*)
fipsCertified indique si *CipherSpec* ou *CipherSuite* doit utiliser la cryptographie certifiée FIPS dans IBM WebSphere MQ sur le canal. L'effet de la définition de *fipsCertified* est identique à la définition de la zone `FipsRequired` de la structure **MQSCO** sur un appel `MQCONNX` .

sslLDAPCRLServers(*LDAPServerList*)

LDAPServerList indique une liste de serveurs LDAP à utiliser pour la vérification de la liste de révocation de certificat.

Pour les connexions client SSL, *LDAPServerList* est une liste de serveurs LDAP à utiliser pour la vérification de la liste de révocation de certificat (CRL). Le certificat fourni par le gestionnaire de files d'attente est vérifié par rapport à l'un des serveurs CRL LDAP répertoriés ; s'il est trouvé, la connexion échoue. Chaque serveur LDAP est essayé à son tour jusqu'à ce que la connectivité soit établie avec l'un d'entre eux. S'il est impossible de se connecter à l'un des serveurs, le certificat est rejeté. Une fois qu'une connexion a été établie avec succès avec l'un d'entre eux, le certificat est accepté ou rejeté en fonction des listes de révocation de certificat présentes sur ce serveur LDAP.

Si *LDAPServerList* est vide, le certificat appartenant au gestionnaire de files d'attente n'est pas vérifié par rapport à une liste de révocation de certificat. Un message d'erreur s'affiche si la liste d'URI LDAP fournie n'est pas valide. La définition de cette zone a le même effet que l'inclusion d'enregistrements `MQAIR` et l'accès à partir d'une structure **MQSCO** sur un `MQCONNX`.

Tâches associées

Comment indiquer que seuls les CipherSpecs certifiés FIPS sont utilisés lors de l'exécution sur MQI Client

Référence associée

SSL et le transport WebSphere MQ pour SOAP

Le transport WebSphere MQ pour SOAP fournit plusieurs options SSL qui peuvent être spécifiées pour être utilisées avec le canal client configuré pour s'exécuter en mode SSL. Les options diffèrent entre les environnements .NET et Java. Les émetteurs et les programmes d'écoute SOAP WebSphere MQ ne traitent que les options SSL applicables à leur environnement particulier. Ils ignorent les options qui ne sont pas applicables.

FIPS (Federal Information Processing Standards) pour UNIX, Linux et Windows

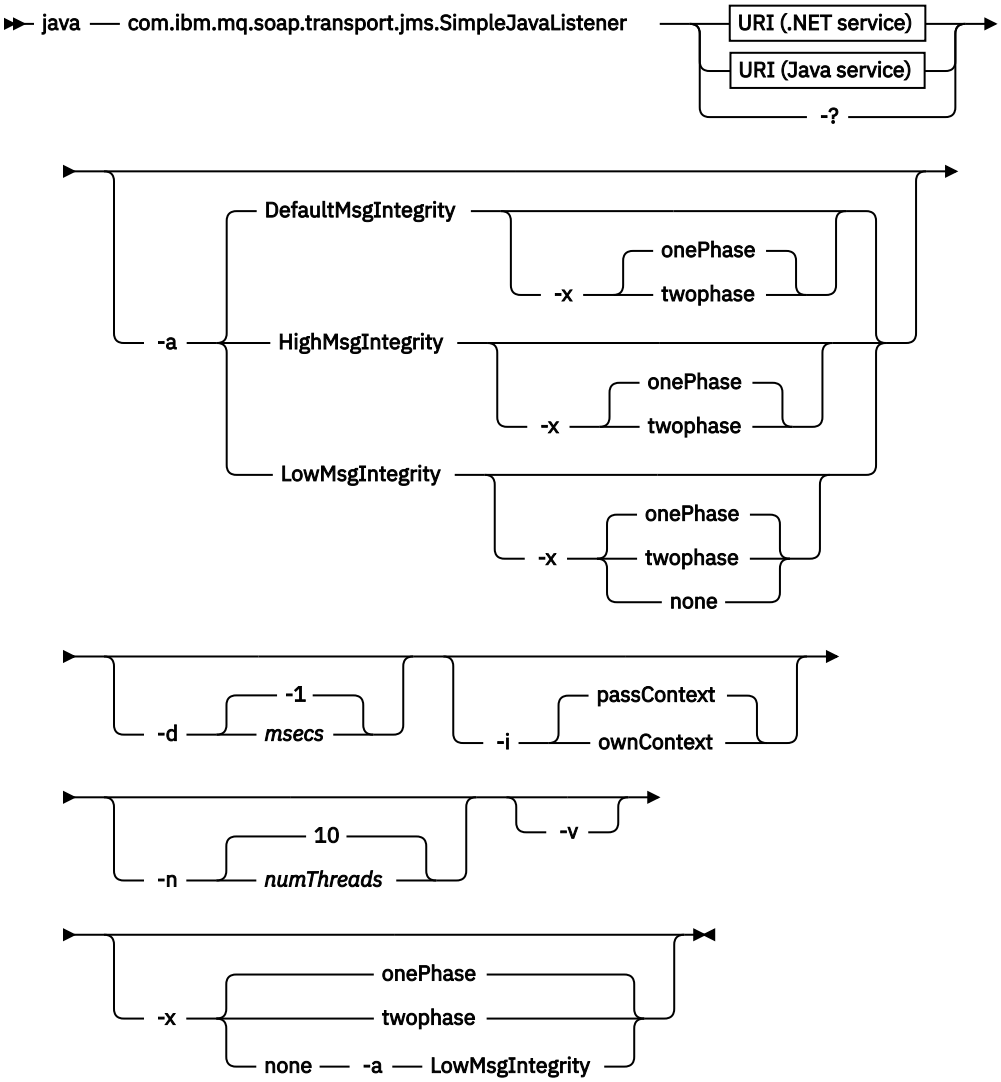
`SimpleJavaListener`: IBM WebSphere MQ Programme d'écoute SOAP pour Axis 1.4

Syntaxe et paramètres du programme d'écoute SOAP IBM WebSphere MQ pour Axis 1.4.

Objet

Démarre le programme d'écoute SOAP IBM WebSphere MQ pour Axis 1.4.

Java



Paramètres obligatoires

URI ***plateforme***
Voir «Syntaxe d'URI et paramètres pour le déploiement de service Web», à la page 1004.

-?
Imprimez le texte d'aide décrivant la façon dont la commande est utilisée.

Paramètres optionnels

-a ***integrityOption***
integrityOption indique le comportement des programmes d'écoute SOAP WebSphere MQ s'il n'est pas possible d'insérer un message de demande ayant échoué dans la file d'attente des messages non livrés. *integrityOption* peut prendre l'une des valeurs suivantes:

DefaultMsgIntegrity
Pour les messages non persistants, le programme d'écoute affiche un message d'avertissement et continue de s'exécuter avec le message d'origine en cours de suppression. Pour les messages persistants, il affiche un message d'erreur, annule le message de demande pour qu'il reste dans la file d'attente des demandes et se ferme. DefaultMsgIntegrity s'applique si l'option -a est omise ou si l'option *integrityOption* n'est pas spécifiée.

LowMsgIntegrity
Pour les messages persistants et non persistants, le programme d'écoute affiche un avertissement et continue de s'exécuter, en supprimant le message.

HighMsgIntegrity
Pour les messages persistants et non persistants, le programme d'écoute affiche un message d'erreur, annule le message de demande afin qu'il reste dans la file d'attente des demandes et se ferme.

L'utilitaire de déploiement vérifie la compatibilité des indicateurs -x et -a . Si -x none est spécifié, -a LowMsgIntegrity doit être spécifié. Si les indicateurs sont incompatibles, l'utilitaire de déploiement se ferme avec un message d'erreur et aucune étape de déploiement n'a été effectuée.

-d ***ms***
msecs indique le nombre de millisecondes pendant lequel le programme d'écoute SOAP WebSphere MQ doit rester actif si des messages de demande ont été reçus sur une unité d'exécution. Si *msecs* est défini sur -1, le programme d'écoute reste actif indéfiniment.

-i ***Contexte***
Contexte indique si les programmes d'écoute transmettent le contexte d'identité. *Context* prend les valeurs suivantes:

passContext
Définissez le contexte d'identité du message de demande d'origine dans le message de réponse. Le programme d'écoute SOAP vérifie qu'il est autorisé à sauvegarder le contexte à partir de la file d'attente des demandes et à le transmettre à la file d'attente des réponses. Il effectue les vérifications lors de l'exécution lors de l'ouverture de la file d'attente de demandes pour sauvegarder le contexte et de la file d'attente de réponses pour transmettre le contexte. S'il ne dispose pas des droits requis, ou si l'appel MQOPEN échoue et que le message de réponse n'est pas traité. Le message de réponse est inséré dans la file d'attente des messages non livrés avec l'en-tête de message non livrés contenant le code retour du MQOPENayant échoué. Le programme d'écoute continue ensuite à traiter les messages entrants suivants de manière normale.

ownContext
Le programme d'écoute SOAP ne transmet pas de contexte. Le contexte renvoyé reflète l'ID utilisateur sous lequel le programme d'écoute s'exécute plutôt que l'ID utilisateur qui a créé le message de demande d'origine.

Les zones du contexte d'origine sont définies par le gestionnaire de files d'attente et non par le programme d'écoute SOAP.

-n ***numThreads***
numThreads indique le nombre d'unités d'exécution dans les scripts de démarrage générés pour le programme d'écoute SOAP WebSphere MQ . La valeur par défaut est 10. Envisagez d'augmenter ce nombre si vous disposez d'un débit de messages élevé.

-v
-v définit la sortie prolixe des commandes externes. Les messages d'erreur sont toujours affichés. Utilisez -v pour générer des commandes que vous pouvez personnaliser pour créer des scripts de déploiement personnalisés.

-w ***serviceDirectory***
serviceDirectory est le répertoire contenant le service Web.

-x ***transactionnalité***
transactionality indique le type de contrôle transactionnel pour le programme d'écoute. *transactionality* peut être définie sur l'une des valeurs suivantes:

onePhase
La prise en charge en une phase de IBM WebSphere MQ est utilisée. Si le système échoue lors du traitement, le message de demande est redistribué à l'application. Les transactions WebSphere MQ garantissent que les messages de réponse sont écrits une seule fois.

twoPhase
La prise en charge en deux phases est utilisée. Si le service est écrit de manière appropriée, le message est distribué une seule fois, en coordination avec d'autres ressources, au sein

d'une seule exécution validée du service. Cette option s'applique uniquement aux connexions de liaisons de serveur.

none
Pas de prise en charge transactionnelle. Si le système échoue lors du traitement, le message de demande peut être perdu, même s'il est persistant. Le service peut avoir ou non été exécuté et les messages de réponse, de rapport ou de rebut peuvent ou non être écrits.

L'utilitaire de déploiement vérifie la compatibilité des indicateurs -x et -a . Pour plus d'informations, voir la description de l'indicateur -a .

Exemple en Java

```
java com.ibm.mq.soap.transport.jms.SimpleJavaListener
-u "jms:/queue?destination=myQ&connectionFactory=()
&initialContextFactory=com.ibm.mq.jms.Nojndi"
-n 20
```

WebSphere MQ Programmes d'écoute SOAP

Un programme d'écoute SOAP WebSphere MQ lit une demande SOAP entrante à partir de la file d'attente spécifiée comme destination dans l'URI. Il vérifie le format du message de demande, puis appelle un service Web à l'aide de l'infrastructure des services Web. Un programme d'écoute WebSphere MQ SOAP renvoie toute réponse ou erreur d'un service Web à l'aide de la file d'attente de destination de réponse dans l'URI. Elle renvoie des rapports WebSphere MQ à la file d'attente de réponses.

Le terme programme d'écoute est utilisé ici dans son sens standard des services Web. Il est distinct du programme d'écoute WebSphere MQ standard appelé par la commande **runmqlsr** .

Description

Le programme d'écoute Java SOAP est implémenté en tant que classe Java et exécute des services à l'aide d'Axis 1.4. Le programme d'écoute .NET est une application de console qui exécute les services .NET Framework 1 ou .NET Framework 2. Pour les services .NET Framework 3, utilisez le canal personnalisé WebSphere MQ pour Microsoft Windows Communication Foundation (WCF).

L'utilitaire de déploiement crée des scripts pour démarrer automatiquement les programmes d'écoute SOAP Java ou .NET. Un programme d'écoute SOAP peut être démarré manuellement à l'aide de la commande **amqSOAPNETListener** ou en appelant la classe `SimpleJavaListener` . Vous pouvez configurer le programme d'écoute SOAP WebSphere MQ à démarrer en tant que service WebSphere MQ en définissant l'option -s dans l'utilitaire de déploiement. Vous pouvez également démarrer les programmes d'écoute à l'aide du déclenchement ou utiliser les scripts de programme d'écoute de début et de fin générés par l'utilitaire de déploiement. Vous pouvez configurer le déclenchement manuellement ou utiliser les options de déploiement -tmq et -tmp pour configurer le déclenchement automatiquement. Vous pouvez arrêter un programme d'écoute en définissant la file d'attente des demandes sur `GET(DISABLED)`.

Tableau 586. Scripts de commande générés par l'utilitaire de déploiement

Infrastructure de service Web	Systèmes UNIX and Linux	Windows Java	Windows .NET
Démarrer le programme d'écoute	**startWMQJListener.sh**	**startWMQJListener.cmd**	**startWMQNListener.cmd**
Arrêter le programme d'écoute	**endWMQJListener.sh**	**endWMQJListener.cmd**	**endWMQNListener.cmd**
Définir le service de programme d'écoute	**defineWMQJListener.sh**	**defineWMQJListener.cmd**	**defineWMQNListener.cmd**

Le programme d'écoute SOAP WebSphere MQ transmet les zones `endpointURL` et `soapAction` du message SOAP à l'infrastructure SOAP. Le programme d'écoute appelle le service via l'infrastructure des services Web et attend la réponse. Le programme d'écoute ne valide pas `endpointURL` et `soapAction`. Les zones sont définies par l'émetteur SOAP WebSphere MQ à partir des données fournies dans l'URI défini par un client SOAP.

Le programme d'écoute crée le message de réponse et l'envoie à la destination de réponse fournie dans l'URI du message de demande. En outre, le programme d'écoute définit l'ID de corrélation dans le message de réponse en fonction de l'option de rapport dans le message de demande. Elle renvoie les paramètres d'expiration, de persistance et de priorité du message de demande. Le programme d'écoute envoie également des messages de rapport aux clients dans certaines circonstances.

En cas d'erreurs de format dans la demande SOAP, le programme d'écoute renvoie un message de rapport au client à l'aide de la file d'attente de destination de réponse. Le gestionnaire de files d'attente renvoie également des messages de rapport au client à l'aide de la file d'attente de destination de réponse, si un rapport a été demandé. Les messages de rapport complets sont écrits dans la file d'attente de réponses, en réponse à un certain nombre d'événements:

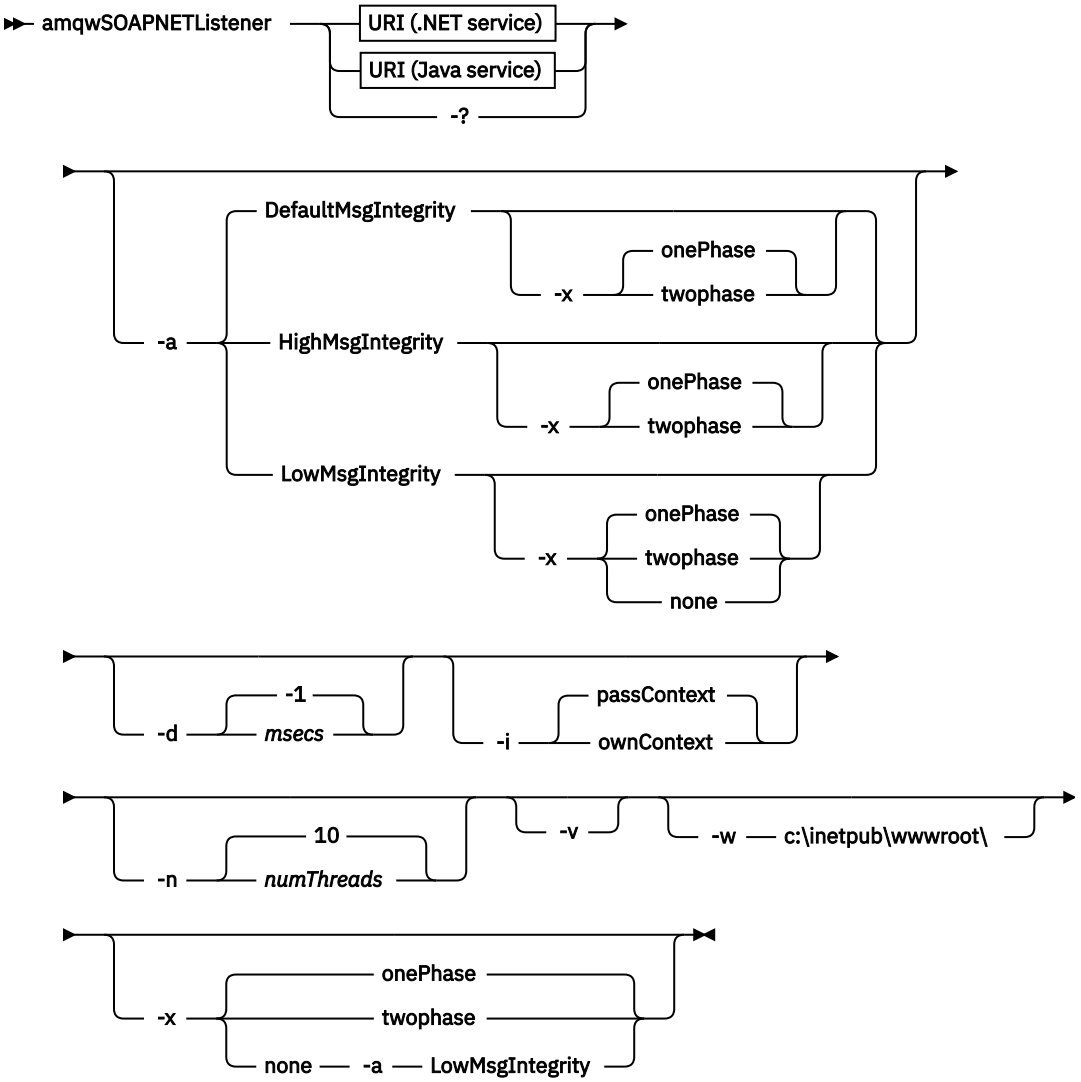
- Une exception.
- Expiration du message.
- Le format du message de demande n'est pas reconnu.
- Echec du contrôle d'intégrité de l'en-tête **MQRFH2** .
- Le format du corps du message principal n'est pas `MQFMT_NONE`.
- Le seuil d'annulation et de nouvelle tentative est dépassé lorsque le programme d'écoute SOAP WebSphere MQ traite la demande.

Les options de rapport WebSphere MQ SOAP sender `MQRO_EXCEPTION_WITH_FULL_DATA` et `MQRO_EXPIRATION_WITH_FULL_DATA` . En raison des options de rapport définies par l'émetteur SOAP WebSphere MQ , le message de rapport contient l'intégralité du message de demande d'origine. L'émetteur SOAP WebSphere MQ définit également l'option `MQRO_DISCARD` , qui entraîne la suppression du message après le renvoi d'un message de rapport. Si les options de rapport ne répondent pas à vos besoins, écrivez vos propres expéditeurs pour utiliser des options de rapport `MQRO_EXCEPTION` et `MQRO_DISCARD` différentes. Si la demande SOAP est envoyée par un autre expéditeur qui n'a pas défini `MQRO_DISCARD`, le message d'échec est écrit dans la file d'attente de rebut.

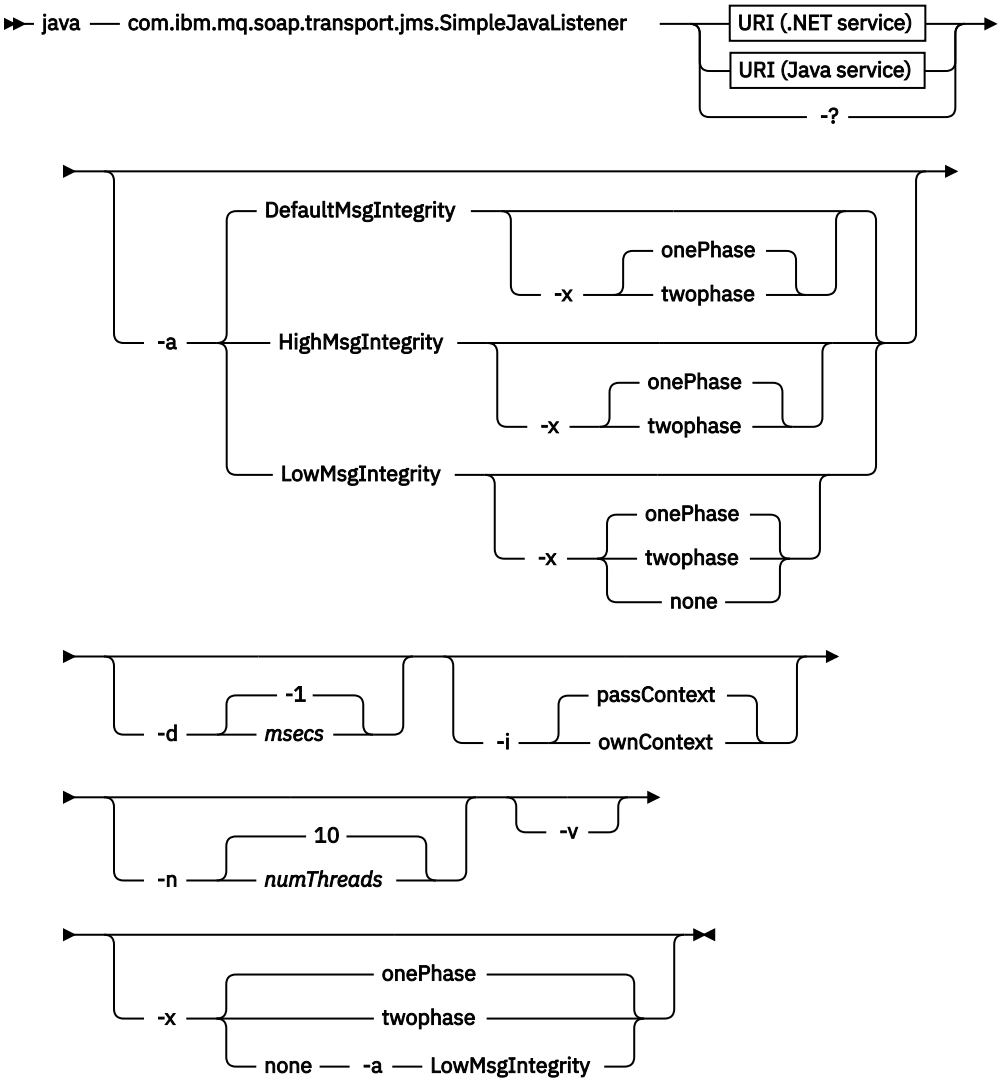
Si le programme d'écoute génère un message de rapport mais échoue dans le processus d'envoi du rapport, le message de rapport est envoyé au DLQ. Vérifiez que votre gestionnaire DLQ gère correctement ces messages.

Si une erreur se produit lors de la tentative d'écriture dans la file d'attente de rebut, un message est consigné dans le journal des erreurs WebSphere MQ . Le fait que le programme d'écoute continue à traiter d'autres messages dépend de la persistance des messages et des options transactionnelles sélectionnées. Si le programme d'écoute s'exécute en mode transactionnel à une phase et qu'il traite un message de demande non persistant, le message d'origine est supprimé. Le programme d'écoute SOAP WebSphere MQ continue de s'exécuter. Si le message de demande est persistant, le message de demande est sauvegardé dans la file d'attente des demandes et le programme d'écoute se ferme. La file d'attente des demandes est définie pour être verrouillée afin d'éviter un redémarrage accidentel déclenché.

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



Paramètres obligatoires

URI ***plateforme***
Voir «Syntaxe d'URI et paramètres pour le déploiement de service Web», à la page 1004.

-?
Imprimez le texte d'aide décrivant la façon dont la commande est utilisée.

Paramètres optionnels

-a ***integrityOption***
integrityOption indique le comportement des programmes d'écoute SOAP WebSphere MQ s'il n'est pas possible d'insérer un message de demande ayant échoué dans la file d'attente des messages non livrés. *integrityOption* peut prendre l'une des valeurs suivantes:

DefaultMsgIntegrity
Pour les messages non persistants, le programme d'écoute affiche un message d'avertissement et continue de s'exécuter avec le message d'origine en cours de suppression. Pour les messages persistants, il affiche un message d'erreur, annule le message de demande pour qu'il reste dans la file d'attente des demandes et se ferme. DefaultMsgIntegrity s'applique si l'option -a est omise ou si l'option *integrityOption* n'est pas spécifiée.

LowMsgIntegrity
Pour les messages persistants et non persistants, le programme d'écoute affiche un avertissement et continue de s'exécuter, en supprimant le message.

HighMsgIntegrity
Pour les messages persistants et non persistants, le programme d'écoute affiche un message d'erreur, annule le message de demande afin qu'il reste dans la file d'attente des demandes et se ferme.

L'utilitaire de déploiement vérifie la compatibilité des indicateurs -x et -a . Si -x none est spécifié, -a LowMsgIntegrity doit être spécifié. Si les indicateurs sont incompatibles, l'utilitaire de déploiement se ferme avec un message d'erreur et aucune étape de déploiement n'a été effectuée.

-d ***ms***
msecs indique le nombre de millisecondes pendant lequel le programme d'écoute SOAP WebSphere MQ doit rester actif si des messages de demande ont été reçus sur une unité d'exécution. Si *msecs* est défini sur -1, le programme d'écoute reste actif indéfiniment.

-i ***Contexte***
Contexte indique si les programmes d'écoute transmettent le contexte d'identité. *Context* prend les valeurs suivantes:

passContext
Définissez le contexte d'identité du message de demande d'origine dans le message de réponse. Le programme d'écoute SOAP vérifie qu'il est autorisé à sauvegarder le contexte à partir de la file d'attente des demandes et à le transmettre à la file d'attente des réponses. Il effectue les vérifications lors de l'exécution lors de l'ouverture de la file d'attente de demandes pour sauvegarder le contexte et de la file d'attente de réponses pour transmettre le contexte. S'il ne dispose pas des droits requis, ou si l'appel MQOPEN échoue et que le message de réponse n'est pas traité. Le message de réponse est inséré dans la file d'attente des messages non livrés avec l'en-tête de message non livrés contenant le code retour du MQOPENayant échoué. Le programme d'écoute continue ensuite à traiter les messages entrants suivants de manière normale.

ownContext
Le programme d'écoute SOAP ne transmet pas de contexte. Le contexte renvoyé reflète l'ID utilisateur sous lequel le programme d'écoute s'exécute plutôt que l'ID utilisateur qui a créé le message de demande d'origine.

Les zones du contexte d'origine sont définies par le gestionnaire de files d'attente et non par le programme d'écoute SOAP.

-n ***numThreads***
numThreads indique le nombre d'unités d'exécution dans les scripts de démarrage générés pour le programme d'écoute SOAP WebSphere MQ . La valeur par défaut est 10. Envisagez d'augmenter ce nombre si vous disposez d'un débit de messages élevé.

-v
-v définit la sortie prolixe des commandes externes. Les messages d'erreur sont toujours affichés. Utilisez -v pour générer des commandes que vous pouvez personnaliser pour créer des scripts de déploiement personnalisés.

-w ***serviceDirectory***
serviceDirectory est le répertoire contenant le service Web.

-x ***transactionnalité***
transactionality indique le type de contrôle transactionnel pour le programme d'écoute. *transactionality* peut être définie sur l'une des valeurs suivantes:

onePhase
La prise en charge en une phase de IBM WebSphere MQ est utilisée. Si le système échoue lors du traitement, le message de demande est redistribué à l'application. Les transactions WebSphere MQ garantissent que les messages de réponse sont écrits une seule fois.

twoPhase
La prise en charge en deux phases est utilisée. Si le service est écrit de manière appropriée, le message est distribué une seule fois, en coordination avec d'autres ressources, au sein

d'une seule exécution validée du service. Cette option s'applique uniquement aux connexions de liaisons de serveur.

none
Pas de prise en charge transactionnelle. Si le système échoue lors du traitement, le message de demande peut être perdu, même s'il est persistant. Le service peut avoir ou non été exécuté et les messages de réponse, de rapport ou de rebut peuvent ou non être écrits.

L'utilitaire de déploiement vérifie la compatibilité des indicateurs -x et -a . Pour plus d'informations, voir la description de l'indicateur -a .

Exemple .NET

```
amqwSOAPNETlistener
-u "jms:/queue?destination=myQ&connectionFactory=()
&targetService=myService&initialContextFactory=com.ibm.mq.jms.Nojndi"
-w C:/wmqsoap/demos
-n 20
```

Exemple en Java

```
java com.ibm.mq.soap.transport.jms.SimpleJavaListener
-u "jms:/queue?destination=myQ&connectionFactory=()
&initialContextFactory=com.ibm.mq.jms.Nojndi"
-n 20
```

Transport IBM WebSphere MQ pour l'expéditeur SOAP

Les classes d'expéditeur sont fournies pour Axis et .NET Framework 1 et .NET Framework 2. L'expéditeur construit une demande SOAP et la place dans une file d'attente, puis se bloque jusqu'à ce qu'il ait lu une réponse de la file d'attente de réponses. Vous pouvez modifier le comportement des classes en transmettant des URI différents à partir d'un client SOAP. Pour .NET Framework 3, utilisez le canal personnalisé WebSphere MQ pour Microsoft Windows Communication Foundation (WCF).

Objet

L'émetteur SOAP WebSphere MQ place une demande SOAP pour appeler un service Web dans une file d'attente de demandes WebSphere MQ . L'expéditeur définit les zones de l'en-tête **MQRFH2** en fonction des options spécifiées dans l'URI ou en fonction des valeurs par défaut.

Si vous devez modifier le comportement d'un expéditeur au-delà de ce qui est possible à l'aide des options d'URI, écrivez votre propre expéditeur. Votre expéditeur peut utiliser le transport IBM WebSphere MQ pour les programmes d'écoute SOAP ou d'autres environnements SOAP. Votre expéditeur doit construire des messages SOAP dans le format défini par WebSphere MQ. Le format est pris en charge par le programme d'écoute SOAP IBM WebSphere MQ , ainsi que par les programmes d'écoute SOAP fournis par WebSphere Application Server et CICS. Votre expéditeur doit suivre les règles d'un demandeur IBM WebSphere MQ . Le programme d'écoute SOAP IBM WebSphere MQ renvoie des messages de réponse et de rapport. Pour plus d'informations sur la définition des options de rapport dans le **MQMD**, voir «Paramètres SOAP MQMD», à la page 980 . Les options de rapport contrôlent les messages de rapport renvoyés par le programme d'écoute SOAP WebSphere MQ .

Description

L'émetteur Java SOAP WebSphere MQ est enregistré avec l'environnement hôte Axis pour le préfixe URI `jms:` . L'expéditeur est implémenté dans la classe `com.ibm.mq.soap.transport.jms.WMQSender`, qui est dérivée de `org.apache.axis.handlers.BasicHandler`. Si l'environnement hôte Axis détecte un préfixe URI `jms:` , il appelle la classe `com.ibm.mq.soap.transport.jms.WMQSender` . La classe se bloque après avoir placé le message jusqu'à ce qu'il ait lu une réponse de la file d'attente de réponses. Si aucune réponse n'est reçue dans un délai imparti, l'expéditeur émet une exception. Si une réponse est reçue dans le délai imparti, le message de réponse est renvoyé au client à l'aide de l'infrastructure Axis. Votre application client doit pouvoir traiter ces messages de réponse.

Pour les services Microsoft .NET Framework 1 et .NET Framework 2, l'émetteur SOAP WebSphere MQ est implémenté dans la classe `IBM.WMQSOAP.MQWebRequest`, qui est dérivée de `System.Net.WebRequest` et `System.Net.IwebRequestCreate`. Si .NET Framework 1 ou .NET Framework 2 détecte un préfixe URI `jms:` , il appelle la classe `IBM.WMQSOAP.MQWebRequest` . L'expéditeur crée un objet `MQWebResponse` pour lire le message de réponse de la file d'attente de réponses et le renvoyer au client.

`com.ibm.mq.soap.transport.jms.WMQSender` est une classe finale et `IBM.WMQSOAP.MQWebRequest` est scellé. Vous ne pouvez pas modifier leur comportement en créant des sous-classes.

Paramètres

Définissez l'URI pour contrôler le comportement de l'expéditeur SOAP IBM WebSphere MQ dans un client SOAP de service Web. L'utilitaire de déploiement crée des stubs de client de service Web incorporant les options d'URI fournies à l'utilitaire de déploiement.

Utilisez une table de définition de canal avec le transport SOAP WebSphere MQ pour l'émetteur SOAP

Une définition de canal de connexion client est une alternative à la définition des propriétés de connexion dans l'attribut ConnectionFactory de l'URI du service Web. Les propriétés de connexion sont les suivantes: clientChannel, clientConnectionet les paramètres SSL .

Description

Créez la table de description de canal du client en définissant des connexions client. Même si un client de services Web se connecte à différents gestionnaires de files d'attente, créez toutes les connexions dans la table des connexions sur un seul gestionnaire de files d'attente. Le nom et l'emplacement par défaut de la table de connexion sont *`queue manager directory`*`/@ipcc/AMQCLCHL.TAB`.

Transmettez l'emplacement de la table de connexion à un client Java en définissant la propriété système `com.ibm.mq.soap.transport.jms.mqchlurl` .

Transmettez l'emplacement de la table de connexion à un client .NET en définissant les variables d'environnement `MQCHLLIB` et `MQCHLTAB` .

Vous pouvez fournir à la fois une table de connexion de canal et des paramètres de connexion de canal dans l'attribut ConnectionFactory de l'URI du service Web. Les valeurs définies dans ConnectionFactory sont prioritaires sur les valeurs de la table de définition de canal.

Utilisation d'une table de définition de canal dans Java

```
java -Dcom.ibm.msg.client.config.location=file:/C:/mydir/myjms.config MyAppClass
```

Figure 22. Démarrage du client Java à l'aide d'un fichier de configuration

```
com.ibm.mq.soap.transport.jms.mqchlurl=file:/C:/ibm/wmq/qmgrs/QM1/@ipcc/AMQCLCHL.TAB
```

Figure 23. `myjms.config`

transactions

Utilisez l'option `-x` lors du démarrage du programme d'écoute pour exécuter des services Web de manière transactionnelle. Sélectionnez l'intégrité des messages en définissant l'option `persistence` dans l'URI de service.

Services Web

Utilisez l'option `-x` lors du démarrage du programme d'écoute pour exécuter des services Web de manière transactionnelle. Sur .NET Framework 1 et 2, le programme d'écoute SOAP utilise Microsoft

Transaction Coordinator (MTS). Sur Axis 1.4, le programme d'écoute SOAP utilise des transactions coordonnées de gestionnaire de files d'attente.

Clients de service Web
Les émetteurs SOAP ne sont pas transactionnels.

Liaisons WebSphere MQ
Vous pouvez définir le type de liaison pour l'expéditeur SOAP. Il peut se connecter en tant qu'application serveur WebSphere MQ ou en tant qu'application client. Vous pouvez également lier l'émetteur SOAP en tant que client XA sur .NET.

Persistance des messages

Sélectionnez le niveau de persistance en définissant l'option `Persistance` dans l'URI.

Transactions de service Web
Vous pouvez utiliser des transactions de service Web, car l'expéditeur SOAP n'est pas transactionnel. Si vous écrivez votre propre expéditeur SOAP et que vous prévoyez d'utiliser des transactions de service Web, ne créez pas d'expéditeur SOAP transactionnel. Vous ne pouvez pas envoyer le message de demande et recevoir le message de réponse dans la même transaction. L'envoi et la réception ne doivent pas être coordonnés par la transaction de service Web.

Syntaxe d'URI et paramètres pour le déploiement de service Web

La syntaxe et les paramètres permettant de déployer un service Web IBM WebSphere MQ sont définis dans un URI. L'utilitaire de déploiement génère un URI par défaut basé sur le nom du service Web. Vous pouvez remplacer les valeurs par défaut en définissant votre propre URI en tant que paramètre de l'utilitaire de déploiement. L'utilitaire de déploiement intègre l'URI dans les modules de remplacement de client de service Web générés.

Objet
Un service Web est spécifié à l'aide d'un identificateur URI (Universal Resource Identifier). Le diagramme de syntaxe indique l'URI pris en charge dans le transport IBM WebSphere MQ pour SOAP. L'URI contrôle les paramètres SOAP spécifiques à IBM WebSphere MQet les options utilisées pour accéder aux services cible. L'URI est compatible avec les services Web hébergés par .NET, Apache Axis 1, WebSphere Application Server, CICS.

Description
L'URI est incorporé dans les classes de client de service Web générées par l'utilitaire de déploiement. Le client transmet l'URI à l'expéditeur SOAP IBM WebSphere MQ dans un message IBM WebSphere MQ . L'URI contrôle le traitement effectué par l'émetteur SOAP IBM WebSphere MQ et le programme d'écoute SOAP IBM WebSphere MQ .

Syntax

The URI syntax is as follows:

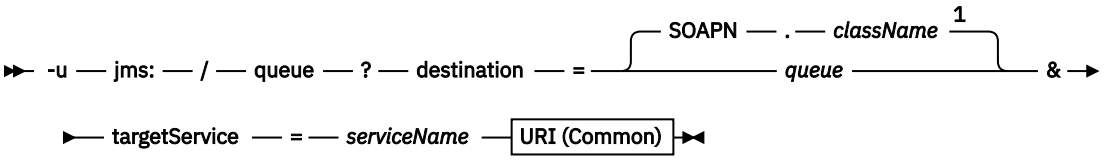
```
jms:/queue?name=value&name=value...
```

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 (&).

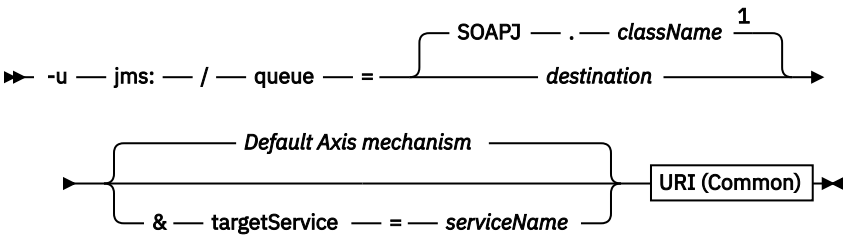
Parameter names are case-sensitive, as are names of IBM WebSphere MQ objects. If any parameter is specified more than once, the final occurrence of the parameter takes effect. Client applications can override a generated parameter by appending another copy of the parameter to the URI. If any additional 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.

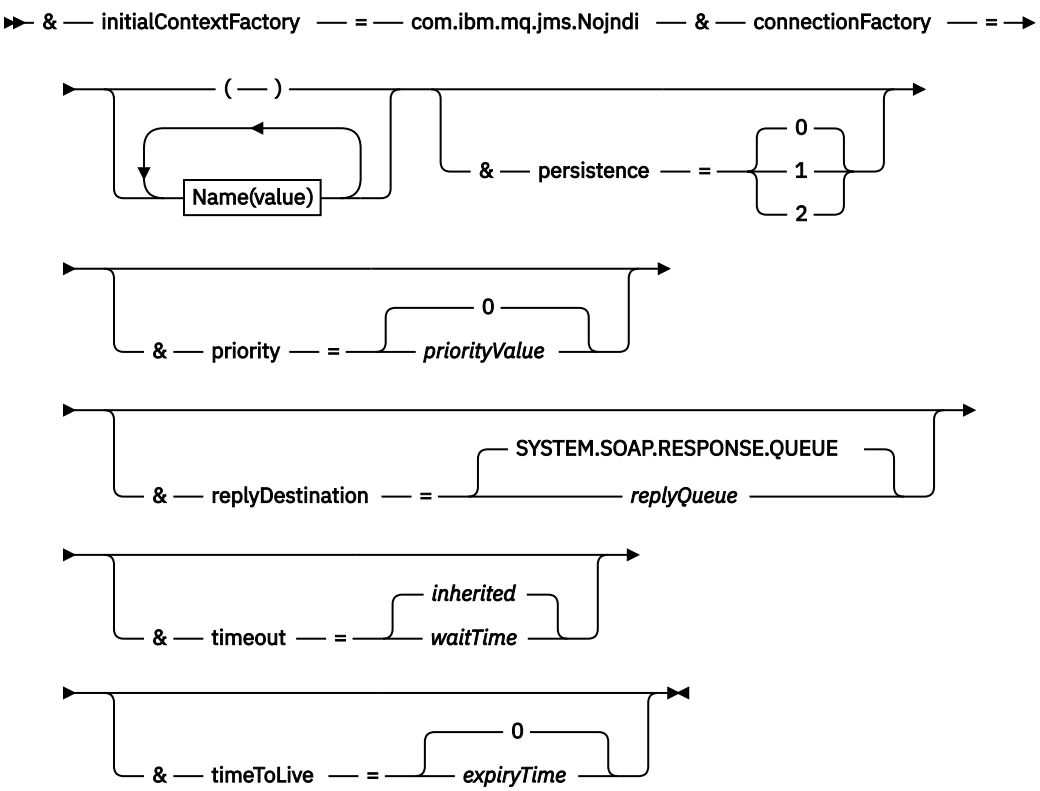
Syntax diagram
URI (.NET service)



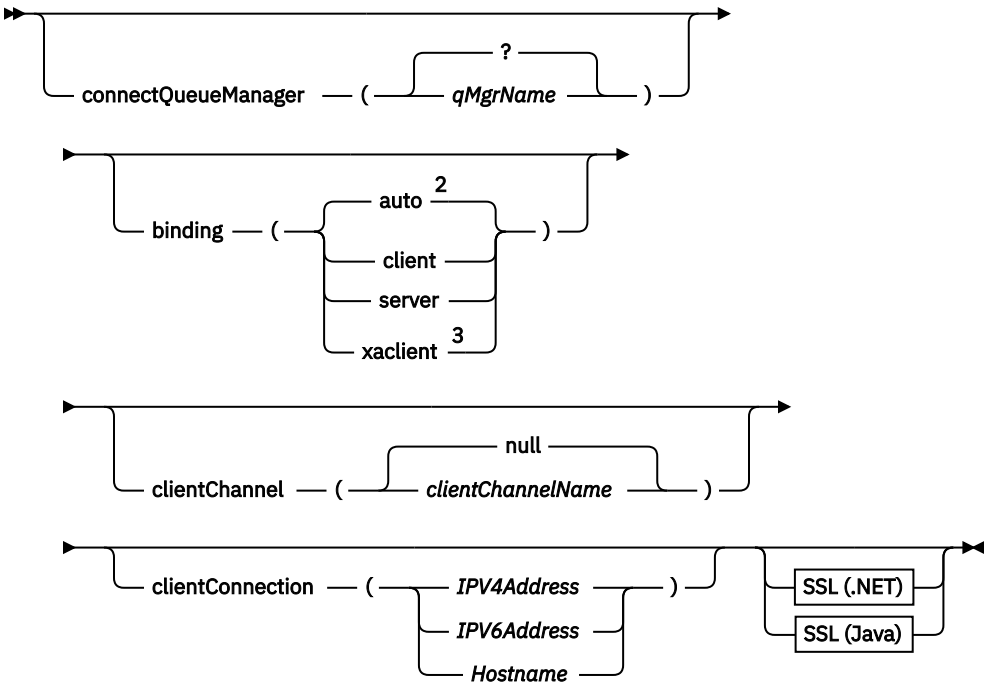
URI (Java service)



URI (Common)



***Name*(*value*)**



Remarques :

[1] The queue manager transforms *className* to a queue name following the steps described in «Transformation du nom de la destination en file d'attente», à la page 1006

[2] `client` is the default if other options appropriate for a client are specified; for example `clientConnection`.

[3] `xaclient` applies to .NET only

Transformation du nom de la destination en file d'attente

1. *className* est précédé de `SOAPJ.` pour les services Java ou de `SOAPN.` pour les services .NET.
2. L'extension de fichier est supprimée du chemin d'accès complet indiqué dans le paramètre *className* .
3. La chaîne résultante est tronquée à 48 caractères maximum
4. Les caractères de séparation de répertoire sont remplacés par des caractères de point.
5. Les espaces imbriqués sont remplacés par des caractères de soulignement.
6. Le signe deux-points suivant un préfixe d'unité est remplacé par un point pour un service .NET.

Remarque : Dans certains environnements, un nom de file d'attente généré par l'utilitaire de déploiement peut ne pas être unique. L'utilitaire de déploiement vérifie si la file d'attente doit être créée. Vous pouvez choisir de remplacer l'utilitaire de déploiement en restructurant la hiérarchie de répertoires de déploiement ou en personnalisant le processus de déploiement fourni.

Paramètres d'URI requis

`destination`=*file d'attente*

queue est le nom de la destination de la demande. Il peut s'agir d'une file d'attente ou d'un alias de file d'attente. S'il s'agit d'un alias de file d'attente, l'alias peut être résolu en rubrique.

- Si le paramètre -u est omis, *queue* est généré à partir de *classname* en suivant les étapes décrites dans «Transformation du nom de la destination en file d'attente», à la page 1006.
- Si le paramètre-u est spécifié, *queue* est obligatoire et doit être le premier paramètre de l'URI après le paramètre `jms:/queueinitial?` chaîne. Indiquez un nom de file d'attente IBM WebSphere MQ ou un nom de file d'attente et un nom de gestionnaire de files d'attente connectés par un symbole @, par exemple `SOAPN.trandemos@WMQSOAP.DEMO.QM`.

- L'utilitaire de déploiement vérifie si le nom de la file d'attente, généré ou fourni, correspond au nom d'une file d'attente existante. L'action effectuée est décrite dans Tableau 587, à la page 1007.

Tableau 587. Validation de la file d'attente

Le script du programme d'écoute existe?	**Le script du programme d'écoute existe dans le répertoire `./generated/server`**		**Le script du programme d'écoute n'existe pas dans le répertoire `./generated/server`**
La file d'attente dans le script d'écoute correspond à la *file d'attente*?	***queue* ne correspond pas à la file d'attente des demandes utilisée dans le script du programme d'écoute**	***queue* correspond à la file d'attente des demandes utilisée dans le script de programme d'écoute**	
La *file d'attente* existe	– *Les exits de déploiement avec une erreur.* – Le service a déjà été déployé dans `./generated/server`, mais avec une file d'attente différente.	– **Le déploiement se poursuit normalement.** – Le service a déjà été déployé dans `./generated/server`	– *Les exits de déploiement avec une erreur.* – Le script de démarrage du programme d'écoute est introuvable dans `./generated/server`, mais la *file d'attente* est utilisée par un autre service ou une autre application.
La *file d'attente* n'existe pas		– **Le déploiement se poursuit avec un avertissement.** – Le déploiement précédent a peut-être échoué car le démarrage est valide, mais la *file d'attente* est manquante.	– **Le déploiement se poursuit normalement.** – Aucun service n'a été déployé à partir de ce répertoire.

`&connectionFactory=`*Nom (valeur)*

Nom est l'un des paramètres suivants:

- GestionnaireconnectQueue(qMgrNom)
- liaison (bindingType)
- clientChannel(canal)
- clientConnection(connexion)
- «Paramètres SSL requis (Java)», à la page 993

Pour une description des valeurs de ces paramètres, voir «Paramètres de la fabrique de connexions», à la page 1009 .

`&targetService=`*serviceName*

[8]Sur .NET, *serviceName* est le nom d'un service .NET situé dans le répertoire de déploiement, par exemple: `targetService=mySservice.asmx`. Dans l'environnement .NET, le paramètre

[8] Service .NET uniquement

targetService permet à un seul programme d'écoute SOAP WebSphere MQ de pouvoir traiter des demandes pour plusieurs services. Ces services doivent être déployés à partir du même répertoire.

Paramètres d'URI facultatifs

&initialContextFactory=*contextFactory*
contextFactory est obligatoire et doit être défini sur `com.ibm.mq.jms.Nojndi`. Vérifiez que `Nojndi.jar` se trouve dans le chemin d'accès aux classes d'un client de services Web WebSphere Application Server. `Nojndi.jar` renvoie des objets Java en fonction du contenu des paramètres `connectionFactory` et `destination` , plutôt que par référence à un répertoire.

&targetService=*serviceName*
[9]Sur l'axe, *serviceName* est le nom qualifié complet d'un service Java , par exemple: `targetService=javaDemos.service.StockQuoteAxis`. Si targetService n'est pas spécifié, un service est chargé à l'aide du mécanisme Axis par défaut.

&persistence=*messagePersistence*
messagePersistence prend l'une des valeurs suivantes:

0
La persistance est héritée de la définition de file d'attente.

1
Le message est non persistant.

2
Le message est persistant

&priority=*priorityValue*
priorityValue est comprise entre 0 et 9. 0 est la priorité basse. La valeur par défaut est propre à l'environnement, qui, dans le cas de IBM WebSphere MQ , est 0.

&replyDestination=*replyToFile d'attente*
File d'attente côté client à utiliser pour le message de réponse. La file d'attente de réponses par défaut est `SYSTEM.SOAP.RESPONSE.QUEUE`.

- Exécutez le script `setupWMQSOAP` pour créer les objets SOAP WebSphere MQ par défaut.
- Spécifiez une file d'attente modèle pour *replyToQueue* afin de créer une file d'attente de réponse dynamique temporaire ou permanente. Pour les files d'attente de réponses dynamiques temporaires et permanentes, une instance distincte de file d'attente dynamique est créée pour chaque demande. Si l'un des événements suivants se produit, la file d'attente est supprimée:
  - La réponse arrive et est traitée.
  - La demande arrive à expiration.
  - Le programme demandeur s'arrête.

  Pour des performances optimales, utilisez des files d'attente dynamiques temporaires plutôt que des files d'attente dynamiques permanentes. N'envoyez pas de message de demande persistant à un URI avec une file d'attente dynamique temporaire. Le programme d'écoute IBM WebSphere MQ SOAP ne parvient pas à traiter le message et génère une erreur. Le client dépasse le délai d'attente de la réponse.
- Le script `setupWMQSOAP` crée une file d'attente de modèle dynamique permanente par défaut appelée `SYSTEM.SOAP.MODEL.RESPONSE.QUEUE`.

&timeout=*waitTime*
Durée, en millisecondes, pendant laquelle le client attend un message de réponse. *waitTime* remplace les valeurs définies par l'infrastructure ou l'application client. S'il n'est pas spécifié, la valeur de l'application, si elle est spécifiée, ou la valeur par défaut de l'infrastructure est héritée.

Remarque : Aucune relation n'est appliquée entre le délai d'attente et timeToLive.

[9] Service Java uniquement

&timeToLive=*expiryTime*
expiryTime est le temps, exprimé en millisecondes, avant l'expiration du message. La valeur par défaut est zéro, ce qui indique une durée de vie illimitée.

Remarque : Aucune relation n'est appliquée entre le délai d'attente et timeToLive.

Paramètres de la fabrique de connexions

connectQueueManager(*qMgrNom*)
qMgrNom indique le gestionnaire de files d'attente auquel le client se connecte. La valeur par défaut est nulle.

binding(*bindingType*)
bindingType indique comment le client est connecté à *Nom qMgr*. La valeur par défaut est auto. *bindingType* prend les valeurs suivantes:

auto
L'expéditeur tente les types de connexion suivants, dans l'ordre:

1. Si d'autres options appropriées à une connexion client sont spécifiées, l'émetteur utilise une liaison client. Les autres options sont clientConnection ou clientChannel.
2. Utilisez une connexion serveur.
3. Utilisez une connexion client.

Utilisez `binding(auto)` dans l' *URI* s'il n'y a pas de gestionnaire de files d'attente local sur le client SOAP. Une connexion client est générée pour le client SOAP.

client
Utilisez `binding(client)` dans l' *URI* pour générer une configuration client pour l'expéditeur SOAP.

server
Utilisez `binding(serveur)` dans l' *URI* pour générer une configuration de serveur pour l'expéditeur SOAP. Si la connexion possède des paramètres de type client, la connexion échoue et une erreur est affichée par l'expéditeur SOAP IBM WebSphere MQ . Les paramètres de type de client sont clientConnection, clientChannelou des paramètres SSL.

xaclient
xaclient est applicable uniquement sur .NET et non pour les clients Java. Utilisez une connexion client XA.

clientChannel(*canal*)

Le client SOAP utilise le *canal* pour établir une connexion client IBM WebSphere MQ . *channel* doit correspondre au nom d'un canal de connexion serveur, sauf si la définition automatique de canal est activée sur le serveur. clientChannel est un paramètre obligatoire, sauf si vous avez fourni une table de définition de canal du client (CCDT).

Indiquez une table de définition de canal du client dans Java en définissant `com.ibm.mq.soap.transport.jms.mqchlurl`. Dans .NET, définissez les variables d'environnement `MQCHLLIB` et `MQCHLTAB` ; voir «Utilisez une table de définition de canal avec le transport SOAP WebSphere MQ pour l'émetteur SOAP», à la page 1003.

clientConnection(*connexion*)
Le client SOAP utilise une *connexion* pour établir une connexion client IBM WebSphere MQ . Le nom d'hôte par défaut est localhostet le port par défaut est 1414. Si la *connexion* est une adresse TCP/IP, elle prend l'un des trois formats et peut être suffixée avec un numéro de port.

Les clients JMS peuvent utiliser le format: `hostname:port` ou'échapper'les crochets à l'aide du format %X , où X est la valeur hexadécimale qui représente le caractère de crochet dans la page de codes de l'URI. Par exemple, en ASCII, %28 et %29 pour `(` et `)` respectivement.

Les clients .Net peuvent utiliser explicitement les crochets: `hostname(port)` ou utiliser le format'échappement'.

Adresse IPv4
Par exemple, `192.0.2.0`.

Adresse IPv6
Par exemple, `2001:DB8:0:0:0:0:0:0`.

Nom d'hôte
Par exemple, `www.example.com%281687%29`, `www.example.com:1687` ou `www.example.com(1687)`.

SSL ***plateforme***
Voir le «Paramètres SSL requis (Java)», à la page 993

Exemples d'URI

Remarque :

1. & dans l'URI est codé en tant que `&`
2. Tous les paramètres répertoriés précédemment sont applicables aux clients.
3. Seuls **destination**, **connectionFactory** et **initialContextFactory** sont applicables au service WCF.

```
jms:/queue?
destination=myQ&amp;connectionFactory=()&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

Figure 24. URI d'un service Axis, fournissant uniquement les paramètres requis

```
jms:/queue?destination=myQ&amp;connectionFactory=()&amp;targetService=MyService.asmx
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

Figure 25. URI d'un service .NET, fournissant uniquement les paramètres requis

```
jms:/queue?destination=myQ@myRQM&amp;connectionFactory=connectQueueManager(myconnQM)
binding(client)clientChannel(myChannel)clientConnection(myConnection)
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

Figure 26. URI d'un service Axis, fournissant des paramètres connectionFactory facultatifs

```
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
```

Figure 27. URI d'un service Axis, fournissant l'option sslPeerName du paramètre connectionFactory

Mécanisme `Nojndi`

Le mécanisme `Nojndi` permet aux programmes JMS qui utilisent des interfaces JNDI d'utiliser le même URI que les programmes WebSphere MQ qui n'utilisent pas JNDI.

Vous pouvez utiliser le transport WebSphere MQ pour SOAP afin d'appeler des services Web sur WebSphere Application Server. WebSphere Application Server SOAP sur JMS recherche les ressources JMS à l'aide de JNDI. Le client de service Web peut s'exécuter sur .NET ou à l'aide d'Axis 1.4pour appeler le service Web et ne pas utiliser JNDI. Pour utiliser la même URL pour le client et le serveur, il doit fournir les mêmes informations, que l'environnement utilise JNDI ou non.

L'URI transmis au transport WebSphere MQ pour SOAP par un client de service Web contient un gestionnaire de files d'attente et des noms de file d'attente WebSphere MQ spécifiques. Ces noms sont analysés et utilisés directement par le support SOAP WebSphere MQ .

Le mécanisme `Nojndi` dirige la fabriqueinitialContext utilisée par un programme JMS vers `com.ibm.mq.jms.Nojndi`. La classe `com.ibm.mq.jms.Nojndi` est une implémentation de l'interface JNDI qui renvoie les objets Java connectionFactory et destination depuis l'URL en tant qu'objets Java

ConnectionFactory et Queue . Si l'implémentation JMS est WebSphere MQ, MQConnectionFactory et MQQueue héritent des classes ConnectionFactory et Queue .

En utilisant le mécanisme `Nojndi` , vous pouvez fournir les mêmes informations de connexion à WebSphere Application Server et .NET à l'aide de la même URL.

W3C URI SOAP sur JMS pour le client WebSphere MQ Axis 2

Définissez un URI W3C SOAP sur JMS pour appeler un service Web à partir d'un client Axis 2 en utilisant WebSphere MQ JMS comme transport SOAP. Le service Web doit être fourni par un serveur qui prend en charge WebSphere MQ JMS et la recommandation de candidat SOAP sur JMS W3C pour la liaison SOAP/JMS.

Description

La recommandation candidate W3C définit la liaison SOAP sur JMS, SOAP over Java Message Service 1.0. Il est également utile pour ses exemples : Schéma d'URI pour Java (tm) Message Service 1.0[10].

Utilisez le diagramme de syntaxe pour créer des URI W3C SOAP sur JMS syntaxiquement corrects et acceptés par le client WebSphere MQ Axis 2. Elle est limitée à la définition de l'URI acceptée par le client WebSphere MQ Axis 2. Il s'agit d'un sous-ensemble de la recommandation W3C à deux égards:

1. La rubrique jms-variant n'est pas prise en charge et ne doit pas être spécifiée dans un URI transmis au client WebSphere MQ Axis 2.
2. Les propriétés suivantes sont omises du diagramme de syntaxe car elles sont des propriétés JMS et ne font pas partie de l'URI.
   a. bindingVersion
   b. contentType
   c. soapAction
   d. requestURI
   e. isFault

Les propriétés JMS sont définies par le client Axis 2 ou le serveur.

Le diagramme étend la recommandation W3C en définissant un paramètre personnalisé, connectionFactory. connectionFactory est utilisé comme alternative à JNDI pour spécifier comment le client Axis 2 se connecte à un gestionnaire de files d'attente à l'aide d'une file d'attente.

Le client WebSphere MQ Axis 2 accepte uniquement les propriétés dans le cadre de l'URI transmis au client par l'application client ou en tant que variables d'environnement. Le client WebSphere MQ Axis 2 n'a pas la capacité de traiter un document WSDL. L'application client ou un outil de développement peut traiter le WSDL et créer l'URI à transmettre au client Axis 2. Une application client WebSphere MQ Axis 2 ne peut pas définir directement les propriétés de message JMS.

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,

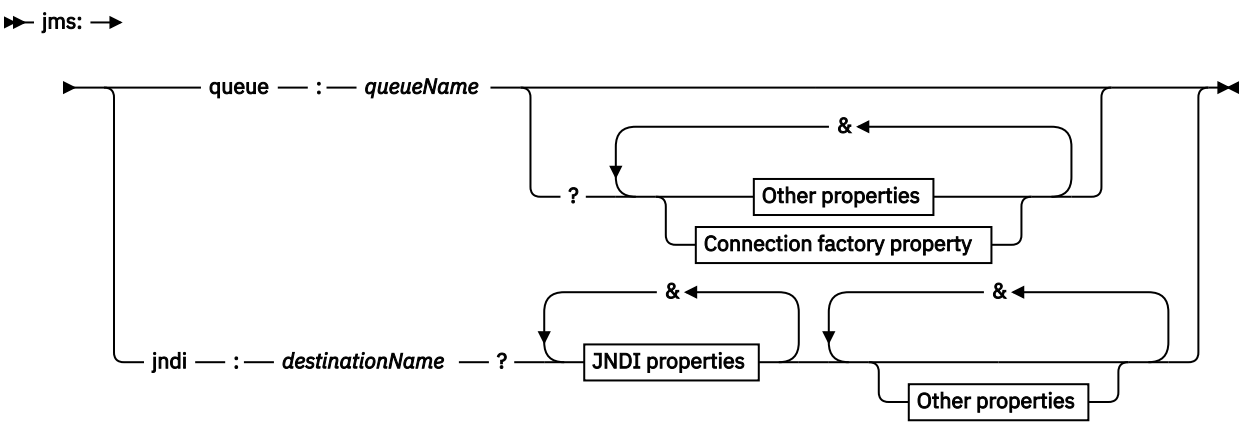
```
set soapjms_targetServer=com.example.org.stockquote
```

If a parameter is set using an environment variable it overrides the value set in the URI.

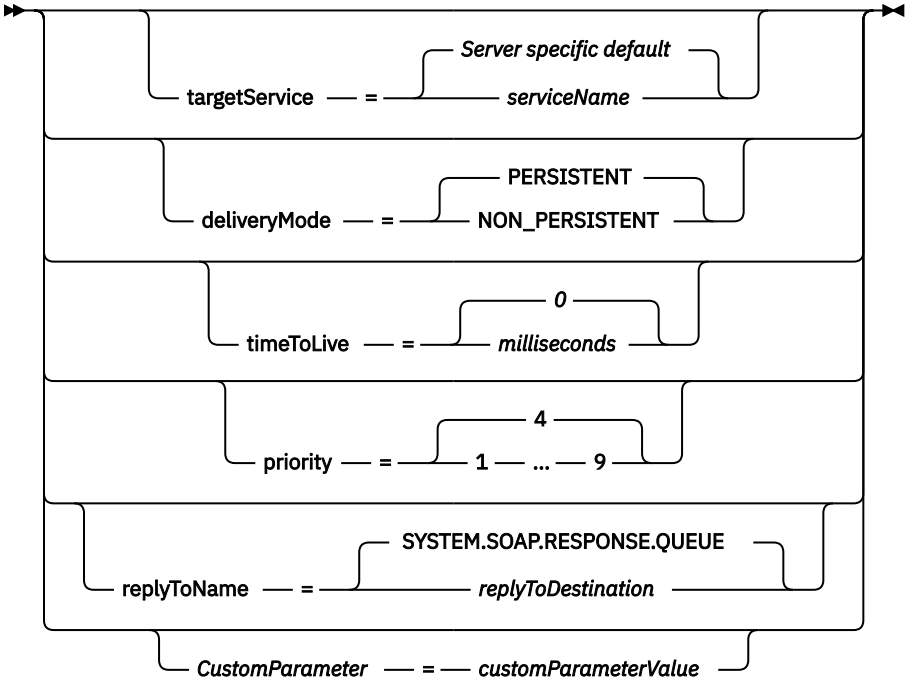
In accordance with the W3C recommendation, all the parameters can be repeated. The last instance of a parameter is used, unless overridden by an environment variable.

[10] Recherchez *Schéma d'URI pour JMS* dans les références de spécification W3C pour le dernier brouillon.

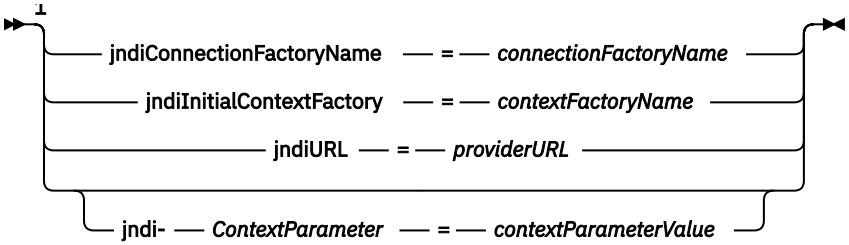
jms-uri



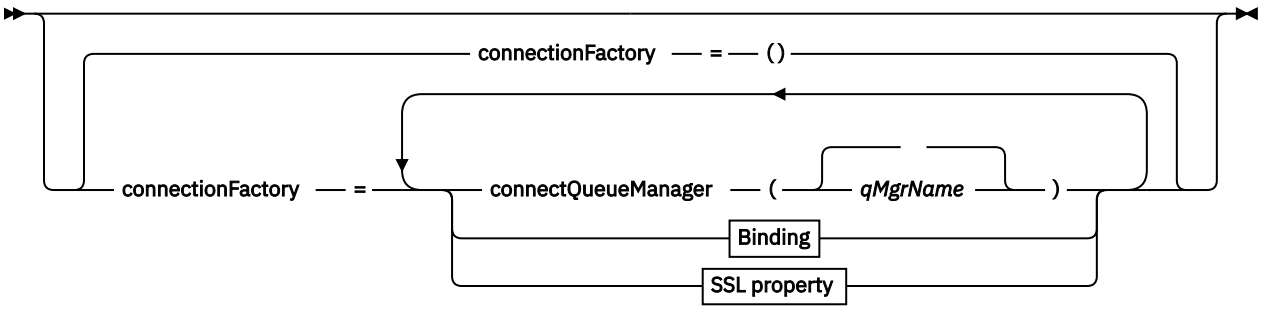
Other properties



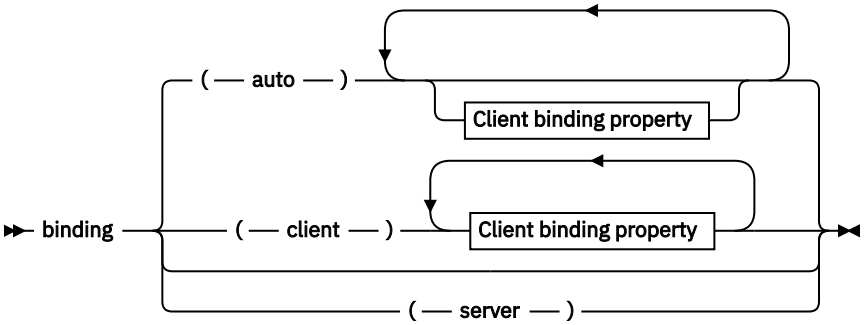
JNDI properties



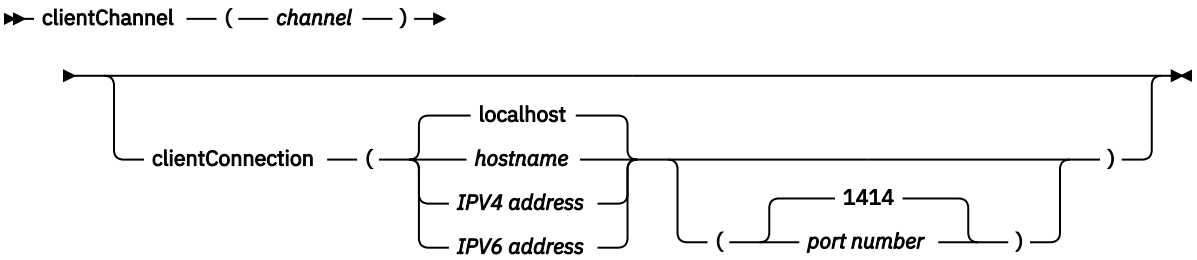
Connection factory property



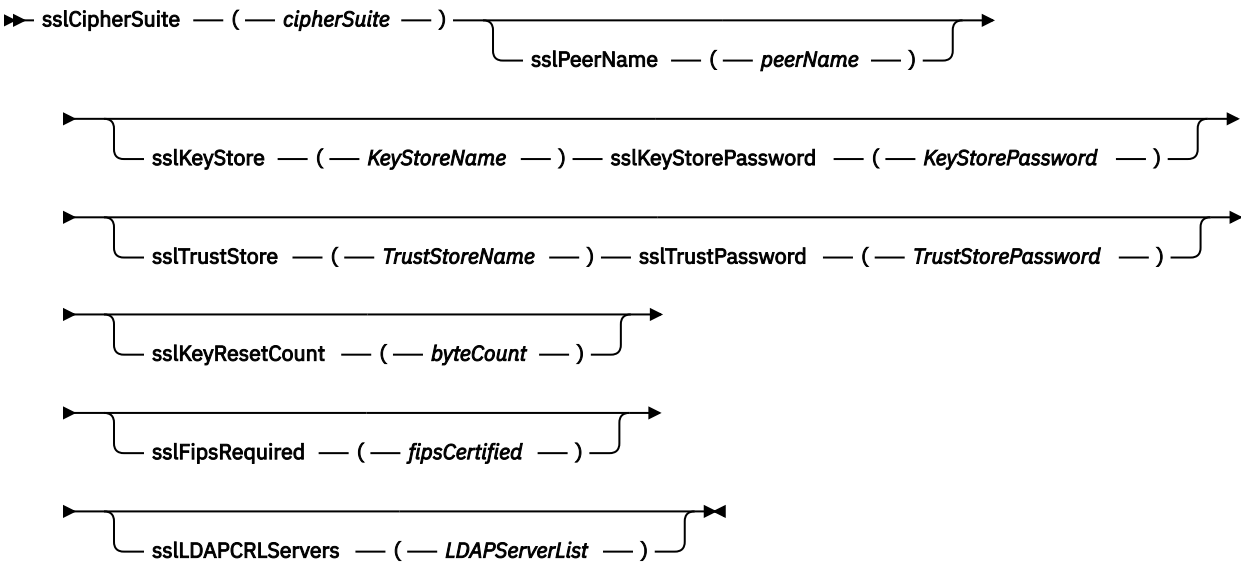
Binding



Client binding property



SSL property



Remarques :

[1] **jndiConnectionFactoryName**, **jndiConnectionFactoryName** and **jndiURL** are all required parameters. **jndi-***ContextParameter* is optional.

Paramètres

connectionFactory=*connectionFactoryParameterList*

connectionFactoryParameterList sont des paramètres qui qualifient la façon dont le client Axis 2 se connecte à un gestionnaire de files d'attente lorsque la variante de destination est queue.

connectionFactory ne doit pas être spécifié avec la variante de destination jndi .

Les paramètres ne sont pas transmis au serveur dans l'URI de la demande.

Si connectionFactory est omis, la file d'attente doit appartenir à un gestionnaire de files d'attente par défaut s'exécutant sur le même serveur que le client Axis 2.

connectionFactoryParameterList:

binding(*bindingType*)

bindingType indique comment le client est connecté à *Nom qMgr*. La valeur par défaut est auto. *bindingType* prend les valeurs suivantes:

auto

L'expéditeur tente les types de connexion suivants, dans l'ordre:

1. Si d'autres options appropriées à une connexion client sont spécifiées, l'émetteur utilise une liaison client. Les autres options sont clientConnection ou clientChannel.
2. Utilisez une connexion serveur.
3. Utilisez une connexion client.

Utilisez `binding(auto)` dans l' *URI* s'il n'y a pas de gestionnaire de files d'attente local sur le client SOAP. Une connexion client est générée pour le client SOAP.

client

Utilisez `binding(client)` dans l' *URI* pour générer une configuration client pour l'expéditeur SOAP.

server

Utilisez `binding(serveur)` dans l' *URI* pour générer une configuration de serveur pour l'expéditeur SOAP. Si la connexion possède des paramètres de type client, la connexion échoue et une erreur est affichée par l'expéditeur SOAP IBM WebSphere MQ . Les paramètres de type de client sont clientConnection, clientChannelou des paramètres SSL.

xaclient

xaclient est applicable uniquement sur .NET et non pour les clients Java. Utilisez une connexion client XA.

clientChannel(*canal*)

Le client SOAP utilise le *canal* pour établir une connexion client IBM WebSphere MQ . *channel* doit correspondre au nom d'un canal de connexion serveur, sauf si la définition automatique de canal est activée sur le serveur. clientChannel est un paramètre obligatoire, sauf si vous avez fourni une table de définition de canal du client (CCDT).

Indiquez une table de définition de canal du client dans Java en définissant `com.ibm.mq.soap.transport.jms.mqchlurl`. Dans .NET, définissez les variables d'environnement `MQCHLLIB` et `MQCHLTAB` ; voir «Utilisez une table de définition de canal avec le transport SOAP WebSphere MQ pour l'émetteur SOAP», à la page 1003.

clientConnection(*connexion*)

Le client SOAP utilise une *connexion* pour établir une connexion client IBM WebSphere MQ . Le nom d'hôte par défaut est localhostet le port par défaut est 1414. Si la *connexion* est une adresse TCP/IP, elle prend l'un des trois formats et peut être suffixée avec un numéro de port.

Les clients JMS peuvent utiliser le format: `hostname:port` ou'échapper'les crochets à l'aide du format %X , où X est la valeur hexadécimale qui représente le caractère de crochet dans la page de codes de l'URI. Par exemple, en ASCII, %28 et %29 pour (et) respectivement.

Les clients .Net peuvent utiliser explicitement les crochets: `hostname(port)` ou utiliser le format'échappement'.

Adresse IPv4
Par exemple, `192.0.2.0`.

Adresse IPv6
Par exemple, `2001:DB8:0:0:0:0:0:0`.

Nom d'hôte
Par exemple, `www.example.com%281687%29`, `www.example.com:1687` ou `www.example.com(1687)`.

sslCipherSuite(*CipherSuite*)
CipherSuite indique la suite sslCipher utilisée sur le canal. La suite de chiffrement CipherSuite spécifiée par le client doit correspondre à la suite de chiffrement CipherSuite spécifiée sur le canal de connexion serveur.

sslFipsRequired(*fipsCertified*)
fipsCertified indique si *CipherSpec* ou *CipherSuite* doit utiliser la cryptographie certifiée FIPS dans IBM WebSphere MQ sur le canal. L'effet de la définition de *fipsCertified* est identique à la définition de la zone `FipsRequired` de la structure **MQSCO** sur un appel `MQCONNX` .

sslKeyStore(*KeyStoreNom*)
KeyStoreName indique la sslKeyStoreName utilisée sur le canal. Le magasin de clés contient la clé privée du client utilisée pour authentifier le client auprès du serveur. Le magasin de clés est facultatif si la connexion SSL est configurée pour accepter les connexions client anonymes.

sslKeyResetCount(*bytecount*)
bytecount indique le nombre d'octets transmis via un canal SSL avant que la clé secrète SSL ne doive être renégociée. Pour désactiver la renégociation des clés SSL, omettez la zone ou définissez-la sur zéro. Zéro est la seule valeur prise en charge dans certains environnements. Voir Renégociation de la clé secrète dans WebSphere MQ classes for Java. La définition de sslKeyResetCount a le même effet que la définition de la zone `KeyResetCount` dans la structure **MQSCO** sur un appel `MQCONNX` .

sslKeyStorePassword(*KeyStoreMot de passe*)
KeyStorePassword indique le sslKeyStorePassword utilisé sur le canal.

sslLDAPCRLServers(*LDAPServerList*)

LDAPServerList indique une liste de serveurs LDAP à utiliser pour la vérification de la liste de révocation de certificat.

Pour les connexions client SSL, *LDAPServerList* est une liste de serveurs LDAP à utiliser pour la vérification de la liste de révocation de certificat (CRL). Le certificat fourni par le gestionnaire de files d'attente est vérifié par rapport à l'un des serveurs CRL LDAP répertoriés ; s'il est trouvé, la connexion échoue. Chaque serveur LDAP est essayé à son tour jusqu'à ce que la connectivité soit établie avec l'un d'entre eux. S'il est impossible de se connecter à l'un des serveurs, le certificat est rejeté. Une fois qu'une connexion a été établie avec succès avec l'un d'entre eux, le certificat est accepté ou rejeté en fonction des listes de révocation de certificat présentes sur ce serveur LDAP.

Si *LDAPServerList* est vide, le certificat appartenant au gestionnaire de files d'attente n'est pas vérifié par rapport à une liste de révocation de certificat. Un message d'erreur s'affiche si la liste d'URI LDAP fournie n'est pas valide. La définition de cette zone a le même effet que l'inclusion d'enregistrements `MQAIR` et l'accès à partir d'une structure **MQSCO** sur un `MQCONNX`.

sslPeerName(*peerName*)
peerName indique le sslPeerNom utilisé sur le canal.

sslTrustStore(*TrustStoreNom*)
TrustStore indique le sslTrustStoreName utilisé sur le canal. Le magasin de clés de confiance contient le certificat public du serveur, ou sa chaîne de clés, pour authentifier le serveur auprès du client. Le magasin de clés de confiance est facultatif si le certificat racine d'une autorité de certification est utilisé pour authentifier le serveur. Dans Java, les certificats racine sont stockés dans le magasin de certificats JRE, `cacerts`.

sslTrustStorePassword(*TrustStoreMot de passe*)
TrustStorePassword indique le sslTrustStorePassword utilisé sur le canal.

CustomParameter=customParameterValeur

CustomParameter est le nom défini par l'utilisateur d'un paramètre personnalisé et *customParameterValue* est la valeur du paramètre.

Les paramètres personnalisés qui ne sont pas utilisés par le client Axis 2 sont envoyés par le client Axis 2 au serveur SOAP. Consultez la documentation du serveur. connectionFactory est un paramètre personnalisé utilisé par le client Axis 2 et qui n'est pas transmis au serveur.

CustomParameter ne doit pas correspondre au nom d'un paramètre existant.

Si *CustomParameter* commence par la chaîne `jndi-` , il est utilisé pour la recherche d'une destination JNDI ; voir jndi-.

deliveryMode=*deliveryMode*
deliveryMode définit la persistance des messages. La valeur par défaut est PERSISTENT.

jndi:*destinationName*
destinationName est un nom de destination JNDI qui est mappé à une file d'attente JMS. Si la variante de destination jndi est spécifiée, vous devez indiquer un *destinationName*.

jndiConnectionFactoryName=*connectionFactoryNom*
connectionFactoryNom définit le nom JNDI de la fabrique de connexions. Si la variante de destination est jndi, *connectionFactoryName* doit être fourni.

jndiInitialContextFactory=*contextFactoryNom*
contextFactoryNom définit le nom JNDI de la fabrique de contexte initial. Si la variante de destination est jndi, *contextFactoryName* doit être fourni. Voir Utilisation de JNDI pour extraire des objets gérés dans une application JMS.

jndiURL=*providerURL*
jndiURL définit le nom d'URL du fournisseur JNDI. Si la variante de destination est jndi, *jndiURL* doit être spécifié.

jndi-*ContextParameter=contextParameterValeur*
jndi-*ContextParameter* est le nom défini par l'utilisateur d'un paramètre personnalisé utilisé pour transmettre des informations au fournisseur JNDI. *contextParameterValeur* correspond aux informations transmises.

priority=*priorityValue*
priorityValue définit la priorité des messages JMS. `0` est faible, 9 est élevé. La valeur par défaut est 4.

queue:*queueName*
queueName est le nom d'une file d'attente JMS dans laquelle la demande SOAP est placée. Si la variante de file d'attente est spécifiée, un nom de file d'attente doit être fourni. Si la file d'attente n'appartient pas à un gestionnaire de files d'attente par défaut sur le même serveur que le client, définissez le paramètre connectionFactory .

replyToName=*replyToDestination*
replyToDestination définit le nom de la file d'attente de destination. Si la variante de destination est jndi, le nom est un nom JNDI qui doit être mappé à une file d'attente. Si la variante est queue , le nom est une file d'attente JMS. La valeur par défaut est `SYSTEM.SOAP.RESPONSE.QUEUE`.

targetService=*serviceName*
Nom utilisé par le serveur SOAP pour démarrer le service Web cible.

Sur Axis, *serviceName* est le nom qualifié complet d'un service Java, par exemple: `targetService=www.example.org.StockQuote`. Si targetService n'est pas spécifié, un service est chargé à l'aide du mécanisme Axis par défaut.

timeToLive=*millisecondes*
Définissez *millisecondes* sur la durée avant l'expiration du message. La valeur par défaut, `0`, est le message qui n'expire jamais.

Exemples

```
jms:jndi:REQUESTQ
    ?jndiURL=file:/C:/JMSAdmin
    &jndiInitialContextFactory=com.sun.jndi.fscontext.RefFSContextFactory
    &jndiConnectionFactoryName=ConnectionFactory
    &replyToName=RESPONSEQ
    &deliveryMode(NON_PERSISTENT)
```

Figure 28. Utilisez jms: jndi pour envoyer une demande SOAP/JMS

```
jms:queue:SOAPJ.demos
    ?connectionFactory=connectQueueManager(QM1)
     Bind(Client)
     ClientChannel(SOAPClient)
     ClientConnection(www.example.org(1418))
    &deliveryMode(NON_PERSISTENT)
```

Figure 29. Utilisez jms: queue pour envoyer une demande SOAP/JMS

Services Web pris en charge

Le code qui a été écrit pour s'exécuter en tant que service Web n'a pas besoin d'être modifié pour utiliser le transport IBM WebSphere MQ pour SOAP. Vous devez déployer les services différemment pour qu'ils s'exécutent avec le transport IBM WebSphere MQ pour SOAP au lieu d'utiliser HTTP.

Description

Le transport WebSphere MQ pour SOAP fournit un programme d'écoute SOAP pour exécuter des services pour .NET Framework 1 et .NET 2, et pour Axis 1.4. Le canal personnalisé WebSphere MQ pour Microsoft Windows Communication Foundation exécute des services pour .NET Framework 3. WebSphere Application Server et CICS prennent en charge l'exécution de services sur WebSphere MQ transport for SOAP. Créez une exportation personnalisée pour utiliser WebSphere Enterprise Service Bus ou WebSphere Process Server.

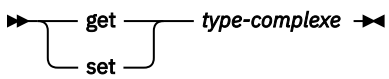
Le programme d'écoute SOAP WebSphere MQ peut traiter des demandes SOAP de manière transactionnelle. Exécutez **amqwdeployWMQService** à l'aide de l'option -x . L'option en deux phases est prise en charge uniquement pour les programmes d'écoute utilisant des liaisons de serveur. D'autres environnements peuvent fournir une prise en charge transactionnelle pour le transport WebSphere MQ pour SOAP. Consultez leur documentation.

WebSphere MQ transport for SOAP ne prend actuellement pas en charge le protocole SOAP sur JMS standard qui a été soumis à W3C. Vous pouvez distinguer un message SOAP/JMS écrit dans la nouvelle norme en recherchant la propriété JMS BindingVersion . WebSphere MQ transport for SOAP ne définit pas la propriété BindingVersion .

Axe 1.4

Une classe Java peut généralement être utilisée sans modification. Les types des arguments des méthodes du service Web doivent être pris en charge par le moteur Axis. Pour plus de détails, reportez-vous à la documentation Axis. Si le service utilise un objet complexe comme argument ou en renvoie un, cet objet doit être conforme à la spécification Java™ . Consultez les exemples dans Figure 32, à la page 1019, Figure 33, à la page 1019et Figure 34, à la page 1020:

1. Disposer d'un constructeur sans paramètre public.
2. Tous les types complexes du bean doivent avoir des méthodes d'accès public et des méthodes d'accès set de la forme:



Préparez le service pour le déploiement à l'aide de l'utilitaire **amqwdeployWMQService** . Le service est appelé par le programme d'écoute SOAP WebSphere MQ qui utilise `axis.jar` pour exécuter le service.

Le seul gestionnaire de transactions en deux phases pris en charge pour Axis 1.4 est WebSphere MQ.

L'utilitaire de déploiement fourni ne prend pas en charge le cas où un service renvoie un objet dans un autre package au service lui-même. Pour utiliser un objet renvoyé dans un autre package, écrivez votre propre utilitaire de déploiement. Vous pouvez baser votre utilitaire de déploiement sur l'exemple fourni ou capturer les commandes qu'il génère à l'aide de l'option -v . Modifiez les commandes pour générer un script personnalisé.

Si le service utilise des classes externes à l'infrastructure Axis et à l'environnement d'exécution SOAP WebSphere MQ , vous devez définir le paramètre CLASSPATHapproprié. Pour modifier CLASSPATH, modifiez le script généré qui démarre ou définit les programmes d'écoute pour inclure les services requis, de l'une des manières suivantes:

- Modifiez le fichier CLASSPATH directement dans le script après l'appel à **amqwsetcp**.
- Créez un script spécifique au service pour personnaliser le CLASSPATH et appelez ce script dans le script généré après l'appel à **amqwsetcp**.
- Créez un processus de déploiement personnalisé pour personnaliser automatiquement le CLASSPATH dans le script généré.

.NET Framework 1 et .NET Framework 2

Un service qui a déjà été préparé en tant que service Web HTTP n'a pas besoin d'être modifié pour être utilisé en tant que service Web WebSphere MQ . Il doit être déployé à l'aide de l'utilitaire **amqwdeployWMQService** .

Le seul gestionnaire de transactions en deux phases pris en charge pour .NET Framework 1 et .NET 2 est Microsoft Transaction Server (MTS).

Si le code de service n'a pas été préparé en tant que service Web HTTP, vous devez le convertir en service Web. Déclarez la classe en tant que service Web et identifiez comment les paramètres de chaque méthode sont formatés. Vous devez vérifier que les arguments des méthodes du service sont compatibles avec l'environnement. Figure 30, à la page 1019 et Figure 31, à la page 1019 montrent une classe .NET qui a été préparée en tant que service Web. Les ajouts effectués sont indiqués en caractères gras.

Figure 30, à la page 1019 utilise le modèle de programmation code-behind pour un service Web .NET. Dans le modèle code-behind, la source du service est séparée du fichier `.asmx` . Le fichier `.asmx` déclare le nom du fichier source associé avec le mot clé Codebehind . WebSphere MQ contient des exemples de services Web .NET en ligne et en code.

La source des services Web .NET doit être compilée avant le déploiement par l'utilitaire de déploiement **amqwdeployWMQService** . Le service est compilé dans une bibliothèque (`.dll`). La bibliothèque doit être placée dans le sous-répertoire `./bin` du répertoire de déploiement.

.NET Framework 3

Créez un canal personnalisé WebSphere MQ pour Microsoft Windows Communication Foundation (WCF) pour appeler les services déployés dans .NET Framework 3. Voir IBM WebSphere MQ custom channel for Microsoft Windows Communication Foundation (WCF) pour une description de la configuration de WCF pour utiliser le transport WebSphere MQ pour SOAP.

WebSphere Application Server

Vous pouvez appeler des services Web hébergés par WebSphere Application Server à l'aide de WebSphere MQ Transport for SOAP ; voir Utilisation de SOAP sur JMS pour le transport de services Web (obsolète).

Vous devez modifier le WSDL généré par le déploiement d'un service JMS sur WebSphere Application Server afin de générer un client de services Web. Le WSDL créé par le déploiement dans WebSphere Application Server inclut un URI avec une référence JNDI à la fabrique JMS `InitialContext`. Vous

devez modifier la référence JNDI à Nojndi et fournir des attributs de connexion comme décrit dans «Syntaxe d'URI et paramètres pour le déploiement de service Web», à la page 1004.

CICS

Vous pouvez appeler des applications CICS à l'aide de WebSphere MQ Transport for SOAP ; voir Configuration de votre système CICS pour les services Web.

WebSphere Enterprise Service Bus et WebSphere Process Server for Multiplatforms

WebSphere ESB et WebSphere Process Server for Multiplatforms prennent en charge SOAP sur JMS, avec une liaison prête à l'emploi, uniquement avec le fournisseur de messagerie WebSphere Application Server par défaut. Créez une liaison personnalisée pour JMS afin de prendre en charge le transport WebSphere MQ pour SOAP. Voir Liaisons de données JMS.

Exemple

```
<%@ WebService Language="C#" CodeBehind="Quote.asmx.cs" Class="Quote.QuoteDotNet" %>
```

Figure 30. Définition de service pour .NET Framework 2: `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();
        }
    }
}
```

Figure 31. Implémentation de service pour .NET Framework 2: `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;
}
```

Figure 32. Interface de service Java JAX-RPC utilisant un type complexe

```
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;
    }
}
```

Figure 33. Implémentation de service Java JAX-RPC à l'aide d'un type complexe

```
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; }
}
```

Figure 34. Implémentation de bean de service Java JAX-RPC d'un type complexe

Transport IBM WebSphere MQ pour les clients de service Web SOAP

Vous pouvez réutiliser un client SOAP sur HTTP existant avec le transport IBM WebSphere MQ pour SOAP. Vous devez apporter de petites modifications au code et au processus de génération afin de convertir le client pour qu'il utilise le transport IBM WebSphere MQ pour SOAP.

Ecriture du code

Les clients JAX-RPC doivent être écrits en Java. Les clients .NET Framework 1 et 2 peuvent être écrits dans n'importe quel langage qui utilise Common Language Runtime. Des exemples de code sont fournis dans C# et Visual Basic.

Le niveau de prise en charge transactionnelle dépend de l'environnement client et du modèle de l'interaction SOAP. La demande SOAP et la réponse SOAP ne peuvent pas faire partie de la même transaction atomique.

Vous devez appeler `IBM.WMQSOAP.Register.Extension()` dans un client .NET Framework 1, .NET Framework 2. Dans un appel de client de service Web Java JAX-RPC `com.ibm.mq.soap.Register.extension` , enregistrez l'émetteur SOAP WebSphere MQ . La méthode enregistre le transport WebSphere MQ pour l'expéditeur SOAP en tant que gestionnaire des messages SOAP à l'aide du protocole `jms:` .

Pour créer un client .NET Framework 3, générez un proxy client Windows Communication Foundation à l'aide de l'outil **svcutil** ; voir Génération d'un proxy client WCF et de fichiers de configuration d'application à l'aide de l'outil svcutil avec des métadonnées à partir d'un service en cours d'exécution.

Bibliothèques requises pour générer et exécuter les clients .NET Framework 1 et 2

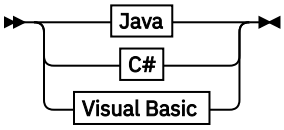
- amqsoap
- Système
- System.Web.Services
- System.Xml

Bibliothèques requises pour générer et exécuter les clients Axis 1.4

- *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

C#

Visual Basic

Exemples client

Figure 35, à la page 1021 est un exemple de client .NET Framework 1 ou .NET Framework 2 C# qui utilise le modèle de programmation en ligne. La méthode **IBM.WMQSOAP.Register.Extension()** enregistre l'émetteur SOAP WebSphere MQ avec .NET comme gestionnaire de protocole `jms:`.

```
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);
             }
        }
    }
}
```

Figure 35. Exemple de client de service Web C#

Figure 36, à la page 1022 est un exemple de client Java qui utilise l'interface client de proxy statique JAX-RPC. La méthode **com.ibm.mq.soap.Register.extension();** enregistre l'émetteur SOAP WebSphere MQ auprès du proxy de service pour gérer le protocole `jms:`.

```
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());
      }
   }
}
```

Figure 36. Exemple de client de service Web Java

Références des exits utilisateur, des exits API et des services installables

Utilisez les liens fournis dans cette section pour vous aider à développer vos exits utilisateur, exits API et applications de services installables:

- «Structure MQIEP», à la page 1023
- «Référence de l'exit de conversion de données», à la page 1026
- «MQ_PUBLISH_EXIT-Exit de publication», à la page 1029
- «Structures de données et appels d'exit de canal», à la page 1038
- «Référence d'exit API», à la page 1102
- «Informations de référence de l'interface des services installables», à la page 1162

Concepts associés

«Référence des applications MQI», à la page 7
Utilisez les liens fournis dans cette section pour vous aider à développer vos applications MQI:

«Les classes IBM WebSphere MQ pour les bibliothèques Java», à la page 1438
L'emplacement des bibliothèques IBM WebSphere MQ classes for Java varie en fonction de la plateforme. Indiquez cet emplacement lorsque vous démarrez une application.

Tâches associées

Développement d'applications

Référence associée

«Référence SOAP», à la page 964
Transport WebSphere MQ pour les informations de référence SOAP classées par ordre alphabétique.

«Informations de référence pour le pont IBM WebSphere MQ pour HTTP», à la page 1226
Rubriques de référence pour IBM WebSphere MQ bridge for HTTP, classées par ordre alphabétique

«Les classes et interfacesIBM WebSphere MQ .NET», à la page 1262
Les classes et interfaces IBM WebSphere MQ .NET sont répertoriées par ordre alphabétique. Les propriétés, les méthodes et les constructeurs sont décrits.

«IBM WebSphere MQ classes C++», à la page 1325
Les classes C++ IBM WebSphere MQ encapsulent l'interface MQI (IBM WebSphere MQ Message Queue Interface). Il existe un fichier d'en-tête C++ unique, **imqi.hpp**, qui couvre toutes ces classes.

WebSphere MQ classes for JMS

Structure MQIEP

La structure MQIEP contient un point d'entrée pour chaque appel de fonction que les exits sont autorisés à effectuer.

Zones

StrucId
Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

ID_STRUC_MQIPE

Version
Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

MQIEP_VERSION_1
Numéro de version de la structure de la version 1.

MQIEP_VERSION_CURRENT_
Version actuelle de la structure.

StrucLength
Type: MQLONG

Taille de la structure MQIEP en octets. La valeur est la suivante:

MQIEP_LENGTH_1

Indicateurs
Type: MQLONG

Fournit des informations sur les adresses de fonction. Un indicateur pour indiquer si la bibliothèque est à unités d'exécution peut être utilisé avec un indicateur pour indiquer si la bibliothèque est une bibliothèque client ou serveur.

La valeur suivante est utilisée pour ne spécifier aucune information de bibliothèque:

MQIEPF_AUCUN

L'une des valeurs suivantes est utilisée pour indiquer si la bibliothèque partagée est à unités d'exécution ou non:

MQIEPF_BIBLIOTHÈQUE_NON_UNITÉ_UNITÉ_EXÉCUTION
Une bibliothèque partagée sans unités d'exécution

MQIEPF_BIBLIOTHEQUE à unités d'exécution
Une bibliothèque partagée à unités d'exécution

L'une des valeurs suivantes est utilisée pour indiquer si la bibliothèque partagée est un client ou une bibliothèque partagée de serveur:

BIBLIOTHÈQUE_CLIENT_MQIEPF
Une bibliothèque partagée client

MQIEPF_BIBLIOTHÈQUE_LOCALE
Une bibliothèque partagée de serveur

Réservé
Type: MQPTR

Appel MQBACK_Call
Type: PMQ_BACK_CALL

Adresse de l'appel MQBACK.

Appel MQBEGIN_Call
Type: PMQ_BEGIN_CALL

Adresse de l'appel MQBEGIN.

Appel MQBUFMH_Call
Type: PMQ_BUFMH_CALL

Adresse de l'appel MQBUFMH.

Appel_MQCB
Type: PMQ_CB_CALL

Adresse de l'appel MQCB.

Appel MQCLOSE_Call
Type: PMQ_CLOSE_CALL

Adresse de l'appel MQCLOSE.

Appel_MQCM
Type: PMQ_CMIT_CALL

Adresse de l'appel MQCMIT.

Appel MQCONN_Call
Type: PMQ_CONN_CALL

Adresse de l'appel MQCONN.

Appel MQCONNX_Call
Type: PMQ_CONNX_CALL

Adresse de l'appel MQCONNX.

MQCRTMH_Appel
Type: PMQ_CRTMH_CALL

Adresse de l'appel MQCRTMH.

Appel_MQCTL
Type: PMQ_CTL_CALL

Adresse de l'appel MQCTL.

Appel_MQDC
Type: PMQ_DISC_CALL

Adresse de l'appel MQDISC.

Appel MQDLTMH_Call
Type: PMQ_DLTMH_CALL

Adresse de l'appel MQDLTMH.

Appel MQDLTMP_Call
Type: PMQ_DLTMP_CALL

Adresse de l'appel MQDLTMP.

Appel_MQGET
Type: PMQ_GET_CALL

Adresse de l'appel MQGET.

Appel MQINQ_Call
Type: PMQ_INQ_CALL

Adresse de l'appel MQINQ.

Appel MQINQMP_Call
Type: PMQ_INQMP_CALL

Adresse de l'appel MQINQMP.

Appel MQMHBUF_Appel
Type: PMQ_MHBUF_CALL

Adresse de l'appel MQMHBUF.

Appel MQOPEN_Call
Type: PMQ_OPEN_CALL

Adresse de l'appel MQOPEN.

Appel MQPUT_Call
Type: PMQ_PUT_CALL

Adresse de l'appel MQPUT.

MQPUT1_Call
Type: PMQ_PUT1_CALL

Adresse de l'appel MQPUT1 .

Appel MQSET_Call
Type: PMQ_SET_CALL

Adresse de l'appel MQSET.

Appel MQSETMP_Call
Type: PMQ_SETMP_CALL

Adresse de l'appel MQSETMP.

Appel_MQSTAT_Appel
Type: PMQ_STAT_CALL

Adresse de l'appel MQSTAT.

MQSUB_Appel
Type: PMQ_SUB_CALL

Adresse de l'appel MQSUB.

Appel MQSUBRQ_Call
Type: PMQ_SUBRQ_CALL

Adresse de l'appel MQSUBRQ.

Appel MQXCNVC_Call
Type: PMQ_XCNVC_CALL

Adresse de l'appel MQXCNVC.

Appel MQXCLWLN_Call
Type: PMQ_XCLWLN_CALL

Adresse de l'appel MQXCLWLN.

Appel MQXDX_Call
Type: PMQ_XDX_CALL

Adresse de l'appel MQXDX.

Appel MQXEP_Call
Type: PMQ_XEP_CALL

Adresse de l'appel MQXEP.

Appel_MQZ
Type: PMQ_ZEP_CALL

Adresse de l'appel MQZEP.

Déclaration C

```
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 */
 };
```

Référence de l'exit de conversion de données

Pour z/OS, vous devez écrire des exits de conversion de données en langage assembleur. Pour les autres plateformes, il est recommandé d'utiliser le langage de programmation C.

Pour vous aider à créer un programme d'exit de conversion de données, les éléments suivants sont fournis:

- Un fichier source squelette
- Un appel de conversion de caractères
- Utilitaire qui crée un fragment de code qui effectue la conversion de données sur les structures de type de données. Cet utilitaire utilise uniquement l'entrée C. Sous z/OS, il génère du code assembleur.

Pour la procédure d'écriture des programmes, voir:

- Ecriture d'un exit de conversion de données pour WebSphere MQ sur les systèmes UNIX and Linux
- Ecriture d'un exit de conversion de données pour WebSphere MQ for Windows

Fichier source squelette

Ils peuvent être utilisés comme point de départ lors de l'écriture d'un programme d'exit de conversion de données.

Les fichiers fournis sont répertoriés dans Tableau 588, à la page 1027.

Tableau 588. Fichiers source de squelette	
Plateforme	**Fichier**
AIX	amqsvfc0.c
IBM i	QMQMSAMP/QCSRC (AMQSVFC4)
HP-UX	amqsvfc0.c
Linux	amqsvfc0.c
z/OS	CSQ4BAX8 («1», à la page 1027) CSQ4BAX9 («2», à la page 1027) CSQ4CAX9 («3», à la page 1027)
Solaris	amqsvfc0.c
systèmes Windows	amqsvfc0.c
Remarques : 1. Illustre l'appel MQXCVNC. 2. Un encapsuleur pour les fragments de code générés par l'utilitaire à utiliser dans tous les environnements à l'exception de CICS. 3. Encapsuleur pour les fragments de code générés par l'utilitaire en vue de leur utilisation dans l'environnement CICS .	

Appel de conversion de caractères

Utilisez l'appel MQXCNVC (conversion de caractères) à partir d'un programme d'exit de conversion de données pour convertir des données de message de type caractères d'un jeu de caractères à un autre. Pour certains jeux de caractères multi-octets (par exemple, les jeux de caractères UCS2), les options appropriées doivent être utilisées.

Aucun autre appel MQI ne peut être effectué à partir de l'exit ; la tentative d'exécution d'un tel appel échoue avec le code anomalie MQRC_CALL_IN_PROGRESS.

Voir «MQXCNVC-Conversion de caractères», à la page 911 pour plus d'informations sur l'appel MQXCNVC et les options appropriées.

Utilitaire permettant de créer un code de sortie de conversion

Utilisez ces informations pour en savoir plus sur la création d'un code de sortie de conversion.

Les commandes permettant de créer un code d'exit de conversion sont les suivantes:

IBM i
CVTMQMDTA (Convertir le type de données WebSphere MQ)

Windows, systèmes UNIX and Linux
`crtmqcvx` (Création de l'exit de conversion WebSphere MQ)

La commande de votre plateforme génère un fragment de code qui effectue la conversion de données sur les structures de type de données, à utiliser dans votre programme d'exit de conversion de données. La commande utilise un fichier contenant une ou plusieurs définitions de structure de langage C. .

Messages d'erreur dans les systèmes Windows, UNIX and Linux

La commande `crtmqcvx` renvoie des messages compris entre AMQ7953 et AMQ7970.

Ces messages sont répertoriés dans Codes anomalie *WebSphere MQ Messages*.

Il existe deux principaux types d'erreur:

- Les erreurs majeures, telles que les erreurs de syntaxe, lorsque le traitement ne peut pas continuer.

  Un message s'affiche à l'écran indiquant le numéro de ligne de l'erreur dans le fichier d'entrée. Le fichier de sortie a peut-être été partiellement créé.
- Autres erreurs lorsqu'un message s'affiche indiquant qu'un problème a été détecté mais que l'analyse syntaxique de la structure peut continuer.

  Le fichier de sortie a été créé et contient des informations d'erreur sur les problèmes qui se sont produits. Ces informations d'erreur sont préfixées par `#error` de sorte que le code généré ne soit accepté par aucun compilateur sans intervention pour corriger les problèmes.

Syntaxe valide

Votre fichier d'entrée pour l'utilitaire doit être conforme à la syntaxe du langage C.

Si vous n'êtes pas familiarisé avec C, reportez-vous à l' exemple C dans cette rubrique.

En outre, tenez compte des règles suivantes:

- typedef est reconnu uniquement avant le mot clé struct.
- Une balise de structure est requise sur vos déclarations de structure.
- Vous pouvez utiliser des crochets vides [ ] pour indiquer un tableau ou une chaîne de longueur variable à la fin d'un message.
- Les tableaux multidimensionnels et les tableaux de chaînes ne sont pas pris en charge.
- Les types de données supplémentaires suivants sont reconnus:
  - MQBOOL
  - MQBYTE
  - MQCHAR
  - MQFLOAT32
  - MQFLOAT64
  - MQSHORT
  - MQLONG
  - MQINT8
  - MQUINT8
  - MQINT16
  - MQUINT16
  - MQINT32
  - MQUINT32
  - MQINT64
  - MQUINT64

  Les zones MQCHAR sont converties en page de codes, mais MQBYTE, MQINT8 et MQUINT8 restent intactes. Si le codage est différent, MQSHORT, MQLONG, MQINT16, MQUINT16, MQINT32, MQUINT32, MQINT64, MQUINT64, MQFLOAT32, MQFLOAT64 et MQBOOL sont convertis en conséquence.
- N'utilisez *pas* les types de données suivants:
  - double
  - Pointeurs

– zones de bits

En effet, l'utilitaire de création de code de sortie de conversion ne permet pas de convertir ces types de données. Pour y remédier, vous pouvez écrire vos propres routines et les appeler à partir de l'exit.

Autres points à noter:

- N'utilisez pas de numéros de séquence dans le fichier d'entrée.
- S'il existe des zones pour lesquelles vous souhaitez fournir vos propres routines de conversion, déclarez-les comme MQBYTE, puis remplacez les macros CMQXCFBA générées par votre propre code de conversion.

Exemple C

```
  struct TEST { MQLONG    SERIAL_NUMBER;
                MQCHAR    ID[5];
                MQINT16   VERSION;
                MQBYTE    CODE[4];
                MQLONG    DIMENSIONS[3];
                MQCHAR    NAME[24];
              } ;
```

Cela correspond aux déclarations suivantes dans d'autres langages de programmation:

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

Pris en charge sur z/OS uniquement

```
 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-Exit de publication

L'appel MQ_PUBLISH_EXIT peut inspecter et modifier les messages distribués aux abonnés.

Objet

Utilisez l'exit de publication pour inspecter et modifier les messages distribués aux abonnés:

- Examiner le contenu d'un message publié pour chaque abonné
- Modifier le contenu d'un message publié pour chaque abonné
- Modifier la file d'attente dans laquelle un message est inséré
- Arrêter la distribution d'un message à un abonné

Syntaxe

MQ_PUBLISH_EXIT(*ExitParms, PubContext, SubContext*)

Paramètres

***ExitParms* (MQPSXP)- Input/Output**
ExitParms contient des informations sur l'appel de l'exit.

***PubContext* (MQPBC) - Input**
PubContext contient des informations contextuelles sur le diffuseur de publications de la publication.

***SubContext* (MQSBC) - Input/Output**
SubContext contient des informations contextuelles sur l'abonné recevant la publication.

MQPSXP-Structure de données de l'exit de publication

La structure MQPSXP décrit les informations qui sont transmises à et renvoyées par l'exit de publication.

Le Tableau 589, à la page 1030 récapitule les zones de la structure:

Tableau 589. Zones dans MQPSXP

Zone	Description
StrucID	Identificateur de structure
Version	Numéro de version de structure
ExitId	Type d'exit appelé
ExitReason	Raison de l'appel de l'exit
ExitResponse	Réponse de l'exit
ExitResponse2	Réponse secondaire de l'exit
Feedback	Code retour
ExitUserArea	Quitter la zone utilisateur
ExitData	Données d'exit
QMgrName	Nom du gestionnaire de files d'attente local
Hconn	Descripteur de connexion
MsgDescPtr	Adresse du descripteur de message (MQMD)
MsgHandle	Traitement des propriétés de message (MQHMSG)
MsgInPtr	Adresse du message d'entrée
MsgInLength	Longueur du message d'entrée
MsgOutPtr	Adresse du message de sortie

Tableau 589. Zones dans MQPSXP (suite)	
Zone	**Description**
MsgOutLength	Longueur du message de sortie
pEntryPoints	Adresse de la structure MQIEP

Zones

***StrucID* (MQCHAR4)**
StrucID est l'identificateur de structure. La valeur est la suivante:

MQPSXP_STRUCID
MQPSXP_STRUCID est l'identificateur de la structure des paramètres d'exit de publication. Pour le langage de programmation C, la constante MQPSXP_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQPSXP_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

StrucID est une zone d'entrée de l'exit.

***Version* (MQLONG)**
Version est le numéro de version de la structure. La valeur est la suivante:

MQPSXP_VERSION_1
MQPSXP_VERSION_1 est la structure des paramètres d'exit de publication version 1. La constante MQPSXP_CURRENT_VERSION est également définie avec la même valeur.

Version est une zone d'entrée de l'exit.

***ExitId* (MQLONG)**
ExitId est le type d'exit appelé. La valeur est la suivante:

MQXT_PUBLISH_EXIT
Exit de publication.

ExitId est une zone d'entrée de l'exit.

***ExitReason* (MQLONG)**
ExitReason est la raison pour laquelle l'exit est appelé. Les valeurs possibles sont les suivantes:

MQXR_INIT
L'exit de cette connexion est appelé pour l'initialisation. L'exit peut acquérir et initialiser les ressources dont il a besoin ; par exemple, la mémoire principale.

MQXR_TERM
L'exit de cette connexion est appelé car l'exit est sur le point d'être arrêté. L'exit doit libérer toutes les ressources qu'il a acquises depuis qu'il a été initialisé ; par exemple, la mémoire principale.

MQXR_PUBLICATION
L'exit est appelé par le gestionnaire de files d'attente avant de placer une publication dans une file d'attente de messages d'un abonné. L'exit peut modifier le message, ne pas le placer dans la file d'attente ou arrêter la publication.

ExitReason est une zone d'entrée de l'exit.

***ExitResponse* (MQLONG)**
Définissez *ExitResponse* dans l'exit pour indiquer comment le traitement doit se poursuivre. *ExitResponse* est l'une des valeurs suivantes:

MQXCC_OK
Définissez MQXCC_OK pour poursuivre le traitement normalement. Définissez MQXCC_OK en réponse à toute valeur de ExitReason.

Si ExitReason a la valeur MQXR_PUBLICATION , les zones *DestinationQName* et *DestinationQMgrName* de la structure MQSBC identifient la destination à laquelle le message est envoyé.

MQXCC_FAILED
Définissez MQXCC_FAILED pour arrêter l'opération de publication. Le code achèvement MQCC_FAILED et le code anomalie 2557 (09FD) (RC2557): MQRC_PUBLISH_EXIT_ERROR sont définis en cas de retour de l'exit.

MQXCC_SUPPRESS_FUNCTION
Définissez MQXCC_SUPPRESS_FUNCTION pour arrêter le traitement normal du message. Définissez MQXCC_SUPPRESS_FUNCTION uniquement si *ExitReason* a la valeur MQXR_PUBLICATION .

Le message continue d'être traité par le gestionnaire de files d'attente en fonction de l'option MQRO_DISCARD_MSG dans la zone *Report* du descripteur de message du message.

- Si l'option MQRO_DISCARD_MSG est spécifiée, le message n'est pas distribué à l'abonné.
- Si l'option MQRO_DISCARD_MSG n'est pas spécifiée, le message est placé dans la file d'attente de rebut. S'il n'y a pas de file d'attente de rebut ou si le message ne peut pas être placé dans la file d'attente de rebut, la publication n'est pas distribuée à l'abonné. La distribution de la publication aux autres abonnés dépend des valeurs des attributs d'objet de rubrique PMSGDLV et NPMSGDLV . Pour plus d'informations sur ces attributs, voir les descriptions des paramètres de la commande DEFINE TOPIC .

ExitResponse est une zone de sortie de l'exit.

***ExitResponse2* (MQLONG)**
ExitResponse2 est réservé pour une utilisation ultérieure.

***Feedback* (MQLONG)**
Feedback est le code retour à utiliser si l'exit renvoie MQXCC_SUPPRESS_FUNCTION dans *ExitResponse*.

En entrée de l'exit, *Feedback* a toujours la valeur MQFB_NONE. Si l'exit renvoie MQXCC_SUPPRESS_FUNCTION, définissez *Feedback* sur la valeur à utiliser pour le message lorsque le gestionnaire de files d'attente le place dans la file d'attente de rebut. En retour de l'exit, si *Feedback* a la valeur d'origine MQFB_NONE, le gestionnaire de files d'attente définit *Feedback* sur MQFB_STOPPED_BY_PUBSUB_EXIT.

Feedback est une zone d'entrée-sortie de l'exit.

***ExitUserArea* (MQBYTE16)**
ExitUserArea est une zone disponible que l'exit peut utiliser. Chaque connexion possède un *ExitUserArea*distinct. La longueur de *ExitUserArea* est donnée par MQ_EXIT_USER_AREA_LENGTH .

La zone *ExitReason* a la valeur MQXR_INIT lors du premier appel de l'exit. *ExitUserArea* est initialisé sur MQXUA_NONE lors du premier appel de l'exit pour une connexion. Les modifications ultérieures apportées à *ExitUserArea* sont conservées entre les appels de l'exit.

ExitUserArea est une zone d'entrée-sortie de l'exit.

***ExitData* (MQCHAR32)**
ExitData est une donnée d'exit fixe définie par le paramètre *PublishExitData* de la strophe dans le fichier d'initialisation du gestionnaire de files d'attente. Les données sont remplies avec des blancs sur toute la longueur de la zone. Si aucune donnée d'exit fixe n'est définie dans le fichier d'initialisation, *ExitData* est vide. La longueur de *ExitData* est donnée par MQ_EXIT_DATA_LENGTH.

ExitData est une zone d'entrée de l'exit.

***QMgrName* (MQCHAR48)**
QMgrName est le nom du gestionnaire de files d'attente local. Le nom est rempli avec des blancs sur toute la longueur de la zone. La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH .

QMgrName est une zone d'entrée de l'exit.

***Hconn* (MQHCONN)**
Hconn est le descripteur représentant une connexion au gestionnaire de files d'attente. Utilisez uniquement *Hconn* comme paramètre pour les appels de fonction de propriété de message MQSETMP, MQINQMMPou MQDLTMP pour utiliser les propriétés de message.

Hconn est une zone d'entrée de l'exit.

***MsgDescPtr* (PMQMD)**
MsgDescPtr est l'adresse du descripteur de message (MQMD) du message en cours de traitement et est une copie du MQMD renvoyé par l'appel MQPUT. L'exit peut modifier le contenu du descripteur de message. Toute modification du contenu du descripteur de message doit être effectuée avec précaution. En particulier, dans le cas où la zone *SubType* de la structure MQSBC est de valeur MQSUBTYPE_PROXY, la zone *CorrelId* du descripteur de message ne doit pas être modifiée.

Aucun descripteur de message n'est transmis à l'exit si *ExitReason* est MQXR_INIT ou MQXR_TERM ; dans ces cas, *MsgDescPtr* est le pointeur null.

MsgDescPtr est une zone d'entrée de l'exit.

***MsgHandle* (MQHMSG)**
MsgHandle est le descripteur des propriétés de message. Utilisez uniquement *MsgHandle* avec les appels de fonction de propriété de message MQSETMP, MQINQMMP ou MQDLTMP pour utiliser les propriétés de message.

MsgHandle est une zone d'entrée de l'exit.

***MsgInPtr* (PMQVOID)**
MsgInPtr est l'adresse des données de message d'entrée. Le contenu de la mémoire tampon adressée par *MsgInPtr* peut être modifié par l'exit ; voir *MsgOutPtr* .

MsgInPtr est une zone d'entrée de l'exit.

***MsgInLength* (MQLONG)**
MsgInLength est la longueur en octets des données de message transmises à l'exit. L'adresse des données est indiquée par *MsgInPtr*.

MsgInLength est une zone d'entrée de l'exit.

***MsgOutPtr* (PMQVOID)**
MsgOutPtr est l'adresse d'une mémoire tampon contenant les données de message renvoyées par l'exit. A l'entrée de l'exit, *MsgOutPtr* est null. En retour de l'exit, si la valeur est toujours null, le gestionnaire de files d'attente envoie le message spécifié par *MsgInPtr* , avec la longueur indiquée par *MsgInLength* .

Si l'exit modifie les données de message, utilisez l'une des procédures suivantes:

- Si la longueur des données ne change pas, les données peuvent être modifiées dans la mémoire tampon adressée par *MsgInPtr* . Dans ce cas, ne modifiez pas *MsgOutPtr* et *MsgOutLength*.
- Si les données modifiées sont plus courtes que les données d'origine, les données peuvent être modifiées dans la mémoire tampon adressée par *MsgInPtr* . Dans ce cas, *MsgOutPtr* doit être défini sur l'adresse de la mémoire tampon du message d'entrée et *MsgOutLength* sur la nouvelle longueur des données du message.
- Si les données modifiées sont ou peuvent être plus longues que les données d'origine, l'exit doit obtenir une nouvelle mémoire tampon de messages. Copiez les données modifiées dans le fichier. Définissez *MsgOutPtr* sur l'adresse de la nouvelle mémoire tampon et définissez *MsgOutLength* sur la longueur des nouvelles données de message. L'exit est responsable de la libération de la mémoire tampon adressée par *MsgOutPtr* lors de l'appel suivant de l'exit.

Remarque : *MsgOutPtr* est toujours le pointeur null en entrée de l'exit, et non l'adresse d'une mémoire tampon de messages précédemment obtenue. Pour libérer la mémoire tampon précédemment obtenue, l'exit doit sauvegarder son adresse et sa longueur. Sauvegardez les informations dans *ExitUserArea*ou dans un bloc de contrôle dont l'adresse est sauvegardée dans *ExitUserArea* .

MsgOutPtr est une zone d'entrée-sortie de l'exit.

***MsgOutLength* (MQLONG)**
MsgOutLength est la longueur en octets des données de message renvoyées par l'exit. En entrée de l'exit, cette zone est toujours à zéro. Lors du retour de l'exit, cette zone est ignorée si *MsgOutPtr* est null. Pour plus d'informations sur la modification des données de message, voir *MsgOutPtr* .

MsgOutLength est une zone d'entrée-sortie de l'exit.

***pEntryPoints* (PMQIEP)**
pEntryPoints est l'adresse d'une structure MQIEP via laquelle les appels MQI et DCI peuvent être effectués.

Déclaration de langage C-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-Structure de données de contexte de publication

La structure MQPBC contient les informations contextuelles, relatives au diffuseur de publications de la publication, qui sont transmises à l'exit de publication.

Le Tableau 590, à la page 1034 récapitule les zones de la structure:

Tableau 590. Zones dans MQPBC

Zone	Description
StrucID	Identificateur de structure
Version	Numéro de version de structure
PubTopicString	Publier la chaîne de rubrique
MsgDescPtr	Adresse du descripteur de message (MQMD)

Zones

***StrucID* (MQCHAR4)**
StrucID est l'identificateur de structure. La valeur est la suivante:

MQPBC_STRUCID
MQPBC_STRUCID est l'identificateur de la structure de contexte de publication. Pour le langage de programmation C, la constante MQPBC_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQPBC_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

StrucID est une zone d'entrée de l'exit.

***Version* (MQLONG)**
Version est le numéro de version de la structure. La valeur est la suivante:

MQPBC_VERSION_1
MQPBC_VERSION_1 est la structure des paramètres d'exit de publication version 1.

MQPBC_VERSION_2
MQPBC_VERSION_2 est la structure du paramètre d'exit de publication version 2. La constante MQPBC_CURRENT_VERSION est également définie avec la même valeur.

Version est une zone d'entrée de l'exit.

***PubTopicString* (MQCHARV)**
PubTopicString est la chaîne de rubrique dans laquelle la publication est en cours.

PubTopicString est une zone d'entrée de l'exit.

***MsgDescPtr* (PMQMD)**
MsgDescPtr est l'adresse d'une copie du descripteur de message (MQMD) pour le message en cours de traitement.

MsgDescPtr est une zone d'entrée de l'exit.

Déclaration de langage C-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-Structure de données de contexte d'abonnement

La structure MQSBC contient les informations contextuelles relatives à l'abonné qui reçoit la publication et qui sont transmises à l'exit de publication.

Le Tableau 591, à la page 1035 récapitule les zones de la structure:

Tableau 591. Champs dans MQSBC

Zone	Description
StrucID	Identificateur de structure
Version	Numéro de version de structure
DestinationQMgrName	Nom du gestionnaire de files d'attente de destination
DestinationQName	Nom de la file d'attente de destination
SubType	Type d'abonnement
SubOptions	Options d'abonnement
ObjectName	Nom d'objet
ObjectString	Chaîne d'objet
SubTopicString	Chaîne de rubrique d'abonnement
SubName	Nom de l'abonnement
SubId	Identificateur d'abonnement
SelectionString	Adresse de la chaîne de sélection
SubLevel	Niveau d'abonnement

Tableau 591. Champs dans MQSBC (suite)	
Zone	**Description**
PSProperties	Propriétés de publication / abonnement

Zones

***StrucID* (MQCHAR4)**
Identificateur de structure. La valeur est la suivante:

MQSBC_STRUCID
MQSBC_STRUCID est l'identificateur de la structure des paramètres d'exit de publication. Pour le langage de programmation C, la constante MQSBC_STRUC_ID_ARRAY est également définie ; MQSBC_STRUC_ID_ARRAY a la même valeur que MQSBC_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

StrucID est une zone d'entrée de l'exit.

***Version* (MQLONG)**
Numéro de version de la structure. La valeur est la suivante:

MQSBC_VERSION_1
Structure des paramètres d'exit de publication de la version 1. La constante MQSBC_CURRENT_VERSION est également définie avec la même valeur.

Version est une zone d'entrée de l'exit.

***DestinationQMgrName* (MQCHAR48)**
DestinationQMgrName est le nom du gestionnaire de files d'attente auquel le message est envoyé. Le nom est rempli avec des blancs sur toute la longueur de la zone. Le nom peut être modifié par l'exit. La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH.

DestinationQMgrName est une zone d'entrée-sortie pour l'exit ; voir remarque.

***DestinationQName* (MQCHAR48)**
DestinationQName est le nom de la file d'attente à laquelle le message est envoyé. Le nom est rempli avec des blancs sur toute la longueur de la zone. Le nom peut être modifié par l'exit. La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH.

DestinationQName est une zone d'entrée-sortie pour l'exit ; voir remarque.

***SubType* (MQLONG)**
SubType indique comment l'abonnement a été créé. Les valeurs valides sont MQSUBTYPE_API, MQSUBTYPE_ADMIN et MQSUBTYPE_PROXY; voir Inquire Subscription Status (Response).

SubType est une zone d'entrée de l'exit.

***SubOptions* (MQLONG)**
SubOptions sont les options d'abonnement ; voir «Options (MQLONG)», à la page 549 pour une description des valeurs que cette zone peut prendre.

SubOptions est une zone d'entrée de l'exit.

***ObjectName* (MQCHAR48)**
ObjectName est le nom de l'objet de rubrique tel qu'il est défini sur le gestionnaire de files d'attente local. La longueur de cette zone est indiquée par MQ_TOPIC_NAME_LENGTH. Le nom de l'objet est le nom de l'objet de rubrique d'administration que le gestionnaire de files d'attente a associé à la chaîne de rubrique. Même si l'abonné a fourni un objet de rubrique dans le cadre de l'abonnement, *ObjectName* peut être un objet de rubrique différent. L'association d'un objet de rubrique à un abonnement dépend de la résolution complète de *SubTopicString*.

ObjectName est une zone d'entrée de l'exit.

***ObjectString* (MQCHARV)**
ObjectString est la chaîne de rubrique complète de la publication à laquelle vous avez souscrit. Tous les caractères génériques de la chaîne d'abonnement d'origine sont résolus. Il est différent de la zone MQSD subscription *ObjectString* décrite dans «ObjectString (MQCHARV)», à la page 549, qui peut contenir des caractères génériques et exclut tout nom d'objet fourni par l'abonné.

ObjectString est une zone d'entrée de l'exit.

***SubTopicString* (MQCHARV)**
SubTopicString est la chaîne de rubrique complète fournie par l'abonné. *SubTopicString* est la combinaison de la chaîne de rubrique définie dans un objet de rubrique et d'une chaîne de rubrique. Un abonné doit fournir un objet de rubrique, une chaîne de rubrique ou les deux. Si l'abonné fournit une chaîne de rubrique, elle peut contenir des caractères génériques.

SubTopicString est une zone d'entrée de l'exit.

***SubName* (MQCHARV)**
SubName est le nom d'abonnement fourni par l'abonné ou est un nom généré.

SubName est une zone d'entrée de l'exit.

***SubId* (MQBYTE 24)**
SubId est l'identificateur d'abonnement interne unique.

SubId est une zone d'entrée de l'exit.

***SelectionString* (MQCHARV)**
SelectionString correspond aux critères de sélection utilisés lors de l'abonnement aux messages d'une rubrique ; voirSélecteurs .

SelectionString est une zone d'entrée de l'exit.

***SubLevel* (MQLONG)**
SubLevel est le niveau d'interception associé à l'abonnement ; voir «SubLevel (MQLONG)», à la page 561 pour plus de détails.

SubLevel est une zone d'entrée de l'exit.

***PSProperties* (MQLONG)**
PSProperties sont les propriétés de publication / abonnement. Ils spécifient comment les propriétés de message liées à la publication / abonnement sont ajoutées aux messages envoyés à cet abonnement. Les valeurs possibles sont MQPSPROP_NONE, MQPSPROP_COMPAT, MQPSPROP_RFH2, MQPSPROP_MSGPROP. Pour obtenir une description de ces valeurs, voir Paramètres facultatifs (Modifier, Copier et Créer un abonnement) .

PSProperties est une zone d'entrée de l'exit.

Remarque : Les vérifications d'autorisation ne sont effectuées que sur les valeurs d'origine de *DestinationQMgrName* et *DestinationQName* avant qu'elles ne soient transmises à l'exit de publication. Aucune nouvelle vérification d'autorisation n'est effectuée lorsque l'exit modifie la file d'attente de destination, en modifiant *DestinationQMgrName* ou *DestinationQName*.

Déclaration de langage C-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;
```

Structures de données et appels d'exit de canal

Cette collection de rubriques fournit des informations de référence sur les appels WebSphere MQ et les structures de données que vous pouvez utiliser lorsque vous écrivez des programmes d'exit de canal.

Ces informations sont des informations d'interface de programmation sensibles au produit. Vous pouvez écrire des exits utilisateur WebSphere MQ dans les langages de programmation suivants:

Plateforme	Langages de programmation
WebSphere MQ for z/OS	Assembleur et C (qui doivent être conformes à l'environnement de programmation système C pour les exits système, décrits dans le manuel *z/OS C/C++ Programming Guide*.)
WebSphere MQ pour IBM i	ILE C, ILE COBOL et ILE RPG
Toutes les autres plateformes WebSphere MQ	C

Vous pouvez également écrire des exits utilisateur dans Java à utiliser uniquement avec des applications Java et JMS. Pour plus d'informations sur la création et l'utilisation d'exits de canal avec WebSphere MQ classes for Java, voir Utilisation d'exits de canal dans WebSphere MQ classes for Java et pour WebSphere MQ classes for JMS, voir Utilisation d'exits de canal avec WebSphere MQ classes for JMS.

Vous ne pouvez pas écrire les exits utilisateur WebSphere MQ dans TAL ou Visual Basic. Toutefois, une déclaration pour la structure MQCD est fournie dans Visual Basic pour une utilisation sur l'appel MQCONNX à partir d'un programme client WebSphere MQ MQI.

Dans un certain nombre de cas dans les descriptions qui suivent, les paramètres sont des tableaux ou des chaînes de caractères dont la taille n'est pas fixe. Pour ces paramètres, une minuscule "n" est utilisée pour représenter une constante numérique. Lorsque la déclaration de ce paramètre est codée, la valeur "n" doit être remplacée par la valeur numérique requise. Pour plus d'informations sur les conventions utilisées dans ces descriptions, voir «Types de données élémentaires», à la page 218.

Fichiers de définition de données

Les fichiers de définition de données sont fournis avec WebSphere MQ pour chacun des langages de programmation pris en charge. Pour plus de détails sur ces fichiers, voir Copy, header, include et module files.

MQ_CHANNEL_EXIT-Exit de canal

L'appel MQ_CHANNEL_EXIT décrit les paramètres qui sont transmis à chacun des exits de canal appelés par l'agent MCA.

Aucun point d'entrée appelé MQ_CHANNEL_EXIT n'est fourni par le gestionnaire de files d'attente ; le nom MQ_CHANNEL_EXIT n'a pas de signification particulière car les noms des exits de canal sont fournis dans la définition de canal MQCD.

Il existe cinq types d'exit de canal:

- Exit de sécurité de canal
- Exit de message de canal
- Exit d'émission de canal
- Exit de réception de canal
- Message du canal-exit de relance

Les paramètres sont similaires pour chaque type de sortie, et la description donnée ici s'applique à tous, sauf mention spécifique.

Syntaxe

MQ_CHANNEL_EXIT *(ChannelExitParms, ChannelDefinition, DataLength, AgentBufferLength, AgentBuffer, ExitBufferLength, ExitBufferAddr)*

Paramètres

L'appel MQ_CHANNEL_EXIT possède les paramètres suivants.

ChannelExitParamètres (MQCXP)-entrée / sortie

Bloc de paramètres d'exit de canal.

Cette structure contient des informations supplémentaires relatives à l'appel de l'exit.L'exit définit les informations de cette structure pour indiquer comment l'agent MCA se déroule.

ChannelDefinition (MQCD)-entrée/sortie

Définition de canal.

Cette structure contient des paramètres définis par l'administrateur pour contrôler le comportement du canal.

DataLength (MQLONG)-entrée/sortie

Longueur des données.

Les données dépendent du type d'exit:

- Pour un exit de sécurité de canal, lorsque l'exit est appelé, ce paramètre contient la longueur de tout message de sécurité dans la zone *AgentBuffer* , si *ExitReason* est MQXR_SEC_MSG. Il est égal à zéro s'il n'y a pas de message. L'exit doit définir cette zone sur la longueur de tout message de sécurité à envoyer à son partenaire s'il définit *ExitResponse* sur MQXCC_SEND_SEC_MSG ou MQXCC_SEND_AND_REQUEST_SEC_MSG. Les données de message se trouvent dans *AgentBuffer* ou *ExitBufferAddr*.

  Le contenu des messages de sécurité est la seule responsabilité des exits de sécurité.
- Pour un exit de message de canal, lorsque l'exit est appelé, ce paramètre contient la longueur du message (y compris l'en-tête de la file d'attente de transmission). L'exit doit définir cette zone sur la longueur du message dans *AgentBuffer* ou *ExitBufferAddr* qui doit être traitée. Elle doit être supérieure ou égale à la longueur de l'en-tête de la file d'attente de transmission (MQXQH).
- Pour un exit d'émission ou de réception de canal, lorsque l'exit est appelé, ce paramètre contient la longueur de la transmission. L'exit doit définir cette zone sur la longueur de la transmission dans *AgentBuffer* ou *ExitBufferAddr* qui doit se poursuivre.

Si un exit de sécurité envoie un message et qu'il n'y a pas d'exit de sécurité à l'autre extrémité du canal, ou que l'autre extrémité définit un *ExitResponse* de MQXCC_OK, l'exit initiateur est à nouveau appelé avec MQXR_SEC_MSG et une réponse nulle (*DataLength*= 0).

AgentBufferLength (MQLONG)-entrée

Longueur de la mémoire tampon de l'agent.

Ce paramètre peut être supérieur à *DataLength* lors de l'appel.

Pour les exits de message de canal, d'envoi et de réception, tout espace inutilisé lors de l'appel peut être utilisé par l'exit pour développer les données en place. Dans ce cas, le paramètre *DataLength* doit être défini de manière appropriée par l'exit.

Dans le langage de programmation C, ce paramètre est transmis par adresse.

AgentBuffer (MQBYTE ×AgentBufferLength)-entrée/sortie

Mémoire tampon de l'agent.

Le contenu de ce paramètre dépend du type d'exit:

- Pour un exit de sécurité de canal, lors de l'appel de l'exit, il contient un message de sécurité si *ExitReason* est MQXR_SEC_MSG. Pour renvoyer un message de sécurité, l'exit peut utiliser cette mémoire tampon ou sa propre mémoire tampon (*ExitBufferAddr*).
- Pour un exit de message de canal, lors de l'appel de l'exit, ce paramètre contient:
  - L'en-tête de file d'attente de transmission (MQXQH), qui inclut le descripteur de message (qui contient lui-même les informations de contexte du message), immédiatement suivi par
  - Données de message

  Si le message doit continuer, l'exit peut effectuer l'une des opérations suivantes:

  - Ne pas toucher au contenu de la mémoire tampon
  - Modifier le contenu en place (renvoi de la nouvelle longueur des données dans *DataLength*; cette valeur ne doit pas être supérieure à *AgentBufferLength*)
  - Copiez le contenu dans le *ExitBufferAddr*, en y apportant les modifications requises

  Les modifications apportées par l'exit à l'en-tête de la file d'attente de transmission ne sont pas vérifiées ; toutefois, des modifications erronées peuvent signifier que le message ne peut pas être placé à la destination.
- Pour un exit d'émission ou de réception de canal, lors de l'appel de l'exit, celui-ci contient les données de transmission. L'exit peut effectuer l'une des opérations suivantes:
  - Ne pas toucher au contenu de la mémoire tampon
  - Modifier le contenu en place (renvoi de la nouvelle longueur des données dans *DataLength*; cette valeur ne doit pas être supérieure à *AgentBufferLength*)
  - Copiez le contenu dans le *ExitBufferAddr*, en y apportant les modifications requises

  Les 8 premiers octets des données ne doivent pas être modifiés par l'exit.

ExitBufferLength (MQLONG)-entrée/sortie

Longueur de la mémoire tampon de sortie.

Lors du premier appel de l'exit, ce paramètre est défini sur zéro. Ensuite, toute valeur renvoyée par l'exit, à chaque appel, est présentée à l'exit lors de son prochain appel. La valeur n'est pas utilisée par l'agent MCA.

Remarque : Ce paramètre ne doit pas être utilisé par des exits écrits dans des langages de programmation qui ne prennent pas en charge le type de données de pointeur.

ExitBufferAddr (MQPTR)-entrée/sortie

Adresse de la mémoire tampon de sortie.

Ce paramètre est un pointeur vers l'adresse d'une mémoire tampon gérée par l'exit, où il peut choisir de renvoyer des données de message ou de transmission (en fonction du type d'exit) à l'agent si la mémoire tampon de l'agent est ou peut ne pas être suffisamment grande, ou s'il est plus pratique pour l'exit de le faire.

Lors du premier appel de l'exit, l'adresse transmise à l'exit est null. Par la suite, toute adresse transmise par l'exit, à chaque appel, est présentée à l'exit lors de son prochain appel.

Remarque : Ce paramètre ne doit pas être utilisé par des exits écrits dans des langages de programmation qui ne prennent pas en charge le type de données de pointeur.

Appel C

```
exitname (&ChannelExitParms, &ChannelDefinition,
         &DataLength, &AgentBufferLength, AgentBuffer,
         &ExitBufferLength, &ExitBufferAddr);
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
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 */
```

Appel COBOL

```
     CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION,
                           DATALENGTH, AGENTBUFFERLENGTH, AGENTBUFFER,
                           EXITBUFFERLENGTH, EXITBUFFERADDR.
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
**   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.
```

Appel RPG (ILE)

```
     C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     C                    CALLP     exitname(MQCXP : MQCD : DATLEN :
     C                                       ABUFL : ABUF : EBUFL :
     C                                       EBUF)
```

La définition de prototype pour l'appel est la suivante:

```
     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                            *
```

Appel de l'assembleur System/390

```
         CALL EXITNAME,(CHANNELEXITPARMS,CHANNELDEFINITION,DATALENGTH, X
```

```
                AGENTBUFFERLENGTH,AGENTBUFFER,EXITBUFFERLENGTH,         X
                EXITBUFFERADDR)
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
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
```

Notes d'utilisation

1. La fonction exécutée par l'exit de canal est définie par le fournisseur de l'exit. L'exit, cependant, doit respecter les règles définies ici et dans le bloc de contrôle associé, le MQCXP.
2. Le paramètre *ChannelDefinition* transmis à l'exit de canal peut être l'une des versions suivantes. Pour plus d'informations, voir la zone *Version* dans la structure MQCD.
3. Si l'exit de canal reçoit une structure MQCD avec la zone *Version* définie sur une valeur supérieure à MQCD_VERSION_1, l'exit doit utiliser la zone *ConnectionName* dans MQCD, de préférence à la zone *ShortConnectionName* .
4. En général, les exits de canal sont autorisés à modifier la longueur des données de message. Cela peut être dû à l'ajout de données au message par l'exit, à la suppression de données du message ou à la compression ou au chiffrement du message. Toutefois, des restrictions spéciales s'appliquent si le message est un segment qui ne contient qu'une partie d'un message logique. En particulier, il ne doit pas y avoir de changement net de la longueur du message à la suite des actions des sorties d'émission et de réception complémentaires.

   Par exemple, il est permis à un exit d'émission de raccourcir le message en le compressant, mais l'exit de réception complémentaire doit restaurer la longueur d'origine du message en le décompressant, de sorte qu'il n'y ait pas de changement net de la longueur du message.

   Cette restriction est due au fait que la modification de la longueur d'un segment provoquerait une inexactitude des décalages des segments ultérieurs du message, ce qui empêcherait le gestionnaire de files d'attente de reconnaître que les segments formaient un message logique complet.

MQ_CHANNEL_AUTO_DEF_EXIT-Exit de définition automatique de canal

L'appel MQ_CHANNEL_AUTO_DEF_EXIT décrit les paramètres transmis à l'exit de définition automatique de canal appelé par l'agent MCA.

Aucun point d'entrée appelé MQ_CHANNEL_AUTO_DEF_EXIT n'est fourni par le gestionnaire de files d'attente ; le nom MQ_CHANNEL_AUTO_DEF_EXIT n'a pas de signification particulière car les noms des exits de définition automatique sont fournis dans le gestionnaire de files d'attente.

Syntaxe

MQ_CHANNEL_AUTO_DEF_EXIT *(ChannelExitParms, ChannelDefinition)*

Paramètres

L'appel MQ_CHANNEL_AUTO_DEF_EXIT possède les paramètres suivants.

ChannelExitParamètres (MQCXP)-entrée / sortie

Bloc de paramètres d'exit de canal.

Cette structure contient des informations supplémentaires relatives à l'appel de l'exit.L'exit définit les informations de cette structure pour indiquer comment l'agent MCA se déroule.

ChannelDefinition (MQCD)-entrée/sortie

Définition de canal.

Cette structure contient des paramètres définis par l'administrateur pour contrôler le comportement des canaux créés automatiquement.L'exit définit des informations dans cette structure pour modifier le comportement par défaut défini par l'administrateur.

Les zones MQCD répertoriées ne doivent pas être modifiées par l'exit:

- *ChannelName*
- *ChannelType*
- *StrucLength*
- *Version*

Si d'autres zones sont modifiées, la valeur définie par l'exit doit être valide. Si la valeur n'est pas valide, un message d'erreur est consigné dans le fichier journal des erreurs ou affiché sur la console (selon l'environnement).

Appel C

```
exitname (&ChannelExitParms, &ChannelDefinition);
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
MQCXP  ChannelExitParms;   /* Channel exit parameter block */
MQCD   ChannelDefinition;  /* Channel definition */
```

Appel COBOL

```
     CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION.
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
**   Channel exit parameter block
 01  CHANNELEXITPARMS.
     COPY CMQCXPV.
**   Channel definition
 01  CHANNELDEFINITION.
     COPY CMQCDV.
```

Appel RPG (ILE)

```
     C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     C                   CALLP     exitname(MQCXP : MQCD)
```

La définition de prototype pour l'appel est la suivante:

```
     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
```

Appel de l'assembleur System/390

```
         CALL EXITNAME,(CHANNELEXITPARMS,CHANNELDEFINITION)
```

Les paramètres transmis à l'exit sont déclarés comme suit:

```
CHANNELEXITPARMS   CMQCXPA  ,  Channel exit parameter block
CHANNELDEFINITION  CMQCDA   ,  Channel definition
```

Notes d'utilisation

1. La fonction exécutée par l'exit de canal est définie par le fournisseur de l'exit. L'exit, cependant, doit respecter les règles définies ici et dans le bloc de contrôle associé, le MQCXP.
2. Le paramètre *ChannelExitParms* transmis à l'exit de définition automatique de canal est une structure MQCXP. La version de MQCXP transmise dépend de l'environnement dans lequel l'exit est exécuté ; voir la description de la zone *Version* dans «MQCXP-Paramètre d'exit de canal», à la page 1086 pour plus de détails.
3. Le paramètre *ChannelDefinition* transmis à l'exit de définition automatique de canal est une structure MQCD. La version de MQCD transmise dépend de l'environnement dans lequel l'exit est en cours d'exécution ; voir la description de la zone *Version* dans «MQCD-Définition de canal», à la page 1045 pour plus de détails.

MQXWAIT-Attendre l'exit

L'appel MQXWAIT attend qu'un événement se produise. Il ne peut être utilisé qu'à partir d'un exit de canal sous z/OS.

L'utilisation de MQXWAIT permet d'éviter les problèmes de performances qui pourraient se produire si un exit de canal effectue une action qui provoque une attente. L'événement MQXWAIT en attente est signalé par un bloc de contrôle d'événement (ECB) MVS . L'ECB est décrit dans la description du bloc de contrôle MQXWD.

Syntaxe

MQXWAIT *(Hconn, WaitDesc, CompCode, Reason)*

Paramètres

L'appel MQXWAIT comporte les paramètres suivants.

Hconn (MQHCONN)-entrée

Descripteur de connexion.

Ce descripteur représente la connexion au gestionnaire de files d'attente.La valeur de *Hconn* a été renvoyée par un appel MQCONN précédent émis lors de l'appel identique ou antérieur de l'exit.

WaitDesc (MQXWD)-entrée/sortie

Descripteur d'attente.

Ce paramètre décrit l'événement à attendre.Pour plus de détails sur les zones de cette structure, voir «MQXWD-Descripteur d'attente d'exit», à la page 1100 .

CompCode (MQLONG)-sortie

Code achèvement.

Il s'agit de l'un des codes suivants:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

Cause (MQLONG)-sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* a pour valeur MQCC_OK :

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptateur non disponible.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Options incorrectes ou incohérentes.

MQRC_XWAIT_ANNULE
(2107, X'83B') Appel MQXWAIT annulé.

MQRC_XWAIT_ERREUR
(2108, X'83C') Appel de l'appel MQXWAIT non valide.

Appel C

```
MQXWAIT (Hconn, &WaitDesc, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONN  Hconn;     /* Connection handle */
MQXWD    WaitDesc;  /* Wait descriptor */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

Appel de l'assembleur System/390

```
         CALL MQXWAIT,(HCONN,WAITDESC,COMPCODE,REASON)
```

Déclarez les paramètres comme suit :

```
HCONN     DS       F  Connection handle
WAITDESC  CMQXWDA  ,  Wait descriptor
COMPCODE  DS       F  Completion code
REASON    DS       F  Reason code qualifying COMPCODE
```

MQCD-Définition de canal

La structure MQCD contient les paramètres qui contrôlent l'exécution d'un canal. Il est transmis à chaque exit de canal appelé à partir d'un agent MCA (Message Channel Agent).

Pour plus d'informations sur les exits de canal, voir «MQ_CHANNEL_EXIT-Exit de canal», à la page 1038. La description de cette rubrique concerne à la fois les canaux de transmission de messages et les canaux MQI.

Zones de nom de sortie

Lorsqu'un exit est appelé, la zone appropriée de *SecurityExit*, *MsgExit*, *SendExit*, *ReceiveExit*et *MsgRetryExit* contient le nom de l'exit actuellement appelé. La signification du nom dans ces zones

dépend de l'environnement dans lequel l'agent MCA s'exécute. Sauf indication contraire, le nom est aligné à gauche dans la zone, sans blancs imbriqués ; le nom est complété par des blancs à la longueur de la zone. Dans les descriptions qui suivent, les crochets ([ ]) indiquent des informations facultatives:

Systèmes UNIX
Le nom de l'exit est le nom d'un module ou d'une bibliothèque chargeable dynamiquement, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire:

```
[path]library(function)
```

Le nom est limité à un maximum de 128 caractères.

z/OS
Le nom d'exit est le nom d'un module de chargement qui est valide pour la spécification sur le paramètre EP de la macro LINK ou LOAD. Le nom est limité à un maximum de huit caractères.

Windows
Le nom d'exit est le nom d'une bibliothèque de liens dynamiques, suffixé avec le nom d'une fonction résidant dans cette bibliothèque. Le nom de la fonction doit être placé entre parenthèses. Le nom de la bibliothèque peut éventuellement être précédé d'un chemin de répertoire et d'une unité:

```
[d:][path]library(function)
```

Le nom est limité à un maximum de 128 caractères.

IBM i
Le nom de l'exit est un nom de programme de 10 octets suivi d'un nom de bibliothèque de 10 octets. Si la longueur des noms est inférieure à 10 octets, chaque nom est complété par des blancs pour en faire 10 octets. Le nom de la bibliothèque peut être `*LIBL` sauf lors de l'appel d'un exit de définition automatique de canal, auquel cas un nom qualifié complet est requis.

Modification des zones MQCD dans un exit de canal

Un exit de canal peut modifier les zones de la base de données MQCD. La valeur modifiée reste dans le MQCD et est transmise à tous les exits restants d'une chaîne d'exit et à toute conversation partageant l'instance de canal. Le MQCD modifié est également utilisé par l'agent MCA pour son traitement normal pendant la durée de vie continue du canal.

Les zones MQCD suivantes ne doivent pas être modifiées par l'exit:

- ChannelName
- ChannelType
- StrucLength
- Version

Référence associée
«Zones», à la page 1047
Cette rubrique répertorie toutes les zones de la structure MQCD et décrit chaque zone.

«Déclaration C», à la page 1073
Cette déclaration est la déclaration C pour la structure MQCD.

«Déclaration COBOL», à la page 1075
Cette déclaration est la déclaration COBOL pour la structure MQCD.

«Déclaration RPG (ILE)», à la page 1077
Cette déclaration est la déclaration RPG pour la structure MQCD.

«Déclaration assembleur System/390», à la page 1080

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQCD.

«Déclaration Visual Basic», à la page 1081
Cette déclaration est la déclaration Visual Basic de la structure MQCD.

«Modification des zones MQCD dans un exit de canal», à la page 1083
Un exit de canal peut modifier les zones de la base de données MQCD. Toutefois, ces modifications ne sont généralement pas prises en compte, sauf dans les circonstances indiquées.

Zones

Cette rubrique répertorie toutes les zones de la structure MQCD et décrit chaque zone.

BatchHeartbeat (MQLONG)
Cette zone indique l'intervalle de temps utilisé pour déclencher un signal de présence de lot pour le canal.

La pulsation par lots permet aux canaux émetteurs de déterminer si l'instance de canal distant est toujours active avant de passer en attente de validation. Un signal de présence par lots se produit si un canal émetteur n'a pas communiqué avec l'instance de canal distant dans l'intervalle de temps spécifié.

La valeur est comprise entre 0 et 999 999 ; les unités sont des millisecondes. La valeur zéro indique que le signal de présence du lot n'est pas activé.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_7.

BatchInterval (MQLONG)
Cette zone indique la durée approximative, en millisecondes, pendant laquelle un canal maintient un lot ouvert, si moins de *BatchSize* messages ont été transmis dans le lot en cours.

Si *BatchInterval* est supérieur à zéro, le lot est arrêté par l'un des événements suivants qui se produit en premier:

- Des messages *BatchSize* ont été envoyés ou
- *BatchInterval* millisecondes se sont écoulées depuis le démarrage du lot.

Si *BatchInterval* est égal à zéro, le lot est arrêté en premier lieu par l'un des événements suivants:

- Des messages *BatchSize* ont été envoyés ou
- la file d'attente de transmission devient vide.

BatchInterval doit être compris entre zéro et 999 999 999.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente lorsque *Version* est inférieur à MQCD_VERSION_4.

BatchSize (MQLONG)
Cette zone indique le nombre maximal de messages pouvant être envoyés via un canal avant la synchronisation du canal.

Cette zone n'est pas pertinente pour les canaux dont le *ChannelType* est MQCHT_SVRCONN ou MQCHT_CLNTCONN.

ChannelMonitoring (MQLONG)
Cette zone indique le niveau en cours de la collecte de données de surveillance pour le canal.

Cette zone n'est pas pertinente pour les canaux dont le ChannelType est MQCHT_CLNTCONN.

Ses valeurs sont l'une des suivantes :

- MQMON_OFF

- MQMON_FAIBLE
- MQMON_MEDIUM
- MQMON_ELEVE

Il s'agit d'une zone d'entrée de l'exit. Elle n'est pas présente si *Version* est inférieur à MQCD_VERSION_8.

ChannelName (MQCHAR20)
Cette zone indique le nom de la définition de canal.

Il doit exister une définition de canal du même nom sur la machine distante pour pouvoir communiquer.

Le nom ne doit utiliser que les caractères suivants:

- Majuscules A-Z
- Minuscules a-z
- Chiffres 0-9
- le point (.)
- la barre oblique (/)
- le caractère de soulignement (_)
- le symbole de pourcentage (%)

et être rempli à droite avec des blancs. Les espaces de début ou imbriqués ne sont pas autorisés.

La longueur de cette zone est donnée par MQ_CHANNEL_NAME_LENGTH.

ChannelStatistics (MQLONG)
Cette zone indique le niveau actuel de collecte de données statistiques pour le canal.

Cette zone n'est pas pertinente pour les canaux dont le ChannelType est MQCHT_CLNTCONN.

Ses valeurs sont l'une des suivantes :

- MQMON_OFF
- MQMON_FAIBLE
- MQMON_MEDIUM
- MQMON_ELEVE

Il s'agit d'une zone d'entrée de l'exit. Elle n'est pas présente si *Version* est inférieur à MQCD_VERSION_8.

ChannelType (MQLONG)
Cette zone indique le type de canal.

Ses valeurs sont l'une des suivantes :

EXPÉDITEUR_MQCH
Expéditeur.

SERVEUR_MQ
Serveur.

MQCHT_RECEIVER
Récepteur.

DEMANDE_MQCHT_DEMANDEUR
Demandeur.

MQCHT_CLNTCONN
Connexion client.

MQCHT_SVRCONN
Connexion serveur (à utiliser par les clients).

MQCHT_CLUSSDR
Emetteur de cluster.

MQCHT_CLUSRCVR
Récepteur de cluster.

Pondération ClientChannel(MQLONG)
Cette zone indique une pondération qui influence la définition de canal de connexion client utilisée.

L'attribut de pondération ClientChannelest utilisé pour que les définitions de canal du client puissent être sélectionnées de manière aléatoire en fonction de leur pondération lorsque plusieurs définitions appropriées sont disponibles. Lorsqu'un client émet une demande de connexion MQCONN à un groupe de gestionnaires de files d'attente en spécifiant un nom de gestionnaire de files d'attente commençant par un astérisque et que plusieurs définitions de canal appropriées sont disponibles dans la table de définition de canal du client (CCDT), la définition à utiliser est sélectionnée de manière aléatoire en fonction de la pondération, avec les définitions ClientChannel(0) applicables sélectionnées en premier par ordre alphabétique.

Spécifiez une valeur comprise entre 0 et 99. La valeur par défaut est 0.

Elle indique qu'aucun équilibrage de charge n'est effectué et que les définitions applicables sont sélectionnées par ordre alphabétique. Pour autoriser un équilibrage de charge, choisissez une valeur comprise entre 1 et 99, où 1 est la pondération la plus faible et 99 la plus élevée. La répartition des messages entre deux ou plusieurs canaux avec des pondérations non nulles est proportionnelle au rapport de ces pondérations. Par exemple, trois canaux avec des valeurs de pondération ClientChannelde 2, 4 et 14 sont sélectionnés à environ 10%, 20% et 70% du temps. Cette distribution n'est pas garantie.

Cet attribut est valide uniquement pour le type de canal de connexion client.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieure à MQCD_VERSION_9.

ClusterPtr (MQPTR)
Cette zone indique l'adresse d'une liste de noms de cluster.

Si *ClustersDefined* est supérieur à zéro, cette adresse correspond à l'adresse d'une liste de noms de cluster.Le canal appartient à chaque cluster répertorié.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_5.

ClustersDefined (MQLONG)
Cette zone indique le nombre de clusters auxquels appartient le canal.

Cette zone indique le nombre de noms de cluster indiqués par *ClusterPtr*.Elle est supérieure ou égale à zéro.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_5.

CLWLChannelPriority (MQLONG)
Cette zone indique la priorité du canal de charge de travail du cluster.

L'algorithme de sélection du gestionnaire de charge de travail sélectionne une destination dont la priorité est la plus élevée dans l'ensemble de destinations sélectionnées en fonction du rang. S'il existe deux gestionnaires de files d'attente de destination possibles, cet attribut peut être utilisé pour effectuer une reprise en ligne d'un gestionnaire de files d'attente sur l'autre gestionnaire de files d'attente. Tous les messages sont envoyés au gestionnaire de files d'attente avec la priorité la plus élevée jusqu'à la fin, puis les messages sont envoyés au gestionnaire de files d'attente avec la priorité la plus élevée suivante.

La valeur est comprise entre 0 et 9. La valeur par défaut est 0.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_8.

Pour plus d'informations, voir Configuration d'un cluster de gestionnaires de files d'attente.

CLWLChannelRank (MQLONG)
Cette zone indique le rang du canal de charge de travail du cluster.

L'algorithme de sélection du gestionnaire de charge de travail sélectionne une destination avec le rang le plus élevé. Lorsque la destination finale est un gestionnaire de files d'attente sur un autre cluster, vous pouvez définir le rang des gestionnaires de files d'attente de passerelle intermédiaires (à l'intersection des clusters voisins) de sorte que l'algorithme de sélection choisisse correctement un gestionnaire de files d'attente de destination plus proche de la destination finale.

La valeur est comprise entre 0 et 9. La valeur par défaut est 0.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_8.

Pour plus d'informations, voir Configuration d'un cluster de gestionnaires de files d'attente .

CLWLChannelWeight (MQLONG)
Cette zone indique le poids du canal de charge de travail du cluster.

Poids du canal de charge de travail du cluster.

L'algorithme de sélection du gestionnaire de charge de travail utilise l'attribut "weight" du canal pour fausser le choix de destination afin que davantage de messages puissent être envoyés à une machine particulière. Par exemple, vous pouvez donner à un canal sur un grand serveur UNIX un "poids" plus grand qu'un autre canal sur un petit PC de bureau, et l'algorithme de sélection choisit le serveur UNIX plus souvent que le PC.

La valeur est comprise entre 1 et 99. La valeur par défaut est 50.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_8.

Pour plus d'informations, voir Configuration d'un cluster de gestionnaires de files d'attente.

ConnectionAffinity (MQLONG)
Cette zone indique si les applications client qui se connectent plusieurs fois à l'aide du même nom de gestionnaire de files d'attente utilisent le même canal client.

Utilisez cet attribut lorsque plusieurs définitions de canal applicables sont disponibles.

La valeur est l'une des suivantes :

MQCAFTY_PREFERENTE_PRÉFÉRÉE
La première connexion dans un processus de lecture d'une table de définition de canal du client (CCDT) crée une liste de définitions applicables en fonction de la pondération avec les éventuelles définitions CLNTWGHT (0) applicables en premier et par ordre alphabétique. Chaque connexion du processus tente de se connecter en utilisant la première définition de la liste. Si la connexion échoue, la définition suivante est utilisée. Les définitions ayant échoué avec des valeurs CLNTWGHT autres que 0 sont déplacées vers la fin de la liste. Les définitons CLNTWGHT(0) restent en début de liste et sont sélectionnées en premier pour chaque connexion.

Chaque processus client portant le même nom d'hôte crée toujours la même liste.

Pour les applications client écrites en C, C + + ou .NET programming framework (y compris .NET entièrement géré), la liste est mise à jour si la CCDT a été modifiée depuis la création de la liste.

Cette valeur est la valeur par défaut.

MQCAFTY_NONE
La première connexion dans un processus de lecture d'une table de définition de canal du client (CCDT) crée une liste de définitions applicables. Toutes les connexions dans un processus sélectionnent une définition applicable en fonction de la pondération avec toute définition applicable CLNTWGHT(0) sélectionnée en premier et dans l'ordre alphabétique.

Pour les applications client écrites en C, C + + ou .NET programming framework (y compris .NET entièrement géré), la liste est mise à jour si la CCDT a été modifiée depuis la création de la liste.

Cet attribut est valide uniquement pour le type de canal de connexion client.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieure à MQCD_VERSION_9.

ConnectionName (MQCHAR264)
Cette zone indique le nom de connexion du canal.

Pour les canaux récepteurs de cluster (lorsqu'ils sont spécifiés), CONNAME est lié au gestionnaire de files d'attente local, et pour les autres canaux, il est lié au gestionnaire de files d'attente cible. La valeur que vous spécifiez dépend du protocole de transmission (*TransportType*) à utiliser:

- Pour MQXPT_LU62, il s'agit du nom qualifié complet de l'unité logique partenaire.
- Pour MQXPT_NETBIOS, il s'agit du nom NetBIOS défini sur la machine distante.
- Pour MQXPT_TCP, il s'agit soit du nom d'hôte, soit de l'adresse réseau de la machine distante spécifiée au format IPv4 en notation décimale à point ou au format hexadécimal IPv6 , soit de la machine locale pour les canaux récepteurs de cluster.
- Pour MQXPT_SPX, il s'agit d'une adresse de style SPX comprenant une adresse réseau de 4 octets, une adresse de noeud de 6 octets et un numéro de socket de 2 octets.

Lors de la définition d'un canal, cette zone n'est pas pertinente pour les canaux dont le *ChannelType* est MQCHT_SVRCONN ou MQCHT_RECEIVER. Toutefois, lorsque la définition de canal est transmise à un exit, cette zone contient l'adresse du partenaire, quel que soit le type de canal.

La longueur de cette zone est indiquée par MQ_CONN_NAME_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_2.

DataConversion (MQLONG)
Cette zone indique si l'agent de canal de transmission de messages tente de convertir les données de message d'application si l'agent de canal de réception de messages ne parvient pas à effectuer cette conversion.

Cette zone s'applique uniquement aux messages qui ne sont pas des segments de messages logiques ; l'agent MCA ne tente jamais de convertir des messages qui sont des segments.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR. Il peut s'agir de :

MQCDC_SENDER_CONVERSION
Conversion par expéditeur.

MQCDC_NO_SENDER_CONVERSION
Pas de conversion par l'expéditeur.

DefReconnect (MQLONG)
L'attribut de canal `DefReconnect` définit la valeur d'attribut de reconnexion par défaut pour un canal de connexion client.

Option de reconnexion automatique du client par défaut. Vous pouvez configurer un IBM WebSphere MQ MQI client pour une reconnexion automatique à une application client. Le IBM WebSphere MQ MQI client essaie de se reconnecter à un gestionnaire de files d'attente après un échec de connexion. Il essaie de se reconnecter sans que l'application client émette un appel MQI `MQCONN` ou `MQCONNX`.

La reconnexion est une option MQCONNX . A l'aide de l'attribut de canal DefReconnect , vous pouvez ajouter un comportement de reconnexion à des applications existantes qui utilisent MQCONN. Vous pouvez également modifier le comportement de reconnexion des applications qui utilisent MQCONNX.

Vous pouvez également définir la valeur DefRecon à partir du fichier mqclient.ini pour définir ou modifier le comportement de reconnexion. La valeur DefRecon du fichier mqclient.ini est prioritaire sur l'attribut de canal DefReconnect .

Syntax

```
DefReconnect( MQRCN_NO|MQRCN_YES|MQRCN_Q_MGR|
MQRCN_DISABLED)
```

Paramètres

MQRCN_NO
MQRCN_NO est la valeur par défaut.

A moins d'être remplacé par MQCONNX, le client n'est pas reconnecté automatiquement.

MQRCN_YES
A moins d'être remplacé par MQCONNX, le client se reconnecte automatiquement.

MQRCN_Q_MGR
A moins d'être remplacé par MQCONNX, le client se reconnecte automatiquement, mais uniquement au même gestionnaire de files d'attente. L'option QMGR a le même effet que MQCNO_RECONNECT_Q_MGR.

MQRCN_DISABLED
La reconnexion est désactivée, même si elle est demandée par le programme client à l'aide de l'appel MQCONNX MQI.

La reconnexion automatique du client n'est pas prise en charge par les classes IBM WebSphere MQ pour Java.

Tableau 592. La reconnexion automatique dépend des valeurs définies dans l'application et dans la définition de canal

DefReconnect	**Options de reconnexion définies dans l'application**			
	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

Concepts associés
reconnexion client automatique
Reconnexion canal et client
Strophe CHANNELS du fichier de configuration client
Référence associée
Options de connexion
Options qui contrôlent l'action de MQCONNX.

Desc (MQCHAR64)
Cette zone peut être utilisée pour les commentaires descriptifs.

Le contenu de la zone n'est pas significatif pour les agents de canal de message. Toutefois, il ne doit contenir que des caractères pouvant être affichés. Elle ne peut pas contenir de caractères nuls ; si

nécessaire, elle est remplie à droite avec des blancs. Dans une installation DBCS, la zone peut contenir des caractères DBCS (avec une longueur de zone maximale de 64 octets).

Remarque : Si cette zone contient des caractères qui ne font pas partie du jeu de caractères du gestionnaire de files d'attente (tel que défini par l'attribut de gestionnaire de files d'attente *CodedCharSetId*), ces caractères peuvent être convertis de manière incorrecte si cette zone est envoyée à un autre gestionnaire de files d'attente.

La longueur de cette zone est donnée par MQ_CHANNEL_DESC_LENGTH.

DiscInterval (MQLONG)
Cette zone indique la durée maximale, en secondes, pendant laquelle le canal attend l'arrivée d'un message dans la file d'attente de transmission avant d'arrêter le canal.

En d'autres termes, il spécifie l'intervalle de déconnexion.

La valeur A de zéro entraîne une attente indéfinie de l'agent MCA.

Pour les canaux de connexion serveur utilisant le protocole TCP, l'intervalle représente la valeur de déconnexion d'inactivité du client, en secondes. Si une connexion serveur n'a pas reçu de communication de la part de son client partenaire pendant cette durée, elle met fin à la connexion. L'intervalle d'inactivité de la connexion serveur s'applique uniquement entre les appels d'API WebSphere MQ d'un client. Par conséquent, aucun client n'est déconnecté lors d'un appel MQGET à exécution longue avec attente.

Cet attribut n'est pas applicable aux canaux de connexion serveur utilisant des protocoles autres que TCP.

Cette zone est pertinente uniquement pour les canaux avec un *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR, MQCHT_CLUSRCVR ou MQCHT_SVRCONN.

ExitDataLongueur (MQLONG)
Cette zone indique la longueur en octets de chaque élément de données utilisateur dans les listes d'éléments de données utilisateur d'exit traités par les zones *MsgUserDataPtr*, *SendUserDataPtr*et *ReceiveUserDataPtr* .

Cette longueur n'est pas nécessairement identique à MQ_EXIT_DATA_LENGTH.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

ExitNameLongueur (MQLONG)
Cette zone indique la longueur en octets de chacun des noms figurant dans les listes de noms d'exit adressées par les zones *MsgExitPtr*, *SendExitPtr*et *ReceiveExitPtr* .

Cette longueur n'est pas nécessairement identique à MQ_EXIT_NAME_LENGTH.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

Liste HdrComp[2] (MQLONG)
Cette zone indique la liste des techniques de compression de données d'en-tête prises en charge par le canal.

La liste contient une ou plusieurs des valeurs suivantes:

MQCOMPRESS_NONE
Aucune compression de données d'en-tête n'est effectuée.

MQCOMPRESS_SYSTEM
La compression de données d'en-tête est effectuée.

Les valeurs inutilisées dans le tableau sont définies sur MQCOMPRESS_NOT_AVAILABLE.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_8.

HeartbeatInterval (MQLONG)
Cette zone indique la durée en secondes entre les flux de signaux de présence.

L'interprétation de cette zone dépend du type de canal, comme suit:

- Pour un type de canal MQCHT_SENDER, MQCHT_SERVER, MQCHT_RECEIVER MQCHT_REQUESTER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR, cette zone correspond à la durée, en secondes, entre les flux de pulsations transmis à partir de l'agent MCA émetteur lorsqu'il n'y a pas de message dans la file d'attente de transmission. Cela donne à l'agent MCA récepteur la possibilité de mettre le canal au repos. Pour être utile, *HeartbeatInterval* doit être inférieur à *DiscInterval*.
- Pour un type de canal MQCHT_CLNTCONN ou MQCHT_SVRCONN avec la zone Conversations de partage MQCD définie sur zéro, cette zone correspond à la durée, en secondes, entre les flux de pulsations transmis à partir de l'agent MCA serveur lorsque cet agent MCA a émis un appel MQGET avec l'option MQGMO_WAIT pour le compte d'une application client. Cela permet à l'agent MCA du serveur de gérer les situations où la connexion client échoue lors d'une opération MQGET avec MQGMO_WAIT.
- Pour un type de canal MQCHT_CLNTCONN ou MQCHT_SVRCONN avec la zone Conversations de partage MQCD définie sur une valeur différente de zéro, cette zone correspond à la durée, en secondes, entre deux flux de pulsations lorsqu'aucun flux de données n'est envoyé ou reçu. Cela permet au canal d'être mis au repos efficacement.

La valeur est comprise entre 0 et 999 999. La valeur utilisée est la plus grande des valeurs spécifiées côté émetteur et côté récepteur, sauf si la valeur 0 est spécifiée de part et d'autre, auquel cas aucun échange de signal de présence n'a lieu.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

KeepAliveIntervalle (MQLONG)
Cette zone indique la valeur transmise à la pile de communications pour la temporisation du signal de présence pour le canal.

La valeur est applicable pour les protocoles de communication TCP/IP et SPX, bien que toutes les implémentations ne prennent pas en charge ce paramètre.

La valeur est comprise entre 0 et 99 999 ; les unités sont des secondes. La valeur zéro indique que le signal de présence du canal n'est pas activé, bien que le signal de présence puisse encore se produire si le signal de présence TCP/IP (plutôt que le signal de présence du canal) est activé. La valeur spéciale suivante est également admise:

MQKAI_AUTO
Automatique.

Cette valeur indique que l'intervalle de signal de présence est calculé à partir de l'intervalle de signal de présence négocié, comme suit:

- Si l'intervalle des pulsations négocié est supérieur à zéro, l'intervalle des pulsations utilisé est l'intervalle des pulsations plus 60 secondes.
- Si l'intervalle des signaux de présence négocié est égal à zéro, l'intervalle des signaux de présence utilisé est égal à zéro.

- Sous z/OS, le signal de présence TCP/IP se produit lorsque TCPKEEP (YES) est spécifié sur l'objet gestionnaire de files d'attente.
- Dans d'autres environnements, le signal de présence TCP/IP se produit lorsque le paramètre KEEPALIVE=YES est spécifié dans la section TCP du fichier de configuration de la mise en file d'attente répartie.

Cette zone est pertinente uniquement pour les canaux dont le *TransportType* est MQXPT_TCP ou MQXPT_SPX.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_7.

LocalAddress (MQCHAR48)
Cette zone indique l'adresse TCP/IP locale définie pour le canal pour les communications sortantes.

Cette zone est vide si aucune adresse spécifique n'est définie pour les communications sortantes.L'adresse peut éventuellement inclure un numéro de port ou une plage de numéros de port. Le format de cette adresse est le suivant:

```
[ip-addr][(low-port[,high-port])]
```

où les crochets ([ ]) désignent des informations facultatives, `ip-addr` est spécifié en notation décimale à point IPv4 , IPv6 hexadécimale ou alphanumérique, et `low-port` et `high-port` sont des numéros de port entre parenthèses. Tous sont facultatifs.

Une adresse IP, un port ou une plage de ports spécifique pour les communications sortantes est utile dans les scénarios de récupération où un canal est redémarré sur une pile TCP/IP différente.

La forme de *LocalAddress* est similaire à *ConnectionName*, mais ne doit pas être confondue avec celle-ci. *LocalAddress* indique les caractéristiques des communications locales, tandis que *ConnectionName* indique comment atteindre un gestionnaire de files d'attente éloignées.

Cette zone est pertinente uniquement pour les canaux avec un *TransportType* de MQXPT_TCP et un *ChannelType* de MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER, MQCHT_CLNTCONN, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

La longueur de cette zone est indiquée par MQ_LOCAL_ADDRESS_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_7.

LongMCAUserIdLength (MQLONG)
Cette zone indique la longueur en octets de l'identificateur utilisateur MCA complet désigné par *LongMCAUserIdPtr*.

Cette zone n'est pas pertinente pour les canaux avec un *ChannelType* de MQCHT_CLNTCONN.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_6.

LongMCAUserIdPtr (MQPTR)
Cette zone indique l'adresse de l'ID utilisateur MCA long.

Si *LongMCAUserIdLength* est supérieur à zéro, cette zone correspond à l'adresse de l'ID utilisateur MCA complet.La longueur de l'identificateur complet est indiquée par *LongMCAUserIdLength*. Les 12 premiers octets de l'identificateur utilisateur MCA sont également contenus dans la zone *MCAUserIdentifier*.

Pour plus de détails sur l'ID utilisateur MCA, reportez-vous à la description de la zone *MCAUserIdentifier* .

Cette zone n'est pas pertinente pour les canaux avec un *ChannelType* de MQCHT_SDR, MQCHT_SVR, MQCHT_CLNTCONN ou MQCHT_CLUSSDR.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_6.

LongRemoteUserIdLongueur (MQLONG)
Cette zone indique la longueur en octets de l'ID utilisateur distant complet désigné par *LongRemoteUserIdPtr*.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_CLNTCONN ou MQCHT_SVRCONN.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_6.

LongRemoteUserIdPtr (MQPTR)
Cette zone indique l'adresse de l'ID utilisateur éloigné long.

Si *LongRemoteUserIdLength* est supérieur à zéro, cet indicateur correspond à l'adresse de l'ID utilisateur distant complet.La longueur de l'identificateur complet est indiquée par *LongRemoteUserIdLength*. Les 12 premiers octets de l'ID utilisateur distant sont également contenus dans la zone *RemoteUserIdentifier*.

Pour plus de détails sur l'ID utilisateur distant, voir la description de la zone *RemoteUserIdentifier* .

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_CLNTCONN ou MQCHT_SVRCONN.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_6.

LongRetry(MQLONG)
Cette zone indique le nombre utilisé après épuisement du nombre indiqué par *ShortRetryCount* .

Il indique le nombre maximal de nouvelles tentatives de connexion à la machine distante, à des intervalles définis par *LongRetryInterval*, avant la consignation d'une erreur à l'opérateur.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

Intervalle LongRetry(MQLONG)
Cette zone indique le nombre maximal de secondes à attendre avant de tenter à nouveau la connexion à la machine distante.

L'intervalle entre les nouvelles tentatives peut être étendu si le canal doit attendre pour devenir actif.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

MaxInstances (MQLONG)
Cette zone indique le nombre maximal d'instances simultanées d'un canal de connexion serveur individuel pouvant être démarrées.

Cette zone est utilisée uniquement sur les canaux de connexion serveur.

La valeur de la zone peut être comprise entre 0 et 999 999 999. La valeur 0 (zéro) empêche tout accès client.

La valeur par défaut de cette zone est 999 999 999.

Si la valeur de cette zone est réduite à un nombre inférieur au nombre d'instances du canal de connexion serveur en cours d'exécution, les instances en cours d'exécution ne sont pas affectées. Toutefois, les nouvelles instances ne peuvent pas démarrer tant qu'un nombre suffisant d'instances existantes n'ont pas cessé de s'exécuter, de sorte que le nombre d'instances en cours d'exécution est inférieur à la valeur de la zone.

MaxInstancesPerClient (MQLONG)
Cette zone indique le nombre maximal d'instances simultanées d'un canal de connexion serveur individuel pouvant être démarrées à partir d'un seul client.

Dans ce contexte, les connexions émanant de la même adresse réseau distante peuvent être considérées comme étant issues du même client.

Cette zone est utilisée uniquement sur les canaux de connexion serveur.

La valeur de la zone peut être comprise entre 0 et 999 999 999. La valeur 0 (zéro) empêche tout accès client.

La valeur par défaut de cette zone est 999 999 999.

Si la valeur de cette zone est réduite à un nombre inférieur au nombre d'instances du canal de connexion serveur en cours d'exécution à partir de clients individuels, les instances en cours d'exécution ne sont pas affectées. Toutefois, les nouvelles instances de l'un de ces clients ne peuvent pas démarrer tant qu'un nombre suffisant d'instances existantes n'ont pas cessé de s'exécuter, de sorte que le nombre d'instances en cours d'exécution, provenant du client qui tente d'en démarrer une nouvelle, est inférieur à la valeur de la zone.

Longueur MaxMsg(MQLONG)
Cette zone indique la longueur maximale des messages pouvant être transmis sur le canal.

Cette valeur est comparée à celle du canal éloigné et la valeur la plus faible des deux est le maximum réel.

MCAName (MQCHAR20)
Cette zone est une zone réservée.

La valeur de cette zone est vide.

La longueur de cette zone est indiquée par MQ_MCA_NAME_LENGTH.

MCASecurityId (MQBYTE40)
Cette zone indique l'identificateur de sécurité de l'agent MCA.

Cette zone n'est pas pertinente pour les canaux avec un *ChannelType* de MQCHT_CLNTCONN.

La valeur spéciale suivante indique qu'il n'y a pas d'identificateur de sécurité:

MQSID_AUCUN
Aucun identificateur de sécurité n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQSID_NONE_ARRAY est également définie ; cette constante a la même valeur que MQSID_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La longueur de cette zone est indiquée par MQ_SECURITY_ID_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_6.

MCAType (MQLONG)
Cette zone indique le type de programme d'agent MCA.

Cette zone s'applique uniquement aux canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

La valeur est l'une des suivantes :

PROCESSUS MQMCAT_PROCESS
.

L'agent MCA fonctionne en tant que processus distinct.

MQMCAT_THREAD
Unité d'exécution (IBM i, UNIXet Windows).

L'agent MCA fonctionne en tant qu'unité d'exécution distincte.

Cette zone n'est pas présente lorsque *Version* est inférieure à MQCD_VERSION_2.

MCAUserIdentifier (MQCHAR12)
Cette zone indique l'ID utilisateur de l'agent MCA.

Cette zone utilise les 12 premiers octets de l'identificateur utilisateur MCA et peut être définie par un agent de sécurité.

Deux zones contiennent l'identificateur utilisateur MCA:

- *MCAUserIdentifier* contient les 12 premiers octets de l'identificateur utilisateur MCA et est rempli avec des blancs si l'identificateur est inférieur à 12 octets. *MCAUserIdentifier* peut être vide.
- *LongMCAUserIdPtr* pointe vers l'ID utilisateur MCA complet, qui peut dépasser 12 octets. Sa longueur est donnée par *LongMCAUserIdLength*. L'identificateur complet ne contient pas de blancs de fin et n'est pas terminé par une valeur NULL. Si l'identificateur est vide, *LongMCAUserIdLength* est égal à zéro et la valeur de *LongMCAUserIdPtr* n'est pas définie.

  Remarque : *LongMCAUserIdPtr* n'est pas présent si *Version* est inférieur à MQCD_VERSION_6.

Si l'ID utilisateur MCA n'est pas vide, il indique l'ID utilisateur à utiliser par l'agent MCA pour l'autorisation d'accès aux ressources WebSphere MQ . Pour les types de canal MQCHT_REQUESTER, MQCHT_RECEIVER et MQCHT_CLUSRCVR, si PutAuthority est MQPA_DEFAULT, il s'agit de l'identificateur utilisateur utilisé pour les vérifications d'autorisation de l'opération d'insertion dans les files d'attente de destination.

Si l'ID utilisateur MCA est vide, l'agent MCA utilise son ID utilisateur par défaut.

L'ID utilisateur MCA peut être défini par un exit de sécurité pour indiquer l'ID utilisateur que l'agent MCA doit utiliser. L'exit peut modifier *MCAUserIdentifier*ou la chaîne pointée par *LongMCAUserIdPtr*. Si les deux sont modifiés mais diffèrent l'un de l'autre, l'agent MCA utilise *LongMCAUserIdPtr* de préférence à *MCAUserIdentifier*. Si l'exit modifie la longueur de la chaîne adressée par *LongMCAUserIdPtr*, *LongMCAUserIdLength* doit être défini en conséquence. Si l'exit augmente la longueur de l'identificateur, il doit allouer de la mémoire de la longueur requise, définir cette mémoire sur l'identificateur requis et placer l'adresse de cette mémoire dans *LongMCAUserIdPtr*. L'exit est responsable de la libération de cette mémoire lorsque l'exit est appelé ultérieurement avec la raison MQXR_TERM.

Pour les canaux avec un *ChannelType* de MQCHT_SVRCONN, si *MCAUserIdentifier* dans la définition de canal est vide, tout identificateur utilisateur transféré à partir du client est copié dans le canal. Cet identificateur utilisateur (après toute modification apportée par l'exit de sécurité sur le serveur) est celui sous lequel l'application client est supposée s'exécuter.

L'identificateur utilisateur MCA n'est pas pertinent pour les canaux avec un *ChannelType* de MQCHT_SDR, MQCHT_SVR, MQCHT_CLNTCONN, MQCHT_CLUSSDR.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La longueur de cette zone est indiquée par MQ_USER_ID_LENGTH. Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD_VERSION_2.

ModeName (MQCHAR8)
Cette zone indique le nom de mode LU 6.2 .

Cette zone est pertinente uniquement si le protocole de transmission (*TransportType*) est MQXPT_LU62et que *ChannelType* n'est pas MQCHT_SVRCONN ou MQCHT_RECEIVER.

Cette zone est toujours vide. Les informations sont contenues dans l'objet côté communications à la place.

La longueur de cette zone est indiquée par MQ_MODE_NAME_LENGTH.

MsgCompListe [16] (MQLONG)
Cette zone indique la liste des techniques de compression de données de message prises en charge par le canal.

La liste contient une ou plusieurs des valeurs suivantes:

MQCOMPRESS_NONE
Aucune compression de données de message n'est effectuée.

MQCOMPRESS_RLE
La compression de données de message est effectuée à l'aide de l'algorithme RLE.

MQCOMPRESS_ZLIBFAST
La compression de données de message est effectuée à l'aide de la technique de compression zlib. Il est préférable d'utiliser une durée de compression rapide.

MQCOMPRESS_ZLIBHIGH
La compression de données de message est effectuée à l'aide de la technique de compression zlib. Il est préférable d'utiliser une compression de haut niveau.

Les valeurs inutilisées dans le tableau sont définies sur MQCOMPRESS_NOT_AVAILABLE.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_8.

MsgExit (MQCHARn)
Cette zone indique le nom de l'exit de message de canal.

Si ce nom n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement après l'extraction d'un message de la file d'attente de transmission (émetteur ou serveur) ou immédiatement avant l'insertion d'un message dans une file d'attente de destination (récepteur ou demandeur).

  L'exit est associé à l'intégralité du message d'application et à l'en-tête de la file d'attente de transmission pour modification.
- Lors de l'initialisation et de l'arrêt du canal.

Cette zone n'est pas pertinente pour les canaux dont le *ChannelType* est MQCHT_SVRCONN ou MQCHT_CLNTCONN; un exit de message n'est jamais appelé pour ces canaux.

Voir «MQCD-Définition de canal», à la page 1045 pour une description du contenu de cette zone dans divers environnements.

La longueur de cette zone est indiquée par MQ_EXIT_NAME_LENGTH.

Remarque : La valeur de cette constante est spécifique à l'environnement.

MsgExitPtr (MQPTR)
Cette zone indique l'adresse de la première zone *MsgExit* .

Si *MsgExitsDefined* est supérieur à zéro, cette adresse est l'adresse de la liste des noms de chaque exit de message de canal dans la chaîne.

Chaque nom se trouve dans une zone de longueur *ExitNameLength*, complétée à droite par des blancs. Il y a des champs *MsgExitsDefined* adjacents les uns aux autres-un pour chaque sortie.

Toutes les modifications apportées à ces noms par un exit sont conservées, bien que l'exit de canal de transmission de messages n'effectue aucune action explicite-il ne modifie pas les exits qui sont appelés.

Si *MsgExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

MsgExitsdéfini (MQLONG)
Cette zone indique le nombre d'exits de message de canal définis dans la chaîne.

Elle est supérieure ou égale à zéro.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

MsgRetry(MQLONG)
Cette zone indique le nombre de fois où MCA tente d'insérer le message, après l'échec de la première tentative.

Cette zone indique le nombre de fois où l'agent MCA tente l'opération d'ouverture ou d'insertion, si le premier MQOPEN ou MQPUT échoue avec le code achèvement MQCC_FAILED.L'effet de cet attribut varie selon que *MsgRetryExit* est vide ou non:

- Si *MsgRetryExit* est vide, l'attribut *MsgRetryCount* contrôle si l'agent MCA tente de nouveau d'effectuer des tentatives. Si la valeur de l'attribut est zéro, aucune nouvelle tentative n'est effectuée. Si la valeur de l'attribut est supérieure à zéro, les nouvelles tentatives sont effectuées à des intervalles définis par l'attribut *MsgRetryInterval* .

  Les nouvelles tentatives sont effectuées uniquement pour les codes anomalie suivants:

  – MQRC_PAGESET_FULL
  – MQRC_PUT_INHIBÉ
  – MQRC_Q_FULL

  Pour les autres codes raison, l'agent MCA passe immédiatement à son traitement d'échec normal, sans relancer le message d'échec.
- Si *MsgRetryExit* n'est pas à blanc, l'attribut *MsgRetryCount* n'affecte pas l'agent MCA ; il s'agit plutôt de l'exit de relance de message qui détermine le nombre de tentatives et les intervalles entre les tentatives ; l'exit est appelé même si l'attribut *MsgRetryCount* est égal à zéro.

  L'attribut *MsgRetryCount* est mis à la disposition de l'exit dans la structure MQCD, mais l'exit qu'il n'est pas nécessaire de prendre en compte-les nouvelles tentatives se poursuivent indéfiniment jusqu'à ce que l'exit renvoie MQXCC_SUPPRESS_FUNCTION dans la zone *ExitResponse* de MQCXP.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_REQUESTER, MQCHT_RECEIVER ou MQCHT_CLUSRCVR.

Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD_VERSION_3.

Exit MsgRetry(MQCHARn)
Cette zone indique le nom de l'exit de relance de message de canal.

L'exit de relance de message est un exit appelé par l'agent MCA lorsque ce dernier reçoit le code achèvement MQCC_FAILED d'un appel MQOPEN ou MQPUT.L'objectif de l'exit est de spécifier un intervalle de temps pendant lequel l'agent MCA attend avant de relancer l'opération MQOPEN ou MQPUT. L'exit peut également être défini pour ne pas relancer l'opération.

L'exit est appelé pour tous les codes raison dont le code achèvement est MQCC_FAILED-les paramètres de l'exit déterminent les codes raison pour lesquels l'agent MCA doit effectuer une nouvelle tentative, le nombre de tentatives et les intervalles de temps.

Lorsque l'opération ne doit plus être tentée, l'agent MCA exécute son traitement d'échec normal ; ce traitement comprend la génération d'un message de rapport d'exception (s'il est spécifié par l'expéditeur) et le placement du message d'origine dans la file d'attente des messages non livrés ou la suppression du message (selon que l'expéditeur a spécifié MQRO_DEAD_LETTER_Q ou MQRO_DISCARD_MSG). Les échecs impliquant la file d'attente de rebut (par exemple, la file d'attente de rebut saturée) n'entraînent pas l'appel de l'exit de relance de message.

Si le nom de l'exit n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement avant d'effectuer l'attente avant de tenter de distribuer à nouveau un message
- Lors de l'initialisation et de l'arrêt du canal

Voir «MQCD-Définition de canal», à la page 1045 pour une description du contenu de cette zone dans divers environnements.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_REQUESTER, MQCHT_RECEIVER ou MQCHT_CLUSRCVR.

La longueur de cette zone est indiquée par MQ_EXIT_NAME_LENGTH.

Remarque : La valeur de cette constante est spécifique à l'environnement.

Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD_VERSION_3.

Intervalle MsgRetry(MQLONG)
Cette zone indique l'intervalle minimal en millisecondes après lequel l'opération d'ouverture ou d'insertion est relancée.

L'effet de cet attribut varie selon que *MsgRetryExit* est vide ou non:

- Si *MsgRetryExit* est vide, l'attribut *MsgRetryInterval* indique la période minimale pendant laquelle l'agent MCA attend avant de relancer un message, si le premier MQOPEN ou MQPUT échoue avec le code achèvement MQCC_FAILED. La valeur zéro signifie que la nouvelle tentative sera effectuée dès que possible après la tentative précédente. Les nouvelles tentatives sont effectuées uniquement si *MsgRetryCount* est supérieur à zéro.

  Cet attribut est également utilisé comme temps d'attente si l'exit de relance de message renvoie une valeur non valide dans la zone *MsgRetryInterval* de MQCXP.
- Si *MsgRetryExit* n'est pas vide, l'attribut *MsgRetryInterval* n'affecte pas l'agent MCA ; c'est plutôt l'exit de relance de message qui détermine la durée d'attente de l'agent MCA. L'attribut *MsgRetryInterval* est mis à la disposition de l'exit dans la structure MQCD, mais l'exit n'est pas requis pour l'honorer.

La valeur est comprise entre 0 et 999 999 999.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_REQUESTER, MQCHT_RECEIVER ou MQCHT_CLUSRCVR.

Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD_VERSION_3.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD_VERSION_4.

MsgRetryUserData (MQCHAR32)
Cette zone indique les données utilisateur de l'exit de relance de message de canal.

Ces données sont transmises à l'exit de relance de message de canal dans la zone *ExitData* du paramètre *ChannelExitParms* (voir MQ_CHANNEL_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_REQUESTER, MQCHT_RECEIVER ou MQCHT_CLUSRCVR.

La longueur de cette zone est donnée par MQ_EXIT_DATA_LENGTH. Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD_VERSION_3.

Cette zone n'est pas pertinente dans WebSphere MQ for IBM i.

Données MsgUser(MQCHAR32)
Cette zone indique les données utilisateur de l'exit de message de canal.

Ces données sont transmises à l'exit de message de canal dans la zone *ExitData* du paramètre *ChannelExitParms* (voir MQ_CHANNEL_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

La longueur de cette zone est donnée par MQ_EXIT_DATA_LENGTH.

Cette zone n'est pas pertinente dans WebSphere MQ for IBM i.

MsgUserDataPtr (MQPTR)
Cette zone indique l'adresse de la première zone *MsgUserData* .

Si *MsgExitsDefined* est supérieur à zéro, cette adresse correspond à l'adresse de la liste des éléments de données utilisateur pour chaque exit de message de canal de la chaîne.

Chaque élément de données utilisateur se trouve dans une zone de longueur *ExitDataLength*, complétée à droite par des blancs. Il y a des champs *MsgExitsDefined* adjacents les uns aux autres-un pour chaque sortie. Si le nombre d'éléments de données utilisateur définis est inférieur au nombre de noms d'exit, les éléments de données utilisateur non définis sont mis à blanc. A l'inverse, si le nombre d'éléments de données utilisateur définis est supérieur au nombre de noms d'exit, les éléments de données utilisateur excédentaires sont ignorés et ne sont pas présentés à l'exit.

Toutes les modifications apportées à ces valeurs par un exit sont conservées. Cela permet à un exit de transmettre des informations à un autre exit. Aucune validation n'est effectuée sur les modifications. Par exemple, des données binaires peuvent être écrites dans ces zones si nécessaire.

Si *MsgExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

NetworkPriority (MQLONG)
Cette zone indique la priorité de la connexion réseau pour le canal.

Lorsque plusieurs chemins d'accès à une destination particulière sont disponibles, le chemin ayant la priorité la plus élevée est choisi. La valeur est comprise entre 0 et 9 ; 0 est la priorité la plus basse.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_5.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD_VERSION_6.

NonPersistentMsgSpeed (MQLONG)
Cette zone indique la vitesse à laquelle les messages non persistants transitent par le canal.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_RECEIVER, MQCHT_REQUESTER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

La valeur est l'une des suivantes :

MQNPMS_NORMAL
Vitesse normale.

Si un canal est défini pour être MQNPMS_NORMAL, les messages non persistants transitent par le canal à une vitesse normale. Cela présente l'avantage que ces messages ne sont pas perdus en cas de défaillance du canal. En outre, les messages persistants et non persistants de la même file d'attente de transmission conservent leur ordre relatif les uns par rapport aux autres.

MQNPMS_FAST
Vitesse rapide.

Si un canal est défini comme MQNPMS_FAST, les messages non persistants transitent par le canal à vitesse rapide. Cela améliore le débit du canal, mais signifie que les messages non persistants sont perdus en cas de défaillance du canal. En outre, il est possible que les messages non persistants passent avant les messages persistants en attente dans la même file d'attente de transmission, c'est-à-dire que l'ordre des messages non persistants n'est pas conservé par rapport aux messages persistants. Toutefois, l'ordre des messages non persistants les uns par rapport aux autres est conservé. De même, l'ordre des messages persistants les uns par rapport aux autres est conservé.

Mot de passe (MQCHAR12)
Cette zone indique le mot de passe utilisé par l'agent MCA lors de la tentative de lancement d'une session SNA sécurisée avec un agent MCA éloigné.

Cette zone ne peut être vide que sur les systèmes UNIX et Windows et n'est pertinente que pour les canaux avec un *ChannelType* de type MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER ou MQCHT_CLNTCONN. Sous z/OS, cette zone n'est pas pertinente.

La longueur de cette zone est indiquée par MQ_PASSWORD_LENGTH. Cependant, seuls les 10 premiers caractères sont utilisés.

Cette zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_2.

PropertyControl (MQLONG)
Cette zone indique ce qu'il advient des propriétés des messages lorsque le message est sur le point d'être envoyé à un gestionnaire de files d'attente V6 ou antérieure (gestionnaire de files d'attente qui ne comprend pas le concept de descripteur de propriété).

La valeur peut être :

COMPATIBILITE_MQPROP_COMPATIBILITÉ
Si le message contient une propriété avec le préfixe **mcd.**, **jms.**, **usr.** ou **mqext.**, toutes les propriétés de message sont distribuées à l'application dans un en-tête MQRFH2 . Sinon, toutes les propriétés du message, à l'exception de celles contenues dans le descripteur de message (ou l'extension), sont supprimées et ne sont plus accessibles à l'application.

Cette valeur est la valeur par défaut ; elle permet aux applications, qui s'attendent à ce que les propriétés liées à JMS se trouvent dans un en-tête MQRFH2 dans les données de message, de continuer à fonctionner sans modification.

MQPROP_NONE
Toutes les propriétés du message, à l'exception de celles du descripteur de message (ou extension), sont supprimées du message avant son envoi au gestionnaire de files d'attente éloignées.

MQPROP_ALL
Toutes les propriétés du message sont incluses dans le message lorsqu'il est envoyé au gestionnaire de files d'attente éloignées. Les propriétés, hormis celles associées au descripteur de message (ou à l'extension), sont placées dans un ou plusieurs en-têtes MQRFH2 dans les données du message.

Cet attribut est applicable aux canaux émetteur, serveur, émetteur de cluster et récepteur de cluster.

«MQIA_ * (Sélecteurs d'attribut d'entier)», à la page 115
«MQPROP_ * (Valeurs de contrôle de propriété de file d'attente et de canal et longueur maximale des propriétés)», à la page 152

PutAuthority (MQLONG)
Cette zone indique si l'ID utilisateur dans les informations contextuelles associées à un message est utilisé pour établir le droit d'insertion du message dans la file d'attente de destination.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_REQUESTER, MQCHT_RECEIVER ou MQCHT_CLUSRCVR. Il peut s'agir de :

MQPA_PAR DEFAUT
L'ID utilisateur par défaut est utilisé.

MQPA_CONTEXT
L'ID utilisateur de contexte est utilisé.

MQPA_ALTERNATE_OR_MCA
L'ID utilisateur de la zone UserIdentifier du descripteur de message est utilisé. Aucun ID utilisateur reçu du réseau n'est utilisé. Cette valeur est prise en charge uniquement sur z/OS.

MQPA_ONLY_MCA
L'ID utilisateur par défaut est utilisé. Aucun ID utilisateur reçu du réseau n'est utilisé. Cette valeur est prise en charge uniquement sur z/OS.

QMgrName (MQCHAR48)
Cette zone indique le nom du gestionnaire de files d'attente auquel un exit peut se connecter.

Pour les canaux avec un *ChannelType* autre que MQCHT_CLNTCONN, cette zone correspond au nom du gestionnaire de files d'attente auquel un exit peut se connecter, qui est toujours non vide sur les systèmes UNIX, Linux et Windows .

La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH.

ReceiveExit (MQCHARn)
Cette zone indique le nom de l'exit de réception de canal.

Si ce nom n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement avant le traitement des données réseau reçues.

  L'exit reçoit la mémoire tampon de transmission complète telle qu'elle a été reçue. Le contenu de la mémoire tampon peut être modifié selon les besoins.
- Lors de l'initialisation et de l'arrêt du canal.

Voir «MQCD-Définition de canal», à la page 1045 pour une description du contenu de cette zone dans divers environnements.

La longueur de cette zone est indiquée par MQ_EXIT_NAME_LENGTH.

Remarque : La valeur de cette constante est spécifique à l'environnement.

ReceiveExitPtr (MQPTR)
Cette zone indique l'adresse de la première zone *ReceiveExit* .

Si *ReceiveExitsDefined* est supérieur à zéro, cette adresse est l'adresse de la liste des noms de chaque exit de réception de canal dans la chaîne.

Chaque nom se trouve dans une zone de longueur *ExitNameLength*, complétée à droite par des blancs. Il y a des champs *ReceiveExitsDefined* adjacents les uns aux autres-un pour chaque sortie.

Toutes les modifications apportées à ces noms par un exit sont conservées, bien que l'exit de canal de transmission de messages n'effectue aucune action explicite-il ne modifie pas les exits qui sont appelés.

Si *ReceiveExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

ReceiveExitsdéfini (MQLONG)
Cette zone indique le nombre d'exits de réception de canal définis dans la chaîne.

Elle est supérieure ou égale à zéro.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

Données ReceiveUser(MQCHAR32)
Ce canal spécifie les données utilisateur d'exit de réception de canal.

Ces données sont transmises à l'exit de réception de canal dans la zone *ExitData* du paramètre *ChannelExitParms* (voir MQ_CHANNEL_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Cela s'applique aux exits sur différentes conversations. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

La longueur de cette zone est donnée par MQ_EXIT_DATA_LENGTH.

Cette zone n'est pas pertinente dans WebSphere MQ for IBM i.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD_VERSION_2.

ReceiveUserDataPtr (MQPTR)
Cette zone indique l'adresse de la première zone *ReceiveUserData* .

Si *ReceiveExitsDefined* est supérieur à zéro, cette adresse correspond à l'adresse de la liste des éléments de données utilisateur pour chaque exit de réception de canal de la chaîne.

Chaque élément de données utilisateur se trouve dans une zone de longueur *ExitDataLength*, complétée à droite par des blancs. Il y a des champs *ReceiveExitsDefined* adjacents les uns aux autres-un pour chaque sortie. Si le nombre d'éléments de données utilisateur définis est inférieur au nombre de noms d'exit, les éléments de données utilisateur non définis sont mis à blanc. A l'inverse, si le nombre d'éléments de données utilisateur définis est supérieur au nombre de noms d'exit, les éléments de données utilisateur excédentaires sont ignorés et ne sont pas présentés à l'exit.

Toutes les modifications apportées à ces valeurs par un exit sont conservées. Cela permet à un exit de transmettre des informations à un autre exit. Aucune validation n'est effectuée sur les modifications. Par exemple, des données binaires peuvent être écrites dans ces zones si nécessaire.

Si *ReceiveExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD_VERSION_5.

RemotePassword (MQCHAR12)
Cette zone indique le mot de passe d'un partenaire.

Cette zone contient des informations valides uniquement si *ChannelType* est MQCHT_CLNTCONN ou MQCHT_SVRCONN.

- Pour un exit de sécurité sur un canal MQCHT_CLNTCONN, ce mot de passe est un mot de passe obtenu de l'environnement. L'exit peut choisir de l'envoyer à l'exit de sécurité sur le serveur.
- Pour un exit de sécurité sur un canal MQCHT_SVRCONN, cette zone peut contenir un mot de passe obtenu à partir de l'environnement sur le client, s'il n'y a pas d'exit de sécurité client. L'exit peut utiliser ce mot de passe pour valider l'identificateur utilisateur dans *RemoteUserIdentifier*.

  S'il existe un exit de sécurité sur le client, ces informations peuvent être obtenues dans un flux de sécurité à partir du client.

La longueur de cette zone est indiquée par MQ_PASSWORD_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_2.

ID RemoteSecurity(MQBYTE40)
Cette zone indique l'identificateur de sécurité de l'utilisateur distant.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_CLNTCONN ou MQCHT_SVRCONN.

La valeur spéciale suivante indique qu'il n'y a pas d'identificateur de sécurité:

MQSID_AUCUN
Aucun identificateur de sécurité n'a été indiqué.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQSID_NONE_ARRAY est également définie ; cette constante a la même valeur que MQSID_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée de l'exit. La longueur de cette zone est indiquée par MQ_SECURITY_ID_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_6.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCD_VERSION_7.

Identificateur RemoteUser(MQCHAR12)
Cette zone indique les 12 premiers octets d'un ID utilisateur d'un partenaire.

Deux zones contiennent l'identificateur d'utilisateur distant:

- *RemoteUserIdentifier* contient les 12 premiers octets de l'identificateur d'utilisateur distant et est rempli avec des blancs si l'identificateur est inférieur à 12 octets. *RemoteUserIdentifier* peut être vide.
- *LongRemoteUserIdPtr* pointe vers l'ID utilisateur distant complet, qui peut dépasser 12 octets. Sa longueur est donnée par *LongRemoteUserIdLength*. L'identificateur complet ne contient pas de blancs de fin et n'est pas terminé par une valeur NULL. Si l'identificateur est vide, *LongRemoteUserIdLength* est égal à zéro et la valeur de *LongRemoteUserIdPtr* n'est pas définie.

  LongRemoteUserIdPtr n'est pas présent si *Version* est inférieur à MQCD_VERSION_6.

L'identificateur d'utilisateur distant est pertinent uniquement pour les canaux dont le *ChannelType* est MQCHT_CLNTCONN ou MQCHT_SVRCONN.

- Pour un exit de sécurité sur un canal MQCHT_CLNTCONN, cette valeur est un identificateur utilisateur obtenu de l'environnement. L'exit peut choisir de l'envoyer à l'exit de sécurité sur le serveur.
- Pour un exit de sécurité sur un canal MQCHT_SVRCONN, cette zone peut contenir un identificateur utilisateur qui a été obtenu à partir de l'environnement sur le client, s'il n'y a pas d'exit de sécurité client. L'exit peut valider cet ID utilisateur (éventuellement avec le mot de passe dans *RemotePassword*) et mettre à jour la valeur dans *MCAUserIdentifier*.

  S'il existe un exit de sécurité sur le client, ces informations peuvent être obtenues dans un flux de sécurité à partir du client.

La longueur de cette zone est indiquée par MQ_USER_ID_LENGTH. Cette zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_2.

SecurityExit (MQCHARn)
Cette zone indique le nom de l'exit de sécurité du canal.

Si ce nom n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement après l'établissement d'un canal.

  Avant que les messages soient transférés, l'exit a la possibilité d'émettre un flux de messages de sécurisation afin de valider des droits de connexion.
- A la réception d'une réponse à un flux de messages de sécurité.

  Tous les flux de messages de sécurité reçus du processeur distant sur la machine distante sont transmis à l'exit.
- Lors de l'initialisation et de l'arrêt du canal.

Voir«MQCD-Définition de canal», à la page 1045 pour une description du contenu de cette zone dans divers environnements.

La longueur de cette zone est indiquée par MQ_EXIT_NAME_LENGTH.

Remarque : La valeur de cette constante est spécifique à l'environnement.

Données SecurityUser(MQCHAR32)
Ce canal indique les données utilisateur de l'exit de sécurité du canal.

Ces données sont transmises à l'exit de sécurité de canal dans la zone *ExitData* du paramètre *ChannelExitParms* (voir MQ_CHANNEL_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Cela s'applique aux exits sur différentes conversations. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

La longueur de cette zone est donnée par MQ_EXIT_DATA_LENGTH.

Cette zone n'est pas pertinente dans WebSphere MQ for IBM i.

SendExit (MQCHARn)
Cette zone indique le nom de l'exit d'émission de canal.

Si ce nom n'est pas vide, l'exit est appelé aux moments suivants:

- Immédiatement avant l'envoi des données sur le réseau.

  L'exit reçoit la mémoire tampon de transmission complète avant qu'elle ne soit transmise. Le contenu de la mémoire tampon peut être modifié selon les besoins.
- Lors de l'initialisation et de l'arrêt du canal.

Voir «MQCD-Définition de canal», à la page 1045 pour une description du contenu de cette zone dans divers environnements.

La longueur de cette zone est indiquée par MQ_EXIT_NAME_LENGTH.

Remarque : La valeur de cette constante est spécifique à l'environnement.

SendExitPtr (MQPTR)
Cette zone indique l'adresse de la première zone *SendExit* .

Si *SendExitsDefined* est supérieur à zéro, cette adresse est l'adresse de la liste des noms de chaque exit d'émission de canal de la chaîne.

Chaque nom se trouve dans une zone de longueur *ExitNameLength*, complétée à droite par des blancs. Il y a des champs *SendExitsDefined* adjacents les uns aux autres-un pour chaque sortie.

Toutes les modifications apportées à ces noms par un exit sont conservées, bien que l'exit d'envoi de message n'effectue aucune action explicite-il ne modifie pas les exits qui sont appelés.

Si *SendExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

SendExitsdéfini (MQLONG)
Cette zone indique le nombre d'exits d'émission de canal définis dans la chaîne.

Elle est supérieure ou égale à zéro.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

Données SendUser(MQCHAR32)
Cette zone indique les données utilisateur de l'exit d'émission de canal.

Ces données sont transmises à l'exit d'émission de canal dans la zone *ExitData* du paramètre *ChannelExitParms* (voir MQ_CHANNEL_EXIT).

Cette zone contient initialement les données qui ont été définies dans la définition de canal. Toutefois, pendant la durée de vie de cette instance MCA, toutes les modifications apportées au contenu de cette zone par un exit de n'importe quel type sont conservées par l'agent MCA et rendues visibles par les appels ultérieurs des exits (quel que soit le type) pour cette instance MCA. Cela s'applique aux exits sur différentes conversations. Ces modifications n'affectent pas la définition de canal utilisée par d'autres instances MCA. Tous les caractères (y compris les données binaires) peuvent être utilisés.

La longueur de cette zone est donnée par MQ_EXIT_DATA_LENGTH.

Cette zone n'est pas pertinente dans WebSphere MQ for IBM i.

SendUserDataPtr (MQPTR)
Cette zone indique l'adresse de la zone *SendUserData* .

Si *SendExitsDefined* est supérieur à zéro, cette adresse correspond à l'adresse de la liste des éléments de données utilisateur pour chaque exit de message de canal de la chaîne.

Chaque élément de données utilisateur se trouve dans une zone de longueur *ExitDataLength*, complétée à droite par des blancs. Il y a des champs *MsgExitsDefined* adjacents les uns aux autres-un pour chaque sortie. Si le nombre d'éléments de données utilisateur définis est inférieur au nombre de noms d'exit, les éléments de données utilisateur non définis sont mis à blanc. A l'inverse, si le nombre d'éléments de données utilisateur définis est supérieur au nombre de noms d'exit, les éléments de données utilisateur excédentaires sont ignorés et ne sont pas présentés à l'exit.

Toutes les modifications apportées à ces valeurs par un exit sont conservées. Cela permet à un exit de transmettre des informations à un autre exit. Aucune validation n'est effectuée sur les modifications. Par exemple, des données binaires peuvent être écrites dans ces zones si nécessaire.

Si *SendExitsDefined* est égal à zéro, cette zone est le pointeur null.

Sur les plateformes où le langage de programmation ne prend pas en charge le type de données de pointeur, cette zone est déclarée comme une chaîne d'octets de la longueur appropriée.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

SeqNumberRetour à la ligne (MQLONG)
Cette zone indique le numéro de séquence de message le plus élevé autorisé.

Lorsque cette valeur est atteinte, le retour à la ligne des numéros de séquence est redémarré à 1.

Cette valeur n'est pas négociable et doit correspondre aux définitions de canal local et éloigné.

Cette zone n'est pas pertinente pour les canaux dont le *ChannelType* est MQCHT_SVRCONN ou MQCHT_CLNTCONN.

SharingConversations (MQLONG)
Cette zone indique le nombre maximal de conversations pouvant partager une instance de canal associée à ce canal.

Cette zone est utilisée sur les canaux de connexion client et de connexion serveur.

La valeur 0 signifie que le canal fonctionne comme dans les versions antérieures à WebSphere MQ Version 7.0 en ce qui concerne les attributs suivants:

- Partage de conversation
- Lecture anticipée
- `STOP CHANNEL(<channelname>) MODE(QUIESCE)`
- Intervalle des pulsations par lots
- Consommation asynchrone du client

La valeur 1 est la valeur minimale pour le comportement de WebSphere MQ V7.0 . Bien qu'une seule conversation soit autorisée sur l'instance de canal, la lecture anticipée, la consommation asynchrone et

le comportement de la version 7 du signal de présence CLNTCONN-SVRCONN et l'arrêt du canal au repos sont disponibles.

Il s'agit d'une zone d'entrée de l'exit. Elle n'est pas présente si *Version* est inférieur à MQCD_VERSION_9.

La valeur par défaut de cette zone est 10.

Remarque : Les limites *MaxInstances* et *MaxInstancesPerClient* appliquées à un canal limitent le nombre d'instances de canal, et non le nombre de conversations pouvant partager ces instances.

Nom ShortConnection(MQCHAR20)
Cette zone indique les 20 premiers octets d'un nom de connexion.

Si la zone *Version* est MQCD_VERSION_1, *ShortConnectionName* contient le nom de connexion complet.

Si la zone *Version* est MQCD_VERSION_2 ou supérieur, *ShortConnectionName* contient les 20 premiers caractères du nom de connexion. Le nom de connexion complet est donné par la zone *ConnectionName* ; *ShortConnectionName* et les 20 premiers caractères de *ConnectionName* sont identiques.

Pour plus de détails sur le contenu de cette zone, voir *ConnectionName* .

Remarque : Le nom de cette zone a été modifié pour MQCD_VERSION_2 et les versions ultérieures de MQCD ; la zone était auparavant appelée *ConnectionName*.

La longueur de cette zone est indiquée par MQ_SHORT_CONN_NAME_LENGTH.

Nombre de ShortRetry(MQLONG)
Cette zone indique le nombre maximal de tentatives de connexion à une machine distante.

Cette zone indique le nombre maximal de tentatives de connexion à la machine distante, à des intervalles définis par *ShortRetryInterval*, avant que les valeurs (normalement plus longues) *LongRetryCount* et *LongRetryInterval* ne soient utilisées.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

Intervalle ShortRetry(MQLONG)
Cette zone indique le nombre maximal de secondes à attendre avant de tenter à nouveau la connexion à la machine distante.

L'intervalle entre les nouvelles tentatives peut être étendu si le canal doit attendre pour devenir actif.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR ou MQCHT_CLUSRCVR.

SSLCipherSpec (MQCHAR32)
Cette zone indique la spécification de chiffrement utilisée lors de l'utilisation de SSL.

Si SSLCipherSpec est vide, le canal n'utilise pas SSL. S'il n'est pas vide, cette zone contient une chaîne spécifiant le CipherSpec utilisé.

Ce paramètre est valide pour tous les types de canal. Il est pris en charge sous AIX, HP-UX, Linux, IBM i, Solaris, Windowset z/OS. Elle est valide uniquement pour les types de canal d'un type de transport (TRPTYPE) TCP.

Il s'agit d'une zone d'entrée de l'exit. La longueur de cette zone est indiquée par MQ_SSL_CIPHER_SPEC_LENGTH. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_7.

SSLClientAuth (MQLONG)
Cette zone indique si l'authentification du client SSL est requise.

Cette zone concerne uniquement les définitions de canal SVRCONN.

Ses valeurs sont l'une des suivantes :

MQSCA_REQUIS
Authentification du client requise.

MQSCA_XX_ENCODE_CASE_ONE facultatif
Authentification client facultative.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_7.

SSLPeerNameLongueur (MQLONG)
Cette zone indique la longueur en octets du nom d'homologue SSL désigné par *SSLPeerNamePtr*.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_7.

SSLPeerNamePtr (MQPTR)
Cette zone indique l'adresse du nom d'homologue SSL.

Lorsqu'un certificat est reçu lors d'un établissement de liaison SSL réussi, le nom distinctif du sujet du certificat est copié dans la zone MQCD accessible par le Ptr SSLPeerNameà la fin du canal qui reçoit le certificat. Elle remplace la valeur SSLPeerName du canal si cette valeur est présente dans la définition de canal de l'utilisateur local. Si un exit de sécurité est spécifié à cette extrémité du canal, il reçoit le nom distinctif du certificat homologue dans le MQCD.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_7.

Remarque : Les applications d'exit de sécurité construites avant l'édition de WebSphere MQ v7.1 peuvent nécessiter une mise à jour. Pour plus d'informations, voir Programmes d'exit de sécurité de canal .

StrucLength (MQLONG)
Cette zone indique la longueur en octets de la structure MQCD.

La longueur n'inclut aucune des chaînes adressées par les zones de pointeur contenues dans la structure. La valeur est l'une des suivantes :

MQCD_LENGTH_4
Longueur de la structure de définition de canal version-4 .

MQCD_LENGTH_5
Longueur de la structure de définition de canal version-5 .

MQCD_LENGTH_6
Longueur de la structure de définition de canal version-6 .

MQCD_LENGTH_7
Longueur de la structure de définition de canal version-7 .

MQCD_LENGTH_8
Longueur de la structure de définition de canal version-8 .

MQCD_LENGTH_9
Longueur de la structure de définition de canal version-9 .

La constante suivante indique la longueur de la version en cours:

MQCD_LONGUEUR_EN_COURS
Longueur de la version en cours de la structure de définition de canal.

Remarque : Ces constantes ont des valeurs spécifiques à l'environnement.

La zone n'est pas présente si *Version* est inférieur à MQCD_VERSION_4.

TpName (MQCHAR64)
Cette zone indique le nom du programme de transaction LU 6.2 .

Cette zone est pertinente uniquement si le protocole de transmission (*TransportType*) est MQXPT_LU62et que *ChannelType* n'est pas MQCHT_SVRCONN ou MQCHT_RECEIVER.

Cette zone est toujours à blanc sur les plateformes sur lesquelles les informations sont contenues dans l'objet côté communications.

La longueur de cette zone est indiquée par MQ_TP_NAME_LENGTH.

TransportType (MQLONG)
Cette zone indique le protocole de transmission à utiliser.

La valeur n'est pas vérifiée si le canal a été lancé à partir de l'autre extrémité.

Ses valeurs sont l'une des suivantes :

MQXPT_LU62
Protocole de transport LU 6.2 .

MQXPT_TCP
Protocole de transport TCP/IP.

MQXPT_NETBIOS
Protocole de transport NetBIOS .

Cette valeur est prise en charge dans les environnements suivants: Windows.

MQXPT_SPX
Protocole de transport SPX.

Cette valeur est prise en charge dans les environnements suivants: Windows, plus les clients WebSphere MQ connectés à ces systèmes.

UseDLQ (MQLONG)
Cette zone indique si la file d'attente de rebut (ou la file d'attente de messages non livrés) est utilisée lorsque les messages ne peuvent pas être distribués par les canaux.

Il peut contenir l'une des valeurs suivantes:

MQUSEDLQ_NO
Les messages qui ne peuvent pas être distribués par un canal sont traités comme un échec. Le canal supprime le message ou se termine conformément au paramètre NPMSPEED.

MQUSEDLQ_OUI
Lorsque l'attribut de gestionnaire de files d'attente DEADQ fournit le nom d'une file d'attente de rebut, il est utilisé, sinon le comportement est NO. YES est la valeur par défaut.

UserIdentifier (MQCHAR12)
Cette zone indique l'ID utilisateur utilisé par l'agent MCA lors de la tentative d'ouverture d'une session SNA sécurisée avec un agent MCA éloigné.

Cette zone ne peut être vide que sur les systèmes UNIX et Windows , et n'est pertinente que pour les canaux dont le *ChannelType* est MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER ou MQCHT_CLNTCONN. Sous z/OS, cette zone n'est pas pertinente.

La longueur de cette zone est indiquée par MQ_USER_ID_LENGTH. Cependant, seuls les 10 premiers caractères sont utilisés.

Cette zone n'est pas présente lorsque *Version* est inférieur à MQCD_VERSION_2.

Version (MQLONG)
La zone `Version` indique le numéro de version le plus élevé que vous pouvez définir pour la structure.

La valeur dépend de l'environnement:

MQCD_VERSION_1
Structure de définition de canal version 1.

MQCD_VERSION_2
Structure de définition de canal version 2.

La version 2 n'est utilisée par aucun produit IBM WebSphere MQ en cours.

MQCD_VERSION_3
Structure de définition de canal version 3.

La version 3 est la plus élevée que vous pouvez définir pour la zone sur MQSeries version 2 dans les environnements suivants: HP Integrity NonStop Serveret les systèmes UNIX and Linux non répertoriés ailleurs.

MQCD_VERSION_4
Structure de définition de canal version 4.

La version 4 n'est utilisée par aucun produit IBM WebSphere MQ en cours.

MQCD_VERSION_5
Structure de définition de canal version 5.

La version 5 est la plus élevée que vous pouvez définir pour la zone sur MQSeries pour OS/390 version 5 édition 2.

MQCD_VERSION_6
Structure de définition de canal version 6.

La version 6 n'est pas la version actuelle de la structure MQCD d'un produit IBM WebSphere MQ existant. Toutefois, une structure MQCD version 6 peut être transmise à MQCONNX à l'aide des zones ClientConnOffset ou ClientConnPtr de la structure MQCNO .

Sur les plateformes réparties, la version 6 est la version par défaut dans les initialiseurs MQCD_DEFAULT et MQCD_CLIENT_CONN_DEFAULT . Si vous souhaitez référencer les zones MQCD_VERSION_7, MQCD_VERSION_8ou MQCD_VERSION_9 du MQCD, initialisez explicitement la zone MQCD **Version** dans MQCD_VERSION_7, MQCD_VERSION_8ou MQCD_VERSION_9 , selon le cas.

Sous z/OS, MQCD_VERSION_7 est la valeur par défaut.

MQCD_VERSION_7
Structure de définition de canal version 7.

La version 7 est la valeur la plus élevée que vous pouvez définir pour la zone sur IBM WebSphere MQ Version 5.3 dans les environnements suivants: AIX, HP-UX, Solaris, Windows, et sur IBM WebSphere MQ for z/OS Version 5.3 et Version 5.3.1. MQCD_VERSION_7 est la valeur par défaut pour les versions de IBM WebSphere MQ for z/OS.

MQCD_VERSION_8
Structure de définition de canal version 8.

La version 8 est la plus élevée que vous pouvez définir pour la zone sur IBM WebSphere MQ Version 6.0 sur toutes les plateformes.

MQCD_VERSION_9
Structure de définition de canal version 9.

La version 9 est la valeur la plus élevée que vous pouvez définir pour la zone sur IBM WebSphere MQ Version 7.0 et IBM WebSphere MQ Version 7.0.1 sur toutes les plateformes.

MQCD_VERSION_10
Structure de définition de canal version 10.

La version 10 est la plus élevée que vous pouvez définir pour la zone sur IBM WebSphere MQ Version 7.1 et IBM WebSphere MQ Version 7.5 sur toutes les plateformes.

Les zones qui existent uniquement dans les versions les plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

MQCD_CURRENT_VERSION
La valeur définie dans MQCD_CURRENT_VERSION est la version actuelle de la structure de définition de canal utilisée.

La valeur de MQCD_CURRENT_VERSION dépend de l'environnement. Il contient la valeur la plus élevée prise en charge par la plateforme.

MQCD_CURRENT_VERSION n'est pas utilisé pour initialiser les structures par défaut fournies dans les fichiers d'en-tête, de copie et d'inclusion fournis pour les différents langages de programmation. L'initialisation par défaut de Version dépend de la plateforme et de l'édition.

Pour IBM WebSphere MQ Version 7.0 et les versions ultérieures, les déclarations MQCD dans les fichiers d'en-tête, de copie et d'inclusion sont initialisées dans MQCD_VERSION_6. Pour utiliser des zones MQCD supplémentaires, les applications doivent définir le numéro de version sur MQCD_CURRENT_VERSION. Si vous écrivez une application portable entre plusieurs environnements, vous devez choisir une version prise en charge dans tous les environnements.

Conseil : Lorsqu'une nouvelle version de la structure MQCD est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit vérifier le numéro de version. Elle doit être supérieure ou égale à la version la plus basse qui contient les zones que l'exit doit utiliser.

XmitQName (MQCHAR48)
Cette zone indique le nom de la file d'attente de transmission à partir de laquelle les messages sont extraits.

Cette zone est pertinente uniquement pour les canaux dont le *ChannelType* est MQCHT_SENDER ou MQCHT_SERVER.

La longueur de cette zone est indiquée par MQ_Q_NAME_LENGTH.

Déclaration C
Cette déclaration est la déclaration C pour la structure MQCD.

```
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 */
 };
```

Déclaration COBOL

Cette déclaration est la déclaration COBOL pour la structure MQCD.

```
** 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 **
```

Déclaration RPG (ILE)

Cette déclaration est la déclaration RPG pour la structure MQCD.

```
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 **
```

Déclaration assembleur System/390

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQCD.

```
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)
```

Déclaration Visual Basic

Cette déclaration est la déclaration Visual Basic de la structure MQCD.

Dans Visual Basic, la structure MQCD peut être utilisée avec la structure MQCNO sur l'appel MQCONNX.

```
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
```

Modification des zones MQCD dans un exit de canal

Un exit de canal peut modifier les zones de la base de données MQCD. Toutefois, ces modifications ne sont généralement pas prises en compte, sauf dans les circonstances indiquées.

Si un programme d'exit de canal modifie une zone dans la structure de données MQCD, la nouvelle valeur est généralement ignorée par le processus de canal WebSphere MQ . Toutefois, la nouvelle valeur reste dans le MQCD et est transmise aux exits restants d'une chaîne d'exit et à toute conversation partageant l'instance de canal.

Si SharingConversations est défini sur FALSE dans la structure MQCXP, des modifications peuvent être apportées à certaines zones en fonction du type de programme d'exit, du type de canal et du code anomalie de l'exit. Le tableau suivant présente les zones qui peuvent être modifiées et affecter le comportement du canal, et dans quelles circonstances. Si un programme d'exit modifie l'une de ces zones dans d'autres circonstances ou dans une zone non répertoriée, la nouvelle valeur est ignorée par le processus de canal. La nouvelle valeur reste dans le MQCD et est transmise à tous les exits restants d'une chaîne d'exit et à toute conversation partageant l'instance de canal.

Tout type de programme d'exit lorsqu'il est appelé pour l'initialisation (MQXR_INIT) peut modifier la zone ChannelName de tout type de canal, tant que MQCXP SharingConverstions est défini sur FALSE. Seul un exit de sécurité peut modifier la zone MCAUserIdentifier , quelle que soit la valeur de MQCXP SharingConverstions.

Zone	**Code raison de sortie**	**Type de sortie**	**Type de canal**
ChannelName	MQXR_INIT	Tous	Tous
TransportType	MQXR_INIT	Tous	Tous
XmitQName	MQXR_INIT	Tous	SDR, RCVR
ModeName	MQXR_INIT	Tous	Tous
TpName	MQXR_INIT	Tous	Tous
BatchSize	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
DiscInterval	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR

Zone	Code raison de sortie	Type de sortie	Type de canal
ShortRetry	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Intervalle ShortRetry	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Nombre de LongRetry	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Intervalle LongRetry	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
SeqNumberRetour à la ligne	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Longueur maximale des messages	MQXR_INIT	Tous	Tous
PutAuthority	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
DataConversion	MQXR_INIT	Tous	Tous
MCAUserIdentifier	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	Sécurité	RCVR, RQSTR, SVRCONN, CLUSRCVR
ConnectionName	MQXR_INIT	Tous	SDR, SVR, RQSTR, CLNTCONN, CLUSSDR, CLUSRCVR
MsgRetryUserData	MQXR_INIT	Tous	RCVR, RQSTR, CLUSRCVR
MsgRetry	MQXR_INIT	Tous	RCVR, RQSTR, CLUSRCVR

Zone	Code raison de sortie	Type de sortie	Type de canal
MsgRetryIntervalle	MQXR_INIT	Tous	RCVR, RQSTR, CLUSRCVR
HeartbeatInterval	MQXR_INIT	Tous	Tous
BatchInterval	MQXR_INIT	Tous	SDR, SVR, CLUSSDR, CLUSRCVR
NonPersistentMsgSpeed	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
MCASecurityId	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	Sécurité	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSSDR, CLUSRCVR
SSLCipherSpec	MQXR_INIT	Tous	Tous
SSLPeerNamePtr	MQXR_INIT	Tous	Tous
SSLPeerNameLongueur	MQXR_INIT	Tous	Tous
SSLClientAuth	MQXR_INIT	Tous	RVR, RCVR, RQSTR, SVRCONN, CLUSRCVR
KeepAliveInterval	MQXR_INIT	Tous	Tous
LocalAddress	MQXR_INIT	Tous	SDR, SVR, RQSTR, CLNTCONN, CLUSSDR, CLUSRCVR
BatchHeartbeat	MQXR_INIT	Tous	SDR, SVR, CLUSSDR, CLUSRCVR
Liste HdrComp	MQXR_INIT	Tous	Tous
Liste MsgComp	MQXR_INIT	Tous	Tous
ChannelMonitoring	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSSDR, CLUSRCVR

Zone	Code raison de sortie	Type de sortie	Type de canal
ChannelStatistics	MQXR_INIT	Tous	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
SharingConversations	MQXR_INIT	Tous	SVRCONN, CLNTCONN
PropertyControl	MQXR_INIT	Tous	SDR, SVR, CLUSSDR, CLUSRCVR

MQCXP-Paramètre d'exit de canal

La structure MQCXP est transmise à chaque type d'exit appelé par un agent MCA (Message Channel Agent), un canal de connexion client ou un canal de connexion serveur.

Voir MQ_CHANNEL_EXIT.

Les zones décrites comme "entrée de l'exit" dans les descriptions qui suivent sont ignorées par le canal lorsque l'exit renvoie le contrôle au canal. Les zones d'entrée que l'exit modifie dans le bloc de paramètres d'exit de canal ne seront pas conservées pour son prochain appel. Les modifications apportées aux zones d'entrée/sortie (par exemple, la zone *ExitUserArea*) sont conservées pour les appels de cette instance de l'exit uniquement. Ces modifications ne peuvent pas être utilisées pour transmettre des données entre des exits différents définis sur le même canal ou entre un même exit défini sur des canaux différents.

Référence associée

«Zones», à la page 1086
Cette rubrique répertorie toutes les zones de la structure MQCXP et décrit chaque zone.

«Déclaration C», à la page 1097
Cette déclaration est la déclaration C pour la structure MQCXP.

«Déclaration COBOL», à la page 1098
Cette déclaration est la déclaration COBOL pour la structure MQCXP.

«Déclaration RPG (ILE)», à la page 1099
Cette déclaration est la déclaration RPG pour la structure MQCXP.

«Déclaration assembleur System/390», à la page 1100
Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQCXP.

Zones

Cette rubrique répertorie toutes les zones de la structure MQCXP et décrit chaque zone.

StrucId (MQCHAR4)
Cette zone indique l'identificateur de structure.

La valeur doit être:

MQCXP_STRUC_ID
Identificateur de la structure des paramètres d'exit de canal.

Pour le langage de programmation C, la constante MQCXP_STRUC_ID_ARRAY est également définie ; cette constante a la même valeur que MQCXP_STRUC_ID, mais est un tableau de caractères au lieu d'une chaîne.

Il s'agit d'une zone d'entrée de l'exit.

Version (MQLONG)
Cette zone indique le numéro de version de la structure.

La valeur dépend de l'environnement:

MQCXP_VERSION_1
Structure des paramètres d'exit de canal Version-1 .

MQCXP_VERSION_2
Structure des paramètres d'exit de canal Version-2 .

La zone contient cette valeur dans les environnements suivants: HP Integrity NonStop Server.

MQCXP_VERSION_3
Structure des paramètres d'exit de canal Version-3 .

La zone contient cette valeur dans les environnements suivants: systèmes UNIX non répertoriés ailleurs.

MQCXP_VERSION_4
Structure des paramètres d'exit de canal Version-4 .

MQCXP_VERSION_5
Structure des paramètres d'exit de canal Version-5 .

MQCXP_VERSION_6
Structure des paramètres d'exit de canal Version-6 .

MQCXP_VERSION_8
Structure des paramètres d'exit de canal Version-8 .

Cette zone contient cette valeur dans les environnements suivants: z/OS, AIX, HP-UX, Linux, IBM i, Solaris, Windows.

Les zones qui existent uniquement dans les versions plus récentes de la structure sont identifiées comme telles dans les descriptions des zones. La constante suivante indique le numéro de version de la version en cours:

MQCXP_CURRENT_VERSION
Version actuelle de la structure des paramètres d'exit de canal.

La valeur dépend de l'environnement.

Remarque : Lorsqu'une nouvelle version de la structure MQCXP est introduite, la présentation de la partie existante n'est pas modifiée. L'exit doit donc vérifier que le numéro de version est égal ou supérieur à la version la plus basse qui contient les zones que l'exit doit utiliser.

Il s'agit d'une zone d'entrée de l'exit.

ExitId (MQLONG)
Cette zone indique le type d'exit appelé et est définie à l'entrée de la routine d'exit.

Valeurs possibles :

MQXT_CHANNEL_SEC_EXIT
Exit de sécurité de canal.

MQXT_CHANNEL_MSG_EXIT
Exit de message de canal.

EXIT MQXT_CHANNEL_SEND_EXIT
Exit d'émission de canal.

MQXT_CHANNEL_RCV_EXIT
Exit de réception de canal.

MQXT_CHANNEL_MSG_RETRY_EXIT
Exit de relance de message de canal.

MQXT_CHANNEL_AUTO_DEF_EXIT
Exit de définition automatique de canal.

Sous z/OS, ce type d'exit est pris en charge uniquement pour les canaux de type MQCHT_CLUSSDR et MQCHT_CLUSRCVR.

Il s'agit d'une zone d'entrée de l'exit.

ExitReason (MQLONG)
Cette zone indique la raison pour laquelle l'exit est appelé et est définie à l'entrée de la routine d'exit.

Il n'est pas utilisé par l'exit de définition automatique. Valeurs possibles :

MQXR_INIT
Initialisation de l'exit.

Cette valeur indique que l'exit est appelé pour la première fois. Il permet à l'exit d'acquérir et d'initialiser les ressources dont il a besoin (par exemple, la mémoire).

MQXR_TERM
Arrêt de l'exit.

Cette valeur indique que l'exit est sur le point d'être arrêté. L'exit doit libérer toutes les ressources qu'il a acquises depuis son initialisation (par exemple: mémoire).

MQXR_MSG
Traitement d'un message.

Cette valeur indique que l'exit est appelé pour traiter un message. Cette valeur s'applique uniquement aux exits de message de canal.

MQXR_XMIT
Traiter une transmission.

Cette valeur s'applique uniquement aux exits d'émission et de réception de canal.

MQXR_SEC_MSG
Message de sécurité reçu.

Cette valeur s'applique uniquement aux exits de sécurité de canal.

MQXR_INIT_SEC
Initiez l'échange de sécurité.

Cette valeur s'applique uniquement aux exits de sécurité de canal.

L'exit de sécurité du récepteur est toujours appelé avec cette raison immédiatement après avoir été appelé avec MQXR_INIT, afin de lui donner la possibilité de lancer un échange de sécurité. S'il refuse l'opportunité (en renvoyant MQXCC_OK à la place de MQXCC_SEND_SEC_MSG ou MQXCC_SEND_AND_REQUEST_SEC_MSG), l'exit de sécurité de l'expéditeur est appelé avec MQXR_INIT_SEC.

Si l'exit de sécurité du récepteur lance un échange de sécurité (en renvoyant MQXCC_SEND_SEC_MSG ou MQXCC_SEND_AND_REQUEST_SEC_MSG), l'exit de sécurité de l'expéditeur n'est jamais appelé avec MQXR_INIT_SEC; il est appelé avec MQXR_SEC_MSG pour traiter le message du récepteur. (Dans les deux cas, il est d'abord appelé avec MQXR_INIT.)

Sauf si l'un des exits de sécurité demande l'arrêt du canal (en définissant *ExitResponse* sur MQXCC_SUPPRESS_FUNCTION ou MQXCC_CLOSE_CHANNEL), l'échange de sécurité doit se terminer du côté qui a lancé l'échange. Par conséquent, si un exit de sécurité est appelé avec MQXR_INIT_SEC et qu'il lance un échange, la prochaine fois que l'exit est appelé, il sera appelé avec MQXR_SEC_MSG. Cela se produit, qu'il y ait un message de sécurité à traiter par l'exit ou non. Un message de sécurité s'affiche si le partenaire renvoie MQXCC_SEND_SEC_MSG ou MQXCC_SEND_AND_REQUEST_SEC_MSG, mais pas si le partenaire renvoie MQXCC_OK ou s'il n'y a pas d'exit de sécurité au niveau du partenaire. S'il n'y a pas de message de sécurité à traiter, l'exit de sécurité à l'extrémité initiatrice est à nouveau appelé avec un *DataLength* égal à zéro.

MQXR_RETRY
Relancez un message.

Cette valeur s'applique uniquement aux exits de relance de message.

MQXR_AUTO_CLUSSDR
Définition automatique d'un canal émetteur de cluster.

Cette valeur se produit uniquement pour les exits de définition automatique de canal.

MQXR_AUTO_RECEIVER
Définition automatique d'un canal récepteur.

Cette valeur se produit uniquement pour les exits de définition automatique de canal.

MQXR_AUTO_SVRCONN
Définition automatique d'un canal de connexion serveur.

Cette valeur se produit uniquement pour les exits de définition automatique de canal.

MQXR_AUTO_CLUSRCVR
Définition automatique d'un canal récepteur de cluster.

Cette valeur se produit uniquement pour les exits de définition automatique de canal.

MQXR_SEC_PARMS
Paramètres de sécurité

Cette valeur s'applique uniquement aux exits de sécurité et indique qu'une structure MQCSP est transmise à l'exit. Pour plus d'informations, voir «MQCSP-Paramètres de sécurité», à la page 314

Remarque :

1. Si plusieurs exits sont définis pour un canal, ils sont chacun appelés avec MQXR_INIT lors de l'initialisation de l'agent MCA. En outre, ils sont chacun appelés avec MQXR_TERM lorsque l'agent MCA est arrêté.
2. Pour l'exit de définition automatique de canal, *ExitReason* n'est pas défini si *Version* est inférieur à MQCXP_VERSION_4. La valeur MQXR_AUTO_SVRCONN est implicite dans ce cas.

Il s'agit d'une zone d'entrée de l'exit.

ExitResponse (MQLONG)
Cette zone indique la réponse de l'exit.

Cette zone est définie par l'exit pour communiquer avec l'agent MCA.Il doit s'agir de l'une des valeurs suivantes:

MQXCC_OK
La sortie a abouti.

- Pour l'exit de sécurité de canal, cette valeur indique que le transfert de message peut maintenant se poursuivre normalement.
- Pour l'exit de relance de message de canal, cette valeur indique que l'agent MCA doit attendre l'intervalle de temps renvoyé par l'exit dans la zone *MsgRetryInterval* de MQCXP, puis relancer le message.

La zone *ExitResponse2* peut contenir des informations supplémentaires.

MQXCC_SUPPRESS_FUNCTION
Supprimer la fonction.

- Pour l'exit de sécurité de canal, cette valeur indique que le canal doit être arrêté.
- Pour l'exit de message de canal, cette valeur indique que le message ne doit pas continuer vers sa destination. A la place, l'agent MCA génère un message de rapport d'exception (s'il a été demandé par l'expéditeur du message d'origine) et place le message contenu dans la mémoire tampon d'origine dans la file d'attente de rebut (si l'expéditeur a spécifié MQRO_DEAD_LETTER_Q) ou le supprime (si l'expéditeur a spécifié MQRO_DISCARD_MSG).

Pour les messages persistants, si l'expéditeur a spécifié MQRO_DEAD_LETTER_Q, mais que l'insertion dans la file d'attente de rebut échoue ou qu'il n'y a pas de file d'attente de rebut, le message d'origine est laissé dans la file d'attente de transmission et le message de rapport n'est pas généré. Le message d'origine est également laissé dans la file d'attente de transmission si le message de rapport ne peut pas être généré avec succès.

La zone *Feedback* de la structure MQDLH au début du message dans la file d'attente de rebut indique la raison pour laquelle le message a été inséré dans la file d'attente de rebut ; ce code retour est également utilisé dans le descripteur de message du message de rapport d'exception (s'il a été demandé par l'expéditeur).

- Pour l'exit de relance de message de canal, cette valeur indique que l'agent MCA n'attend pas et ne relance pas le message ; à la place, l'agent MCA poursuit immédiatement son traitement d'échec normal (le message est placé dans la file d'attente des messages non livrés ou supprimé, comme indiqué par l'expéditeur du message).
- Pour l'exit de définition automatique de canal, MQXCC_OK ou MQXCC_SUPPRESS_FUNCTION doit être spécifié. Si aucune de ces valeurs n'est spécifiée, MQXCC_SUPPRESS_FUNCTION est utilisée par défaut et la définition automatique est abandonnée.

Cette réponse n'est pas prise en charge pour les exits d'émission et de réception de canal.

MQXCC_SEND_SEC_MSG
Envoyer un message de sécurité.

Cette valeur ne peut être définie que par un exit de sécurité de canal. Il indique que l'exit a fourni un message de sécurité qui doit être transmis au partenaire.

MQXCC_SEND_AND_REQUEST_SEC_MSG
Envoyer un message de sécurité nécessitant une réponse.

Cette valeur ne peut être définie que par un exit de sécurité de canal. Il indique

- que l'exit a fourni un message de sécurité qui peut être transmis au partenaire, et
- que l'exit requiert une réponse du partenaire. Si aucune réponse n'est reçue, le canal doit être arrêté, car l'exit n'a pas encore décidé si les communications peuvent se poursuivre.

MQXCC_SUPPRESS_EXIT
Supprimer l'exit.

- Cette valeur peut être définie par tous les types d'exit de canal autres qu'un exit de sécurité ou un exit de définition automatique. Il supprime tout autre appel de cet exit (comme si son nom était vide dans la définition de canal), jusqu'à l'arrêt du canal, lorsque l'exit est de nouveau appelé avec un *ExitReason* de MQXR_TERM.
- Si un exit de relance de message renvoie cette valeur, les nouvelles tentatives de messages pour les messages suivants sont contrôlées par les attributs de canal *MsgRetryCount* et *MsgRetryInterval* de manière normale. Pour le message en cours, l'agent MCA effectue le nombre de relances en attente, à des intervalles définis par l'attribut de canal *MsgRetryInterval* , mais uniquement si le code raison est un code raison que l'agent MCA relancera normalement (voir la zone *MsgRetryCount* décrite dans «MQCD-Définition de canal», à la page 1045). Le nombre de nouvelles tentatives en attente correspond à la valeur de l'attribut *MsgRetryCount* , moins le nombre de fois où l'exit a renvoyé MQXCC_OK pour le message en cours ; si ce nombre est négatif, aucune nouvelle tentative n'est effectuée par l'agent MCA pour le message en cours.

MQXCC_CANAL_FERMÉ
Fermez le canal.

Cette valeur peut être définie par n'importe quel type d'exit de canal, à l'exception d'un exit de définition automatique.

Si le partage de conversations n'est pas activé, cette valeur ferme le canal.

Si le partage de conversations est activé, cette valeur met fin à la conversation. Si cette conversation est la seule conversation sur le canal, le canal se ferme également.

Cette zone est une zone d'entrée-sortie de l'exit.

ExitResponse2 (MQLONG)
Cette zone indique la réponse secondaire de l'exit.

Cette zone est définie sur zéro à l'entrée de la routine d'exit.Il peut être défini par l'exit pour fournir des informations supplémentaires aux fonctions de canal WebSphere MQ . Il n'est pas utilisé par l'exit de définition automatique.

L'exit peut définir une ou plusieurs des valeurs suivantes. Si plusieurs valeurs sont requises, elles sont ajoutées. Les combinaisons qui ne sont pas valides sont notées ; d'autres combinaisons sont autorisées.

MQXR2_PUT_WITH_DEF_ACTION
Insertion avec l'action par défaut.

Cette valeur est définie par l'exit de message de canal du récepteur. Il indique que le message doit être inséré avec l'action par défaut de l'agent MCA, c'est-à-dire soit l'ID utilisateur par défaut de l'agent MCA, soit le contexte *UserIdentifier* dans le MQMD (descripteur de message) du message.

La valeur est zéro, ce qui correspond à la valeur initiale définie lors de l'appel de l'exit. La constante est fournie à des fins de documentation.

MQXR2_PUT_WITH_DEF_USERID
Insertion avec l'ID utilisateur par défaut.

Cette valeur ne peut être définie que par l'exit de message de canal du récepteur. Il indique que le message doit être inséré avec l'ID utilisateur par défaut de l'agent MCA.

MQXR2_PUT_WITH_MSG_USERID
Insertion avec l'ID utilisateur du message.

Cette valeur ne peut être définie que par l'exit de message de canal du récepteur. Il indique que le message doit être inséré avec le contexte *UserIdentifier* dans le MQMD (descripteur de message) du message (il peut avoir été modifié par l'exit).

Un seul des éléments MQXR2_PUT_WITH_DEF_ACTION, MQXR2_PUT_WITH_DEF_USERIDet MQXR2_PUT_WITH_MSG_USERID doit être défini.

MQXR2_USE_AGENT_BUFFER
Utilisez la mémoire tampon de l'agent.

Cette valeur indique que les données à transmettre se trouvent dans *AgentBuffer*et non dans *ExitBufferAddr*.

La valeur est zéro, ce qui correspond à la valeur initiale définie lors de l'appel de l'exit. La constante est fournie à des fins de documentation.

MQXR2_USE_EXIT_BUFFER
Utilisez la mémoire tampon d'exit.

Cette valeur indique que les données à transmettre se trouvent dans *ExitBufferAddr*et non dans *AgentBuffer*.

Un seul des paramètres MQXR2_USE_AGENT_BUFFER et MQXR2_USE_EXIT_BUFFER doit être défini.

MQXR2_DEFAULT_CONTINUATION
Continuation par défaut.

La suite de l'exit suivant dans la chaîne dépend de la réponse du dernier exit appelé:

- Si MQXCC_SUPPRESS_FUNCTION ou MQXCC_CLOSE_CHANNEL sont renvoyés, aucun autre exit de la chaîne n'est appelé.
- Sinon, l'exit suivant de la chaîne est appelé.

MQXR2_CONTINUE_CHAIN
Passez à l'exit suivant.

MQXR2_SUPPRESS_CHAIN
Ignorer les exits restants dans la chaîne.

Il s'agit d'une zone d'entrée-sortie pour l'exit.

Commentaires en retour (MQLONG)
Cette zone indique le code retour.

Cette zone est définie sur MQFB_NONE à l'entrée de la routine d'exit.

Si un exit de message de canal associe la zone *ExitResponse* à la valeur MQXCC_SUPPRESS_FUNCTION, la zone *Feedback* indique le code retour qui identifie la raison pour laquelle le message a été placé dans la file d'attente de rebut (message non distribué) et est également utilisée pour envoyer un rapport d'exception, le cas échéant. Dans ce cas, si la zone *Feedback* est MQFB_NONE, le code retour suivant est utilisé:

MQFB_STOPPED_BY_MSG_EXIT
Message arrêté par l'exit de message de canal.

La valeur renvoyée dans cette zone par les exits de sécurité de canal, d'envoi, de réception et de relance de message n'est pas utilisée par l'agent MCA.

La valeur renvoyée dans cette zone par les exits de définition automatique n'est pas utilisée si *ExitResponse* est MQXCC_OK, mais elle est utilisée pour le paramètre *AuxErrorDataInt1* dans le message d'événement.

Il s'agit d'une zone d'entrée-sortie de l'exit.

Longueur MaxSegment(MQLONG)
Cette zone indique la longueur maximale en octets pouvant être envoyée en une seule transmission.

Il n'est pas utilisé par l'exit de définition automatique. Il est intéressant pour un exit d'émission de canal, car cet exit doit s'assurer qu'il n'augmente pas la taille d'un segment de transmission à une valeur supérieure à *MaxSegmentLength*. La longueur inclut les 8 octets initiaux que l'exit ne doit pas modifier. La valeur est négociée entre les fonctions de canal WebSphere MQ lorsque le canal est lancé. Pour plus d'informations sur les longueurs de segment, voir Ecriture de programmes d'exit de canal .

La valeur de cette zone n'est pas significative si *ExitReason* est MQXR_INIT.

Il s'agit d'une zone d'entrée de l'exit.

Zone ExitUser(MQBYTE16)
Cette zone indique la zone utilisateur de l'exit-une zone disponible pour l'exit.

Il est initialisé à zéro binaire avant le premier appel de l'exit (qui a un *ExitReason* défini sur MQXR_INIT), puis toutes les modifications apportées à cette zone par l'exit sont conservées entre les appels de l'exit.

La valeur suivante est définie:

MQXUA_NONE
Aucune information utilisateur.

La valeur est zéro binaire pour la longueur de la zone.

Pour le langage de programmation C, la constante MQXUA_NONE_ARRAY est également définie ; cette constante a la même valeur que MQXUA_NONE, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ_EXIT_USER_AREA_LENGTH. Il s'agit d'une zone d'entrée-sortie pour l'exit.

ExitData (MQCHAR32)
Cette zone indique les données d'exit.

Cette zone est définie à l'entrée de la routine d'exit pour indiquer que les fonctions de canal WebSphere MQ proviennent de la définition de canal.Si aucune information de ce type n'est disponible, cette zone est vide.

La longueur de cette zone est donnée par MQ_EXIT_DATA_LENGTH.

Il s'agit d'une zone d'entrée de l'exit.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCXP_VERSION_2.

MsgRetry(MQLONG)
Cette zone indique le nombre de fois où le message a été relancé.

La première fois que l'exit est appelé pour un message particulier, cette zone a la valeur zéro (aucune nouvelle tentative n'a encore été effectuée).Lors de chaque appel ultérieur de l'exit pour ce message, la valeur est incrémentée d'un par l'agent MCA.

Il s'agit d'une zone d'entrée de l'exit. La valeur de cette zone n'est pas significative si *ExitReason* est MQXR_INIT. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_2.

Intervalle MsgRetry(MQLONG)
Cette zone indique l'intervalle minimal en millisecondes après lequel l'opération d'insertion est relancée.

La première fois que l'exit est appelé pour un message particulier, cette zone contient la valeur de l'attribut de canal *MsgRetryInterval* .L'exit peut laisser la valeur inchangée ou la modifier pour spécifier un intervalle de temps différent en millisecondes. Si l'exit renvoie MQXCC_OK dans *ExitResponse*, l'agent MCA attend au moins cet intervalle de temps avant de relancer l'opération MQOPEN ou MQPUT. L'intervalle de temps spécifié doit être supérieur ou égal à zéro.

La seconde fois et les suivantes où l'exit est appelé pour ce message, cette zone contient la valeur renvoyée par l'appel précédent de l'exit.

Si la valeur renvoyée dans la zone *MsgRetryInterval* est inférieure à zéro ou supérieure à 999 999 999 et que *ExitResponse* est MQXCC_OK, l'agent MCA ignore la zone *MsgRetryInterval* dans MQCXP et attend à la place l'intervalle spécifié par l'attribut de canal *MsgRetryInterval* .

Il s'agit d'une zone d'entrée-sortie pour l'exit. La valeur de cette zone n'est pas significative si *ExitReason* est MQXR_INIT. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_2.

MsgRetryRaison (MQLONG)
Cette zone indique le code raison de la tentative précédente d'insertion du message.

Cette zone est le code raison de la tentative précédente d'insertion du message ; il s'agit de l'une des valeurs MQRC_ *.

Il s'agit d'une zone d'entrée de l'exit. La valeur de cette zone n'est pas significative si *ExitReason* est MQXR_INIT. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_2.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCXP_VERSION_3.

HeaderLength (MQLONG)
Cette zone indique la longueur des informations d'en-tête.

Cette zone est pertinente uniquement pour un exit de message et un exit de relance de message.La valeur correspond à la longueur des structures d'en-tête de routage au début des données de message ; il s'agit de la structure MQXQH, de MQMDE (en-tête d'extension de description de message) et (pour un message de liste de distribution) de la structure MQDH et des tableaux d'enregistrements MQOR et MQPMR qui suivent la structure MQXQH.

L'exit de message peut examiner ces informations d'en-tête et les modifier si nécessaire, mais les données renvoyées par l'exit doivent toujours être au format correct. L'exit ne doit pas, par exemple, chiffrer ou compresser les données d'en-tête à l'extrémité émettrice, même si l'exit de message à l'extrémité réceptrice effectue des modifications de compensation.

Si l'exit de message modifie les informations d'en-tête de manière à modifier sa longueur (par exemple, en ajoutant une autre destination à un message de liste de distribution), il doit modifier la valeur de *HeaderLength* en conséquence avant de renvoyer le message.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La valeur de cette zone n'est pas significative si *ExitReason* est MQXR_INIT. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_3.

PartnerName (MQCHAR48)
Cette zone indique le nom du partenaire.

Nom du partenaire, comme suit:

- Pour les canaux SVRCONN, il s'agit de l'ID utilisateur connecté sur le client.
- Pour tous les autres types de canal, il s'agit du nom du gestionnaire de files d'attente du partenaire.

Lorsque l'exit est initialisé, cette zone est vide car le gestionnaire de files d'attente ne connaît pas le nom du partenaire tant que la négociation initiale n'a pas eu lieu.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_3.

FAPLevel (MQLONG)
Niveau des formats et protocoles négociés.

Il s'agit d'une zone d'entrée de l'exit. Les modifications apportées à cette zone ne doivent être effectuées que sur instruction du service IBM .La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_3.

CapabilityFlags (MQLONG)
Cette zone indique les indicateurs de capacité.

Les éléments suivants sont définis:

MQCF_AUCUN
Aucun indicateur.

LIST_LISTES MQCF_DIST
Listes de distribution prises en charge.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_3.

ExitNumber (MQLONG)
Cette zone indique le nombre ordinal de l'exit.

Nombre ordinal de l'exit, dans le type défini dans *ExitId*.Par exemple, si l'exit appelé est le troisième exit de message défini, cette zone contient la valeur 3. Si le type d'exit est un type pour lequel une liste d'exits ne peut pas être définie (par exemple, un exit de sécurité), cette zone a la valeur 1.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_3.

Les zones suivantes de cette structure ne sont pas présentes si *Version* est inférieur à MQCXP_VERSION_5.

ExitSpace (MQLONG)
Cette zone indique le nombre d'octets dans la mémoire tampon de transmission réservée à l'exit à utiliser.

Cette zone est pertinente uniquement pour un exit d'émission.Il indique la quantité d'espace en octets que les fonctions de canal WebSphere MQ réservent dans la mémoire tampon de transmission pour l'exit à utiliser. Ce champ permet à la sortie d'ajouter à la mémoire tampon de transmission une petite quantité de données (typiquement ne dépassant pas quelques centaines d'octets) à utiliser par une sortie de réception complémentaire à l'autre extrémité. Les données ajoutées par l'exit d'émission doivent être supprimées par l'exit de réception.

La valeur est toujours zéro sous z/OS.

Remarque : Cette fonction ne doit pas être utilisée pour envoyer de grandes quantités de données, car elle risque de dégrader les performances, voire d'empêcher le fonctionnement du canal.

En définissant *ExitSpace* , l'exit est garanti qu'il y a toujours au moins le nombre d'octets disponibles dans la mémoire tampon de transmission à utiliser par l'exit. Toutefois, l'exit peut utiliser moins que la quantité réservée ou plus que la quantité réservée s'il y a de l'espace disponible dans la mémoire tampon de transmission. L'espace de sortie dans la mémoire tampon est fourni après les données existantes.

ExitSpace peut être défini par l'exit uniquement lorsque *ExitReason* a la valeur MQXR_INIT; dans tous les autres cas, la valeur renvoyée par l'exit est ignorée. En entrée de l'exit, *ExitSpace* est égal à zéro pour l'appel MQXR_INIT et correspond à la valeur renvoyée par l'appel MQXR_INIT dans les autres cas.

Si la valeur renvoyée par l'appel MQXR_INIT est négative ou qu'il y a moins de 1024 octets disponibles dans la mémoire tampon de transmission pour les données de message après la réservation de l'espace d'exit demandé pour tous les exits d'émission de la chaîne, l'agent MCA génère un message d'erreur et ferme le canal. De même, si, lors du transfert de données, les exits de la chaîne d'exit d'émission allouent plus d'espace utilisateur qu'ils n'en réservent de sorte qu'il reste moins de 1024 octets dans la mémoire tampon de transmission pour les données de message, l'agent MCA génère un message d'erreur et ferme le canal. La limite de 1024 permet aux flux de contrôle et d'administration du canal d'être traités par la chaîne d'exits d'émission, sans qu'il soit nécessaire de segmenter les flux.

Il s'agit d'une zone d'entrée-sortie pour l'exit si *ExitReason* est MQXR_INIT, et d'une zone d'entrée dans tous les autres cas. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_5.

ID SSLCertUser(MQCHAR12)
Cette zone indique l' UserId associé au certificat distant.

Elle est vide sur toutes les plateformes à l'exception de z/OS

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_6.

SSLRemCertIssNameLongueur (MQLONG)
Cette zone indique la longueur en octets du nom distinctif complet de l'émetteur du certificat distant désigné par SSLCertRemoteIssuerNamePtr.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_6. La valeur est zéro s'il ne s'agit pas d'un canal SSL.

SSLRemCertIssNamePtr (PMQVOID)
Cette zone indique l'adresse du nom distinctif complet de l'émetteur du certificat distant.

Sa valeur est le pointeur null s'il ne s'agit pas d'un canal SSL.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_6.

Remarque : Le comportement des exits de sécurité de canal lors de la détermination du nom distinctif du sujet et du nom distinctif de l'émetteur a été modifié dans la version WebSphere MQ v7.1. Pour plus d'informations, voir Programmes d'exit de sécurité de canal.

SecurityParms (PMQCSP)
Cette zone indique l'adresse de la structure MQSCP utilisée pour spécifier un ID utilisateur et un mot de passe.

La valeur initiale de cette zone est le pointeur null.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_6.

Compression CurHdr(MQLONG)
Cette zone indique la technique actuellement utilisée pour compresser les données d'en-tête.

Il est défini sur l'une des valeurs suivantes:

MQCOMPRESS_NONE
Aucune compression de données d'en-tête n'est effectuée.

MQCOMPRESS_SYSTEM
La compression de données d'en-tête est effectuée.

La valeur peut être modifiée par l'exit de message d'un canal émetteur en une des valeurs prises en charge négociées accessibles à partir de la zone de liste HdrCompde MQCD. Cela permet la technique utilisée pour compresser les données d'en-tête à choisir pour chaque message en fonction du contenu du message. La valeur modifiée est utilisée pour le message en cours uniquement. Le canal se termine si l'attribut est modifié avec une valeur non prise en charge. La valeur est ignorée si elle est modifiée en dehors de l'exit de message d'un canal émetteur.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_6.

Compression CurMsg(MQLONG)
Cette zone indique la technique actuellement utilisée pour compresser les données de message.

Il est défini sur l'une des valeurs suivantes:

MQCOMPRESS_NONE
Aucune compression de données d'en-tête n'est effectuée.

MQCOMPRESS_RLE
La compression de données de message est effectuée à l'aide de l'algorithme RLE.

MQCOMPRESS_ZLIBFAST
La compression de données de message est effectuée à l'aide de la technique de compression zlib. Il est préférable d'utiliser une durée de compression rapide.

MQCOMPRESS_ZLIBHIGH
La compression de données de message est effectuée à l'aide de la technique de compression zlib. Il est préférable d'utiliser une compression de haut niveau.

La valeur peut être modifiée par l'exit de message d'un canal émetteur sur l'une des valeurs prises en charge négociées accessibles à partir de la zone de liste MsgCompdu MQCD. Cela permet la technique utilisée pour compresser les données de message à décider pour chaque message en fonction du contenu du message. La valeur modifiée est utilisée pour le message en cours uniquement. Le canal se termine si l'attribut est modifié avec une valeur non prise en charge. La valeur est ignorée si elle est modifiée en dehors de l'exit de message d'un canal émetteur.

Il s'agit d'une zone d'entrée-sortie pour l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_6.

Hconn (MQHCONN)
Cette zone indique le descripteur de connexion utilisé par l'exit s'il doit effectuer des appels MQI dans l'exit.

Cette zone n'est pas pertinente pour les exits exécutés sur des canaux de connexion client, où elle contient la valeur MQHC_UNUSABLE_HCONN (-1).

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_7.

SharingConversations (MQBOOL)
Cette zone indique si la conversation est la seule qui peut être en cours d'exécution sur cette instance de canal ou si plusieurs conversations peuvent être en cours d'exécution sur cette instance de canal.

Elle indique également si le programme d'exit est soumis au risque que le MQCD soit modifié par un autre programme d'exit exécuté en même temps.

Cette zone concerne uniquement les programmes d'exit exécutés sur des canaux de connexion client ou serveur.

Il est défini sur l'une des valeurs suivantes:

FALSE
L'instance d'exit est la seule instance d'exit qui peut actuellement être en cours d'exécution sur cette instance de canal. Cela permet à l'exit de mettre à jour en toute sécurité les zones MQCD sans conflit avec d'autres exits s'exécutant sur d'autres instances de canal. Le fait que les modifications apportées aux zones MQCD soient traitées par le canal est défini par la table des zones MQCD dans «Modification des zones MQCD dans un exit de canal», à la page 1083.

TRUE
L'instance d'exit n'est pas la seule instance d'exit qui peut être en cours d'exécution sur cette instance de canal. Les modifications apportées à MQCD ne sont pas traitées par le canal, à l'exception des modifications répertoriées dans la table des zones MQCD dans «Modification des zones MQCD dans un exit de canal», à la page 1083 pour les raisons d'exit autres que MQXR_INIT. Si cet exit met à jour les zones MQCD, vérifiez qu'il n'y a pas de conflit d'autres exits qui s'exécutent simultanément sur d'autres conversations en fournissant une sérialisation parmi les exits qui s'exécutent sur cette instance de canal.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si *Version* est inférieur à MQCXP_VERSION_7.

MCAUserSource (MQLONG)
Cette zone indique la source de l'ID utilisateur MCA fourni.

Il peut contenir l'une des valeurs suivantes:

MQUSRC_MAP
L'ID utilisateur est indiqué dans l'attribut MCAUSER.

MQUSRC_CHANNEL
L'ID utilisateur est transmis à partir du partenaire entrant ou spécifié dans la zone MCAUSER définie dans l'objet canal.

Il s'agit d'une zone d'entrée de l'exit. La zone n'est pas présente si la version est inférieure à MQCXP_VERSION_8.

Points pEntry(PMQIEP)
Cette zone indique l'adresse du point d'entrée d'interface pour l'appel MQI ou DCI.

La zone n'est pas présente si *Version* est inférieure à MQCXP_VERSION_8.

Déclaration C
Cette déclaration est la déclaration C pour la structure MQCXP.

```
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 */
;
```

Déclaration COBOL

Cette déclaration est la déclaration COBOL pour la structure MQCXP.

```
**   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.
```

Déclaration RPG (ILE)

Cette déclaration est la déclaration RPG pour la structure MQCXP.

```
     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
```

Déclaration assembleur System/390

Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQCXP.

```
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-Descripteur d'attente d'exit

La structure MQXWD est un paramètre d'entrée-sortie dans l'appel MQXWAIT.

Cette structure est prise en charge uniquement sur z/OS.

Référence associée

«Zones», à la page 1100
Cette rubrique répertorie toutes les zones de la structure MQXWD et décrit chaque zone.

«Déclaration C», à la page 1101
Cette déclaration est la déclaration C pour la structure MQXWD.

«Déclaration assembleur System/390», à la page 1101
Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQXWD.

Zones

Cette rubrique répertorie toutes les zones de la structure MQXWD et décrit chaque zone.

StrucId (MQCHAR4)
Cette zone indique l'identificateur de structure.

La valeur doit être:

ID_STRUCD_MQXWD_
Identificateur de la structure de descripteur d'attente d'exit.

Pour le langage de programmation C, la constante MQXWD_STRUC_ID_ARRAY est également définie ; cette constante a la même valeur que MQXWD_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

La valeur initiale de cette zone est MQXWD_STRUC_ID.

Version (MQLONG)
Cette zone indique le numéro de version de la structure.

La valeur doit être:

MQXWD_VERSION_1
Numéro de version de la structure de descripteur d'attente d'exit.

La valeur initiale de cette zone est MQXWD_VERSION_1.

Reserved1 (MQLONG)
Cette zone est réservée. Sa valeur doit être zéro.

Il s'agit d'une zone d'entrée.

Reserved2 (MQLONG)
Cette zone est réservée. Sa valeur doit être zéro.

Il s'agit d'une zone d'entrée.

Reserved3 (MQLONG)
Cette zone est réservée. Sa valeur doit être zéro.

Il s'agit d'une zone d'entrée.

ECB (MQLONG)
Cette zone indique le bloc de contrôle d'événement à attendre.

Cette zone indique le bloc de contrôle d'événement (ECB) à attendre.Il doit être défini sur zéro avant que l'appel MQXWAIT ne soit émis ; une fois terminé, il contient le code postal.

Cette zone est une zone d'entrée/sortie.

Déclaration C
Cette déclaration est la déclaration C pour la structure MQXWD.

```
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 */
};
```

Déclaration assembleur System/390
Cette déclaration est la déclaration d'assembleur System/390 pour la structure MQXWD.

```
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)
```

Référence d'exit API

Cette section fournit des informations de référence présentant principalement un intérêt pour un programmeur qui écrit des exits API.

Remarques générales sur l'utilisation

Remarques :

1. Toutes les fonctions d'exit peuvent émettre l'appel MQXEP ; cet appel est conçu spécifiquement pour être utilisé à partir des fonctions d'exit d'API.
2. La fonction MQ_INIT_EXIT ne peut pas émettre d'appels MQ autres que MQXEP.
3. Vous ne pouvez pas émettre l'appel MQDISC pour la connexion en cours.
4. Si une fonction d'exit émet l'appel MQCONN ou l'appel MQCONNX avec l'option MQCNO_HANDLE_SHARE_NONE, l'appel se termine avec le code anomalie MQRC_ALREADY_CONNECTED et le descripteur renvoyé est identique à celui transmis à l'exit en tant que paramètre.
5. En général, lorsqu'une fonction d'exit API émet un appel MQI, les exits API ne sont pas appelés de manière récursive. Toutefois, si une fonction d'exit émet l'appel MQCONNX avec les options MQCNO_HANDLE_SHARE_BLOCK ou MQCNO_HANDLE_SHARE_NO_BLOCK, l'appel renvoie un nouveau descripteur partagé. Cela fournit à la suite d'exit un descripteur de connexion qui lui est propre et donc une unité de travail indépendante de l'unité de travail de l'application. La suite de sorties peut utiliser cet indicateur pour insérer et extraire des messages dans sa propre unité de travail, et valider ou abandonner cette unité de travail ; tout cela peut être fait sans affecter de quelque manière que ce soit l'unité de travail de l'application.

   Etant donné que la fonction d'exit utilise un descripteur de connexion différent de celui utilisé par l'application, les appels MQ émis par la fonction d'exit entraînent l'appel des fonctions d'exit d'API appropriées. Les fonctions de sortie peuvent donc être appelées de manière récursive. Notez que la zone *ExitUserArea* dans MQAXP et la zone de chaîne d'exit ont une portée de descripteur de connexion. Par conséquent, une fonction d'exit ne peut pas utiliser ces zones pour signaler à une autre instance qu'elle a appelée de manière récursive qu'elle est déjà active.
6. Les fonctions de sortie peuvent également insérer et extraire des messages dans l'unité d'oeuvre de l'application. Lorsque l'application valide ou annule l'unité de travail, tous les messages de l'unité de travail sont validés ou annulés ensemble, quelle que soit la personne qui les a placés dans l'unité de travail (application ou fonction de sortie). Cependant, l'exit peut amener l'application à dépasser les limites du système plus tôt que dans le cas contraire (par exemple, en dépassant le nombre maximal de messages non validés dans une unité de travail).

   Lorsqu'une fonction d'exit utilise l'unité de travail de l'application de cette manière, la fonction d'exit doit généralement éviter d'émettre l'appel MQCMIT, car elle valide l'unité de travail de l'application et peut nuire au bon fonctionnement de l'application. Toutefois, la fonction d'exit peut parfois avoir besoin d'émettre l'appel MQBACK si la fonction d'exit rencontre une erreur grave qui empêche la validation de l'unité de travail (par exemple, une erreur lors de l'insertion d'un message dans l'unité de travail de l'application). Lorsque MQBACK est appelé, veillez à ce que les limites de l'unité de travail de l'application ne soient pas modifiées. Dans ce cas, la fonction d'exit doit définir les valeurs appropriées pour s'assurer que le code achèvement MQCC_WARNING et le code anomalie MQRC_BACKED_OUT sont renvoyés à l'application, afin que l'application puisse détecter le fait que l'unité de travail a été annulée.

Si une fonction d'exit utilise le descripteur de connexion de l'application pour émettre des appels MQ , ces appels ne génèrent pas eux-mêmes d'autres appels de fonctions d'exit d'API.

7. Si une fonction d'exit MQXR_BEFORE s'arrête de manière anormale, le gestionnaire de files d'attente peut être en mesure d'effectuer une reprise après incident. Si tel est le cas, le gestionnaire de files d'attente poursuit le traitement comme si la fonction d'exit avait renvoyé MQXCC_FAILED. Si le gestionnaire de files d'attente ne peut pas effectuer de reprise, l'application est arrêtée.
8. Si une fonction d'exit MQXR_AFTER s'arrête de manière anormale, le gestionnaire de files d'attente peut être en mesure d'effectuer une reprise après incident. Si tel est le cas, le gestionnaire de files d'attente poursuit le traitement comme si la fonction d'exit avait renvoyé MQXCC_FAILED. Si le gestionnaire de files d'attente ne peut pas effectuer de reprise, l'application est arrêtée. Sachez que dans ce dernier cas, les messages extraits en dehors d'une unité de travail sont perdus (c'est la même situation que l'application qui échoue immédiatement après la suppression d'un message de la file d'attente).
9. Le processus MCA effectue une validation en deux phases.

   Si un exit API intercepte un MQCMIT d'un processus MCA préparé et tente d'effectuer une action dans l'unité de travail, l'action échoue avec le code anomalie MQRC_UOW_NOT_AVAILABLE.
10. Dans un environnement multi-installation, le seul moyen de disposer d'un exit qui fonctionne à la fois avec Websphere MQ version 7.0 et version 7.1 consiste à écrire l'exit d'une manière qui se lie à la version 7.0 avec mqm.Lib et, pour les exits non principaux ou relocalisés, à s'assurer que l'application trouve la valeur correcte mqm.Lib pour l'installation à laquelle le gestionnaire de files d'attente est actuellement associé, avant le lancement de l'application. (Par exemple, exécutez la commande **setmqenv -m QM** avant de lancer l'application, même si le gestionnaire de files d'attente appartient à une installation de version 7.0 .)
11. Lorsque plusieurs installations de IBM WebSphere MQ sont disponibles, utilisez les exits écrits pour une version antérieure de IBM WebSphere MQ, car les nouvelles fonctionnalités ajoutées dans la version ultérieure risquent de ne pas fonctionner avec les versions antérieures. Pour plus d'informations sur les modifications entre les éditions, voir Nouveautés de WebSphere MQ 7.5.

Structure des paramètres d'exit API IBM WebSphere MQ (MQAXP)

La structure MQAXP, un bloc de contrôle externe, est utilisée comme paramètre d'entrée ou de sortie de l'exit API. Cette rubrique fournit également des informations sur le fonctionnement de l'exit de processus des gestionnaires de files d'attente.

MQAXP possède la déclaration C suivante:

```
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 */
};
```

La liste de paramètres suivante est transmise lorsque des fonctions d'un exit API sont appelées:

StrucId (MQCHAR4)-entrée
Identificateur de la structure du paramètre d'exit, avec la valeur:

```
MQAXP_STRUC_ID.
```

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

Version (MQLONG)-entrée
Numéro de version de la structure, avec la valeur:

MQAXP_VERSION_1
Structure des paramètres d'exit d'API version 1.

MQAXP_VERSION_2
Structure des paramètres d'exit d'API version 2.

MQAXP_CURRENT_VERSION
Numéro de version en cours de la structure de paramètres d'exit d'API.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

ExitId (MQLONG)-entrée
Identificateur de l'exit, défini à l'entrée de la routine d'exit, indiquant le type de l'exit:

MQXT_API_EXIT
Exit API.

ExitReason (MQLONG)-entrée
Motif de l'appel de l'exit, défini à l'entrée de chaque fonction d'exit:

MQXR_CONNECTION
L'exit est appelé pour s'initialiser avant un appel MQCONN ou MQCONNX ou pour se terminer après un appel MQDISC.

MQXR_AVANT
L'exit est appelé avant l'exécution d'un appel API ou avant la conversion de données sur une requête MQGET.

MQXR_APRES
L'exit est appelé après l'exécution d'un appel API.

ExitResponse (MQLONG)-sortie
La réponse de l'exit, initialisée à l'entrée de chaque fonction d'exit pour:

MQXCC_OK
Continuez normalement.

Cette zone doit être définie par la fonction d'exit pour communiquer au gestionnaire de files d'attente le résultat de l'exécution de la fonction d'exit. Il doit s'agir de l'une des valeurs suivantes :

MQXCC_OK
La fonction d'exit a abouti. Continuez normalement.

Cette valeur peut être définie par toutes les fonctions d'exit MQXR_ *. ExitResponse2 est utilisé pour décider si les fonctions de sortie doivent être appelées ultérieurement dans la chaîne.

MQXCC_ECHEC
La fonction d'exit a échoué en raison d'une erreur.

Cette valeur peut être définie par toutes les fonctions d'exit MQXR_ *. Le gestionnaire de files d'attente définit CompCode sur MQCC_FAILED et la raison sur:

- MQRC_API_EXIT_INIT_ERROR si la fonction est MQ_INIT_EXIT
- MQRC_API_EXIT_TERM_ERROR si la fonction est MQ_TERM_EXIT
- MQRC_API_EXIT_ERROR pour toutes les autres fonctions d'exit

Les valeurs définies peuvent être modifiées par une fonction de sortie plus tard dans la chaîne.

ExitResponse2 est ignoré ; le gestionnaire de files d'attente poursuit le traitement comme si MQXR2_SUPPRESS_CHAIN avait été renvoyé.

MQXCC_SUPPRESS_FUNCTION
Supprimez la fonction d'API WebSphere MQ .

Cette valeur ne peut être définie que par une fonction de sortie MQXR_BEFORE. Il ignore l'appel d'API. S'il est renvoyé par MQ_DATA_CONV_ON_GET_EXIT, la conversion de données est ignorée. Le gestionnaire de files d'attente définit CompCode sur MQCC_FAILED et Cause sur MQRC_SUPPRESSED_BY_EXIT, mais les valeurs définies peuvent être modifiées par une fonction d'exit plus loin dans la chaîne. Les autres paramètres de l'appel restent tels que l'exit les a laissés. ExitResponse2 est utilisé pour décider si les fonctions de sortie doivent être appelées ultérieurement dans la chaîne.

Si cette valeur est définie par une fonction d'exit MQXR_AFTER ou MQXR_CONNECTION, le gestionnaire de files d'attente poursuit le traitement comme si MQXCC_FAILED avait été renvoyé.

MQXCC_LISTE_FONCTION
Ignorez la fonction d'API WebSphere MQ .

Cette valeur ne peut être définie que par une fonction de sortie MQXR_BEFORE. Il ignore l'appel d'API. S'il est renvoyé par MQ_DATA_CONV_ON_GET_EXIT, la conversion de données est ignorée. La fonction d'exit doit définir CompCode et Raison sur les valeurs à renvoyer à l'application, mais les valeurs définies peuvent être modifiées par une fonction d'exit plus tard dans la chaîne. Les autres paramètres de l'appel restent tels que l'exit les a laissés. ExitResponse2 est utilisé pour décider si les fonctions de sortie doivent être appelées ultérieurement dans la chaîne.

Si cette valeur est définie par une fonction d'exit MQXR_AFTER ou MQXR_CONNECTION, le gestionnaire de files d'attente poursuit le traitement comme si MQXCC_FAILED avait été renvoyé.

MQXCC_SUPPRESS_EXIT
Supprimez toutes les fonctions d'exit appartenant à l'ensemble d'exits.

Cette valeur ne peut être définie que par les fonctions d'exit MQXR_BEFORE et MQXR_AFTER. Il ignore *tous* les appels ultérieurs des fonctions d'exit appartenant à cet ensemble d'exits pour cette connexion logique. Ce contournement se poursuit jusqu'à ce que la demande de déconnexion logique se produise, lorsque la fonction MQ_TERM_EXIT est appelée avec un ExitReason de MQXR_CONNECTION.

La fonction d'exit doit définir CompCode et Raison sur les valeurs à renvoyer à l'application, mais les valeurs définies peuvent être modifiées par une fonction d'exit plus tard dans la chaîne. Les autres paramètres de l'appel restent tels que l'exit les a laissés. ExitResponse2 est ignoré.

Si cette valeur est définie par une fonction d'exit MQXR_CONNECTION, le gestionnaire de files d'attente poursuit le traitement comme si MQXCC_FAILED avait été renvoyé.

Pour plus d'informations sur l'interaction entre ExitResponse et ExitResponse2et son effet sur le traitement de sortie, voir «Fonctionnement de l'exit de processus des gestionnaires de files d'attente», à la page 1108.

ExitResponse2 (MQLONG)-sortie
Il s'agit d'un code de réponse d'exit secondaire qui qualifie le code de réponse d'exit principal pour les fonctions d'exit MQXR_BEFORE. Il est initialisé pour:

```
MQXR2_DEFAULT_CONTINUATION
```

sur l'entrée d'une fonction d'exit d'appel d'API WebSphere MQ . Il peut ensuite être défini sur l'une des valeurs suivantes:

MQXR2_DEFAULT_CONTINUATION
Indique s'il faut continuer avec l'exit suivant dans la chaîne, en fonction de la valeur de ExitResponse.

Si ExitResponse est MQXCC_SUPPRESS_FUNCTION ou MQXCC_SKIP_FUNCTION, ignorez les fonctions d'exit ultérieurement dans la chaîne MQXR_BEFORE et les fonctions d'exit correspondantes dans la chaîne MQXR_AFTER. Appelez les fonctions d'exit dans la

chaîne MQXR_AFTER qui correspondent aux fonctions d'exit précédemment dans la chaîne MQXR_BEFORE.

Sinon, appelez l'exit suivant dans la chaîne.

MQXR2_SUPPRESS_CHAIN
Supprimez la chaîne.

Ignorez les fonctions d'exit ultérieurement dans la chaîne MQXR_BEFORE et les fonctions d'exit correspondantes dans la chaîne MQXR_AFTER pour cet appel d'API. Appelez les fonctions d'exit dans la chaîne MQXR_AFTER qui correspondent aux fonctions d'exit précédemment dans la chaîne MQXR_BEFORE.

MQXR2_CONTINUE_CHAIN
Passez à l'exit suivant dans la chaîne.

Pour plus d'informations sur l'interaction entre ExitResponse et ExitResponse2et son effet sur le traitement de sortie, voir «Fonctionnement de l'exit de processus des gestionnaires de files d'attente», à la page 1108.

Commentaires en retour (MQLONG)-entrée / sortie
Communiquez les codes retour entre les appels de fonction de sortie. Cette opération est initialisée pour:

```
MQFB_NONE (0)
```

avant d'invoquer la première fonction de la première sortie dans une chaîne.

Les exits peuvent définir cette zone sur n'importe quelle valeur, y compris toute valeur MQFB_ * ou MQRC_ * valide. Les exits peuvent également définir cette zone sur une valeur de retour d'informations définie par l'utilisateur comprise entre MQFB_APPL_FIRST et MQFB_APPL_LAST.

APICallerType (MQLONG)-entrée
Type d'appelant de l'API, indiquant si l'appelant de l'API WebSphere MQ est externe ou interne au gestionnaire de files d'attente: MQXACT_EXTERNAL ou MQXACT_INTERNAL.

Zone ExitUser(MQBYTE16)-entrée / sortie
Zone utilisateur, disponible pour tous les exits associés à un objet ExitInfoparticulier. Il est initialisé sur MQXUA_NONE (zéros binaires pour la longueur de la zone ExitUser) avant d'appeler la première fonction d'exit (MQ_INIT_EXIT) pour hconn. A partir de ce moment, toutes les modifications apportées à cette zone par une fonction d'exit sont conservées dans les appels de fonctions du même exit.

Cette zone est alignée sur un multiple de 4 MQLONGs.

Les exits peuvent également ancrer tout stockage qu'ils allouent à partir de cette zone.

Pour chaque hconn, chaque exit d'une chaîne d'exits possède une zone ExitUserdifférente. La zone ExitUserne peut pas être partagée par les exits d'une chaîne et le contenu de la zone ExitUserd'un exit n'est pas disponible pour un autre exit d'une chaîne.

Pour les programmes C, la constante MQXUA_NONE_ARRAY est également définie avec la même valeur que MQXUA_NONE, mais sous la forme d'un tableau de caractères au lieu d'une chaîne.

La longueur de cette zone est indiquée par MQ_EXIT_USER_AREA_LENGTH.

ExitData (MQCHAR32)-entrée
Données d'exit, définies à l'entrée de chaque fonction d'exit sur les 32 caractères des données spécifiques à l'exit qui sont fournies dans l'exit. Si vous ne définissez aucune valeur dans l'exit, cette zone est vide.

La longueur de cette zone est donnée par MQ_EXIT_DATA_LENGTH.

ExitInfoNom (MQCHAR48)-entrée
Nom des informations d'exit, défini à l'entrée de chaque fonction d'exit sur le nom ApiExit_name spécifié dans les définitions d'exit dans les sections.

ExitPDArea (MQBYTE48)-entrée / sortie
Une zone d'identification des incidents, initialisée à MQXPDA_NONE (zéros binaires pour la longueur de la zone) pour chaque appel d'une fonction d'exit.

Pour les programmes C, la constante MQXPDA_NONE_ARRAY est également définie avec la même valeur que MQXPDA_NONE, mais sous la forme d'un tableau de caractères au lieu d'une chaîne.

Le gestionnaire d'exit écrit toujours cette zone dans la trace WebSphere MQ à la fin d'un exit, même si la fonction aboutit.

La longueur de cette zone est indiquée par MQ_EXIT_PD_AREA_LENGTH.

QMgrName (MQCHAR48)-entrée
Nom du gestionnaire de files d'attente auquel l'application est connectée, qui a appelé un exit suite au traitement d'un appel API WebSphere MQ .

Si le nom d'un gestionnaire de files d'attente fourni sur un appel MQCONN ou MQCONNX est vide, cette zone est toujours définie sur le nom du gestionnaire de files d'attente auquel l'application est connectée, que l'application soit serveur ou client.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

La longueur de cette zone est indiquée par MQ_Q_MGR_NAME_LENGTH.

ExitChainAreaPtr (PMQACH)-entrée/sortie
Permet de communiquer des données entre les appels de différents exits dans une chaîne. Il est défini sur un pointeur NULL avant d'appeler la première fonction (MQ_INIT_EXIT avec ExitReason MQXR_CONNECTION) du premier exit dans une chaîne d'exits. La valeur renvoyée par l'exit lors d'un appel est transmise à l'appel suivant.

Pour plus d'informations sur l'utilisation de la zone de chaîne d'exit, voir «Zone de chaîne d'exit et en-tête de zone de chaîne d'exit (MQACH)», à la page 1111 .

Hconfig (MQHCONFIG)-entrée
Descripteur de configuration représentant l'ensemble des fonctions en cours d'initialisation. Cette valeur est générée par le gestionnaire de files d'attente sur la fonction MQ_INIT_EXIT et est ensuite transmise à la fonction d'exit d'API. Il est défini à l'entrée de chaque fonction d'exit.

Vous pouvez utiliser Hconfig comme pointeur vers la structure MQIEP pour effectuer des appels MQI et DCI. Vous devez vérifier que les 4 premiers octets de HConfig correspondent à l' StrucId de la structure MQIEP avant d'utiliser le paramètre HConfig comme pointeur vers la structure MQIEP.

Fonction (MQLONG)-entrée
Identificateur de la fonction, dont les valeurs valides sont les constantes MQXF_ * décrites dans «Constantes externes», à la page 1113.

Le gestionnaire d'exit définit cette zone sur la valeur correcte, à l'entrée de chaque fonction d'exit, en fonction de l'appel de l'API WebSphere MQ qui a abouti à l'appel de l'exit.

ExitMsgIdentificateur (MQHMSG)-entrée/sortie
Lorsque la fonction est MQXF_GET et que ExitReason est MQXR_AFTER, un descripteur de message valide est renvoyé dans cette zone pour permettre à l'exit d'API d'accéder aux zones de descripteur de message et à toute autre propriété correspondant à la chaîne ExitProperties spécifiée dans la structure MQXEPO lors de l'enregistrement de l'exit d'API.

Toutes les propriétés de descripteur de message qui sont renvoyées dans le descripteur ExitMsgne seront pas disponibles à partir de MsgHandle dans la structure MQGMO, si elle a été spécifiée, ou dans les données de message.

Lorsque la fonction est MQXF_GET et que ExitReason est MQXR_BEFORE, si le programme d'exit définit cette zone sur MQHM_NONE, il supprime le remplissage des propriétés de descripteur ExitMsg.

Cette zone n'est pas définie si la version est inférieure à MQAXP_VERSION_2.

Fonctionnement de l'exit de processus des gestionnaires de files d'attente

Le traitement effectué par le gestionnaire de files d'attente en cas de retour d'une fonction d'exit dépend à la fois de ExitResponse et de ExitResponse2.

Le Tableau 593, à la page 1108 récapitule les combinaisons possibles et leurs effets pour une fonction d'exit MQXR_BEFORE, en indiquant:

- Qui définit les paramètres CompCode et Reason de l'appel API
- Indique si les fonctions d'exit restantes dans la chaîne MQXR_BEFORE et les fonctions d'exit correspondantes dans la chaîne MQXR_AFTER sont appelées
- Indique si l'appel d'API est appelé

Pour une fonction d'exit MQXR_AFTER:

- CompCode et Reason sont définis de la même manière que MQXR_BEFORE
- ExitResponse2 est ignoré (les autres fonctions d'exit de la chaîne MQXR_AFTER sont toujours appelées)
- MQXCC_SUPPRESS_FUNCTION et MQXCC_SKIP_FUNCTION ne sont pas valides

Pour une fonction d'exit MQXR_CONNECTION:

- CompCode et Reason sont définis de la même manière que MQXR_BEFORE
- ExitResponse2 est ignoré
- MQXCC_SUPPRESS_FUNCTION, MQXCC_SKIP_FUNCTION, MQXCC_SUPPRESS_EXIT ne sont pas valides

Dans tous les cas où un exit ou le gestionnaire de files d'attente définit CompCode et Reason, les valeurs définies peuvent être modifiées par un exit appelé ultérieurement ou par l'appel API (si l'appel API est appelé ultérieurement).

Tableau 593. Traitement de l'exit MQXR_BEFORE

Valeur de ExitResponse	CompCode et raison définie par	Valeur de ExitResponse2 (continuation par défaut) Chain	Valeur de l'API ExitResponse2 (continuation par défaut)
MQXCC_OK	quitter	Y	Y
MQXCC_SUPPRESS_EXIT	quitter	Y	Y
MQXCC_SUPPRESS_FUNCTION	gestionnaire de files d'attente	N	N
FONCTION MQXCC_SKIP	quitter	N	N
MQXCC_ECHEC	gestionnaire de files d'attente	N	N

Fonctionnement de l'exit de processus des clients

En général, les clients traitent les fonctions d'exit de la même manière que les applications serveur et l'attribut *QMgrName* de cette structure s'applique que la fonction se trouve sur un serveur ou sur un client.

Toutefois, le client n'ayant aucun concept du fichier *mqs.ini* , les sections *ApiExitCommon* et *APIExitTemplate* ne s'appliquent pas. Seule la section *ApiExitLocal* s'applique et cette section est configurée dans le fichier *mqclient.ini* .

Structure de contexte d'exit API IBM WebSphere MQ (MQAXC)

La structure MQAXC, un bloc de contrôle externe, est utilisée comme paramètre d'entrée pour un exit d'API.

MQAXC possède la déclaration C suivante:

```
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 */
 };
```

Les paramètres de MQAXC sont les suivants:

StrucId (MQCHAR4)-entrée
Identificateur de la structure de contexte d'exit, avec la valeur MQAXC_STRUC_ID. Pour les programmes C, la constante MQAXC_STRUC_ID_ARRAY est également définie, avec la même valeur que MQAXC_STRUC_ID, mais sous la forme d'un tableau de caractères au lieu d'une chaîne.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

Version (MQLONG)-entrée
Numéro de version de la structure, avec la valeur:

MQAXC_VERSION_2
Numéro de version de la structure de contexte d'exit.

MQAXC_CURRENT_VERSION
Numéro de version en cours de la structure de contexte d'exit.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

Environnement (MQLONG)-entrée
Environnement à partir duquel un appel d'API WebSphere MQ a été émis qui a entraîné l'exécution d'une fonction d'exit. Les valeurs admises pour cette zone sont les suivantes:

MQXE_AUTRES
Cette valeur est cohérente avec les appels qu'un exit API voit si l'exit est appelé à partir d'une application serveur. Cela signifie qu'un exit d'API s'exécute sans modification sur un client et ne voit rien de différent.

Si l'exit a réellement besoin de déterminer s'il est en cours d'exécution sur le client, il peut le faire en consultant les zones *ChannelName* et *ChannelDefinition* .

MQXE_MCA
Agent MCA

MQXE_MCA_SVRCONN
Un agent MCA agissant pour le compte d'un client

SERVEUR_COMMAND_MQ
Serveur de commandes

MQXE_MQSC
Interpréteur de commandes runmqsc

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

UserId (MQCHAR12)-entrée
ID utilisateur associé à l'application. En particulier, dans le cas de connexions client, cette zone contient l'ID utilisateur de l'utilisateur adopté par opposition à l'ID utilisateur sous lequel le code de canal est exécuté. Si un ID utilisateur vide provient du client, aucune modification n'est apportée à l'ID utilisateur déjà utilisé. Autrement dit, aucun nouvel ID utilisateur n'est adopté.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit. La longueur de cette zone est indiquée par MQ_USER_ID_LENGTH.

Dans le cas d'un client, il s'agit de l'ID utilisateur envoyé du client au serveur. Notez qu'il peut ne pas s'agir de l'ID utilisateur effectif sur lequel le client s'exécute dans le gestionnaire de files d'attente, car il peut y avoir une configuration MCAUser ou CHLAUTH qui modifie l'ID utilisateur.

SecurityId (MQBYTE40)-entrée
Extension de l'ID utilisateur exécutant l'application. Sa longueur est donnée par MQ_SECURITY_ID_LENGTH.

Dans le cas d'un client, il s'agit de l'ID utilisateur envoyé du client au serveur. Notez qu'il peut ne pas s'agir de l'ID utilisateur effectif sur lequel le client s'exécute dans le gestionnaire de files d'attente, car il peut y avoir une configuration MCAUser ou CHLAUTH qui modifie l'ID utilisateur.

ConnectionName (MQCHAR264)-entrée
Zone de nom de connexion, définie sur l'adresse du client. Par exemple, pour TCP/IP, il s'agit de l'adresse IP du client.

La longueur de cette zone est indiquée par MQ_CONN_NAME_LENGTH.

Dans le cas d'un client, il s'agit de l'adresse du partenaire du gestionnaire de files d'attente.

LongMCAUserIdLength (MQLONG)-entrée
Longueur de l'ID utilisateur MCA long.

Lorsque MCA se connecte au gestionnaire de files d'attente, cette zone est définie sur la longueur de l'ID utilisateur MCA long (ou sur zéro s'il n'existe pas d'identificateur de ce type).

Dans le cas d'un client, il s'agit de l'identificateur d'utilisateur long client.

LongRemoteUserIdLongueur (MQLONG)-entrée
Longueur de l'ID utilisateur éloigné long.

Lorsque MCA se connecte au gestionnaire de files d'attente, cette zone est définie sur la longueur de l'ID utilisateur distant long. Sinon, cette zone sera définie sur zéro

Dans le cas d'un client, définissez cette zone sur zéro.

LongMCAUserIdPtr (MQPTR)-entrée
Adresse de l'ID utilisateur MCA long.

Lorsque MCA se connecte au gestionnaire de files d'attente, cette zone est définie sur l'adresse de l'identificateur utilisateur MCA long (ou sur un pointeur NULL s'il n'existe pas d'identificateur de ce type).

Dans le cas d'un client, il s'agit de l'identificateur d'utilisateur long client.

LongRemoteUserIdPtr (MQPTR)-entrée
Adresse de l'ID utilisateur distant long.

Lorsque MCA se connecte au gestionnaire de files d'attente, cette zone est définie sur l'adresse de l'identificateur d'utilisateur distant long (ou sur un pointeur NULL s'il n'existe pas d'identificateur de ce type).

Dans le cas d'un client, définissez cette zone sur zéro.

ApplName (MQCHAR28)-entrée
Nom de l'application ou du composant qui a émis l'appel API WebSphere MQ .

Les règles de génération du ApplName sont les mêmes que pour la génération du nom par défaut d'une instruction MQPUT.

La valeur de cette zone est trouvée en demandant le nom du programme au système d'exploitation. Sa longueur est donnée par MQ_APPL_NAME_LENGTH.

ApplType (MQLONG)-entrée
Type d'application ou de composant qui a émis l'appel API WebSphere MQ .

La valeur est MQAT_DEFAULT pour la plateforme sur laquelle l'application est compilée ou correspond à l'une des valeurs MQAT_ * définies.

Le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

ProcessId (MQPID)-entrée
Identificateur de processus du système d'exploitation.

Le cas échéant, le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

ThreadId (MQTID)-entrée
Identificateur de l'unité d'exécution MQ . Il s'agit du même identificateur utilisé dans la trace MQ et dans les vidages FFST , mais il peut être différent de l'identificateur d'unité d'exécution du système d'exploitation.

Le cas échéant, le gestionnaire d'exit définit cette zone à l'entrée de chaque fonction d'exit.

ChannelName (MQCHAR)-entrée
Nom du canal, complété par des blancs, le cas échéant et connu.

S'il n'est pas applicable, cette zone est définie sur des caractères NULL.

Reserved1 (MQBYTE4)-entrée
Cette zone est réservée.

ChanneDefinition (PMQCD)-entrée
Un pointeur vers la définition de canal utilisée, le cas échéant et connue.

S'il n'est pas applicable, cette zone est définie sur des caractères NULL.

Notez que le pointeur n'est terminé que si la connexion est en cours de traitement pour le compte d'un canal WebSphere MQ et que cette définition de canal a été lue.

En particulier, la définition de canal n'est pas donnée sur le serveur lorsque le premier appel MQCONN est effectué pour le canal. En outre, si le pointeur est rempli, la structure (et les éventuelles sous-structures) pointée par le pointeur doivent être traitées comme étant en lecture seule ; toute mise à jour de la structure conduirait à des résultats imprévisibles et n'est pas supportée.

Dans le cas d'un client, les zones autres que celles dont la valeur est spécifiée pour un client contiennent des valeurs appropriées pour une application client.

Zone de chaîne d'exit et en-tête de zone de chaîne d'exit (MQACH)

Si nécessaire, une fonction d'exit peut acquérir du stockage pour une zone de chaîne d'exit et définir le paramètre ExitChainAreaPtr dans MQAXP pour qu'il pointe vers ce stockage.

Les exits (des fonctions d'exit identiques ou différentes) peuvent acquérir plusieurs zones de chaîne d'exit et les lier ensemble. Les zones de chaîne d'exit doivent uniquement être ajoutées ou supprimées de cette liste lorsqu'elles sont appelées à partir du gestionnaire d'exit. Cela permet de s'assurer qu'il n'y a pas de problèmes de sérialisation causés par des unités d'exécution différentes qui ajoutent ou suppriment des zones de la liste en même temps.

Une zone de chaîne d'exit doit commencer par une structure d'en-tête MQACH, dont la déclaration C est:

```
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 */
};
```

Les zones de l'en-tête de la zone de chaîne d'exit sont les suivantes:

StrucId (MQCHAR4)-entrée
Identificateur de structure de zone de chaîne d'exit, avec une valeur initiale, définie par MQACH_DEFAULT, de MQACH_STRUC_ID.

Pour les programmes C, la constante MQACH_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQACH_STRUC_ID, mais sous la forme d'un tableau de caractères au lieu d'une chaîne.

Version (MQLONG)-entrée
Numéro de version de la structure, comme suit:

MQACH_VERSION_1
Numéro de version de la structure du paramètre d'exit.

VERSION_MQACH_CURRENT_VERSION
Numéro de version en cours de la structure de contexte d'exit.

La valeur initiale de cette zone, définie par MQACH_DEFAULT, est `MQACH_CURRENT_VERSION`.

Remarque : Si vous introduisez une nouvelle version de cette structure, la présentation de la partie existante ne change pas. Les fonctions d'exit doivent vérifier que le numéro de version est égal ou supérieur à la version la plus basse contenant les zones que la fonction d'exit doit utiliser.

StrucLength (MQLONG)-entrée
Longueur de la structure MQACH. Les exits peuvent utiliser cette zone pour déterminer le début des données d'exit, en lui affectant la longueur de la structure créée par l'exit.

La valeur initiale de cette zone, définie par MQACH_DEFAULT, est MQACH_CURRENT_LENGTH.

ChainAreaLongueur (MQLONG)-entrée
Longueur de la zone de chaîne d'exit, définie sur la longueur totale de la zone de chaîne d'exit en cours, y compris l'en-tête MQACH.

La valeur initiale de cette zone, définie par MQACH_DEFAULT, est zéro.

ExitInfoNom (MQCHAR48)-entrée
Nom des informations d'exit.

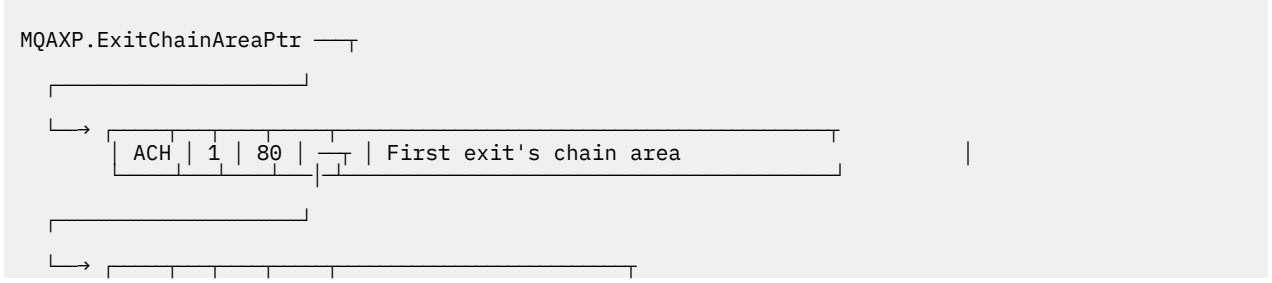
Lorsqu'un exit crée une structure MQACH, il doit initialiser cette zone avec son propre nom ExitInfo, de sorte que cette structure MQACH puisse être trouvée ultérieurement par une autre instance de cet exit ou par un exit associé.

La valeur initiale de cette zone, définie par MQACH_DEFAULT, est une chaîne de longueur zéro ({""}).

NextChainAreaPtr (PMQACH)-entrée
Un pointeur vers la zone de chaîne d'exit suivante avec une valeur initiale, définie par MQACH_DEFAULT, de pointeur null (NULL).

Les fonctions de sortie doivent libérer le stockage pour toutes les zones de chaîne de sortie qu'elles acquièrent et manipuler les pointeurs de chaîne pour supprimer leurs zones de chaîne de sortie de la liste.

Une zone de chaîne de sortie peut être construite comme suit:



```
      | ACH | 1 | 64 | ──┬ | Second exit's chain area  |
      └─────┴───┴────┴───┼┴──────────────────────────┘
                         │
  ┌──────────────────────┘
  │
  └──→ ... etc.
```

Constantes externes

Utilisez cette rubrique comme informations de référence pour les constantes externes disponibles pour l'API.

Les constantes externes suivantes sont disponibles pour les exits API:

MQXF_ * (identificateurs de fonction d'exit)

```
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_ * (raisons de l'exit)

```
MQXR_BEFORE                     1       X'00000001'
MQXR_AFTER                      2       X'00000002'
MQXR_CONNECTION                 3       X'00000003'
```

MQXE_ * (environnements)

```
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* _ * (constantes supplémentaires)

```
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* _ * (constantes null)

```
MQXPDA_NONE                    X'00...00' (48 nulls)
MQXPDA_NONE_ARRAY              '\0','\0',...,'\0','\0'
```

MQXCC_ * (codes achèvement)

```
MQXCC_FAILED                  -8
```

MQRC_ * (codes anomalie)

MQRC_API_EXIT_ERROR 2374 X'00000946'
Un appel de fonction d'exit a renvoyé un code de réponse non valide ou a échoué d'une manière ou d'une autre, et le gestionnaire de files d'attente ne peut pas déterminer l'action suivante à effectuer.

Examinez les zones ExitResponse et ExitResponse2 de MQAXP pour déterminer le code de réponse incorrect et modifiez l'exit pour qu'il renvoie un code de réponse valide.

MQRC_API_EXIT_INIT_ERROR 2375 X'00000947'
Le gestionnaire de files d'attente a rencontré une erreur lors de l'initialisation de l'environnement d'exécution pour une fonction d'exit d'API.

MQRC_API_EXIT_TERM_ERROR 2376 X'00000948'
Le gestionnaire de files d'attente a rencontré une erreur lors de la fermeture de l'environnement d'exécution pour une fonction d'exit d'API.

MQRC_EXIT_REASON_ERROR 2377 X'00000949'
La valeur de la zone ExitReason fournie sur un appel d'enregistrement de point d'entrée d'exit (MQXEP) est erronée.

Examinez la valeur de la zone ExitReason pour déterminer et corriger la valeur de la raison de sortie incorrecte.

MQRC_RESERVED_VALUE_ERROR 2378 X'0000094A'
La valeur de la zone Réservé est erronée.

Examinez la valeur de la zone Réservé pour déterminer et corriger la valeur Réservée.

Types de langage C

Cette rubrique fournit des informations sur les typedefs associés aux exits API disponibles en langage C.

Voici les définitions de type de langage C associées aux exits API:

```
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;
```

L'appel d'enregistrement de point d'entrée d'exit (MQXEP)

Utilisez ces informations pour en savoir plus sur MQXEP, l'appel de langage C MQXEP et le prototype de fonction C MQXEP.

Utilisez l'appel MQXEP pour:

1. Enregistrez les points d'appel d'exit d'API avant et après WebSphere MQ sur lesquels appeler les fonctions d'exit
2. Indiquez les points d'entrée de la fonction d'exit
3. Désenregistrer les points d'entrée de la fonction d'exit

Vous codez généralement les appels MQXEP dans la fonction d'exit MQ_INIT_EXIT, mais vous pouvez les spécifier dans n'importe quelle fonction d'exit suivante.

Si vous utilisez un appel MQXEP pour enregistrer une fonction d'exit déjà enregistrée, le deuxième appel MQXEP aboutit et remplace la fonction d'exit enregistrée.

Si vous utilisez un appel MQXEP pour enregistrer une fonction d'exit NULL, l'appel MQXEP aboutit et la fonction d'exit est désenregistrée.

Si des appels MQXEP sont utilisés pour enregistrer, désenregistrer et réenregistrer une fonction d'exit particulière pendant la durée de vie d'une demande de connexion, la fonction d'exit précédemment enregistrée est réactivée. Toute mémoire encore allouée et associée à cette instance de fonction d'exit est disponible pour être utilisée par les fonctions de l'exit. (Cette mémoire est généralement libérée lors de l'appel de la fonction d'exit d'arrêt.)

L'interface de MQXEP est la suivante:

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason)
```

où :

Hconfig (MQHCONFIG)-entrée
Descripteur de configuration représentant l'exit API qui inclut l'ensemble de fonctions en cours d'initialisation. Cette valeur est générée par le gestionnaire de files d'attente immédiatement avant l'appel de la fonction MQ_INIT_EXIT et est transmise dans MQAXP à chaque fonction d'exit d'API.

ExitReason (MQLONG)-entrée
Motif pour lequel le point d'entrée est enregistré, pour les raisons suivantes:

- Initialisation ou arrêt du niveau de connexion (MQXR_CONNECTION)
- Avant un appel API WebSphere MQ (MQXR_BEFORE)
- Après un appel API WebSphere MQ (MQXR_AFTER)

Fonction (MQLONG)-entrée
Identificateur de fonction, dont les valeurs valides sont les constantes MQXF_ * (voir «Constantes externes», à la page 1113).

EntryPoint (PMQFUNC)-entrée
Adresse du point d'entrée de la fonction d'exit à enregistrer. La valeur NULL indique que la fonction d'exit n'a pas été fournie ou qu'un enregistrement précédent de la fonction d'exit est en cours de désenregistrement.

ExitOpts(MQXEPO)
Les exits API peuvent spécifier des options qui contrôlent la manière dont les exits API sont enregistrés. Si un pointeur null est spécifié pour cette zone, les valeurs par défaut de la structure MQXEPO sont utilisées.

CompCode (MQLONG)-sortie
Le code achèvement, dont les valeurs valides sont les suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

Cause (MQLONG)-sortie
Code anomalie qui qualifie le code achèvement.

Si le code achèvement est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED:

ERREUR MQRC_HCONFIG_ERROR
(2280, X'8E8') Le descripteur de configuration fourni n'est pas valide. Utilisez le descripteur de configuration de MQAXP.

MQRC_EXIT_MOTIF-ERREUR
(2377, X' 949') La raison d'appel de la fonction d'exit fournie n'est pas valide ou n'est pas valide pour l'identificateur de fonction d'exit fourni.

Utilisez l'une des raisons d'appel de fonction d'exit valides (valeur MQXR_ *) ou utilisez une combinaison d'identificateur de fonction et de raison d'exit valide. (Voir Tableau 594, à la page 1116.)

Erreur de fonction MQRC_FUNCTION_ERROR
(2281, X'8E9') L'identificateur de fonction fourni n'est pas valide pour la raison de l'exit API. Le tableau suivant présente les combinaisons valides d'identificateurs de fonction et de ExitReasons.

Tableau 594. Combinaisons valides d'identificateurs de fonction et de ExitReasons

Function	ExitReason
MQXF_INIT MQXF_TERME	MQXR_CONNECTION

Tableau 594. Combinaisons valides d'identificateurs de fonction et de ExitReasons (suite)	
Function	**ExitReason**
MQXF_CONN MQXF_CONNX MQXF_DISQUE MQXF_OPEN MQXF_FERMETURE MQXF_PUT1 MQXF_PUT MQXF_GET MQXF_INQ MQXF_SET MQXF_DÉBUT MQXF_CMIT MQXF_RETOUR MQXF_STAT MQXF_CB MQXF_CTL MQXF_RAPPEL MQXF_SUB MQXF_SUBRQ	MQXR_AVANT MQXR_APRES
MQXF_DATA_CONV_ON_GET	MQXR_AVANT

MQRC_RESOURCE_PROBLEM
(2102, X'836') Une tentative d'enregistrement ou de désenregistrement d'une fonction d'exit a échoué en raison d'un problème de ressource.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Une tentative d'enregistrement ou de désenregistrement d'une fonction d'exit a échoué de manière inattendue.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Nom ExitProperties non valide.

MQRC_XEPO_ERREUR
(2507, X'09CB') La structure des options d'exit n'est pas valide.

Appel du langage C MQXEP

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason);
```

Déclaration pour la liste de paramètres:

```
    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 */
```

Prototype de fonction C MQXEP

```
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 */
```

Fonctions de sortie

Cette section fournit des informations générales pour vous aider lors de l'utilisation des appels de fonction et décrit comment appeler les fonctions d'exit individuelles.

Utilisez ces informations pour comprendre les règles générales applicables aux routines d'exit d'API, ainsi que pour configurer et nettoyer l'environnement d'exécution de l'exit.

Règles générales pour les routines d'exit d'API

Les règles générales suivantes s'appliquent lors de l'appel de routines d'exit d'API.

- Dans tous les cas, les fonctions d'exit d'API sont gérées avant la validation des paramètres d'appel d'API et avant les contrôles de sécurité (dans le cas de MQCONN, MQCONNX ou MQOPEN).
- Les valeurs des zones entrées dans et en sortie d'une routine d'exit sont les suivantes:
  - Lors de l'entrée dans une fonction d'exit d'API *avant* WebSphere MQ , la valeur d'une zone peut être définie par le programme d'application ou par un appel de fonction d'exit précédent.
  - Dans la sortie d'une fonction d'exit d'API *avant* WebSphere MQ , la valeur d'une zone peut être laissée inchangée ou définie sur une autre valeur par la fonction d'exit.
  - En entrée d'une fonction d'exit d'API *après* WebSphere MQ , la valeur d'une zone peut être la valeur définie par le gestionnaire de files d'attente après le traitement de l'appel API WebSphere MQ , ou peut être définie sur une valeur par un appel de fonction d'exit précédent dans la chaîne de fonctions d'exit.
  - Dans la sortie d'une fonction d'exit d'appel d'API *après* WebSphere MQ , la valeur d'une zone peut être laissée inchangée ou définie sur une autre valeur par la fonction d'exit.
- Les fonctions d'exit doivent communiquer avec le gestionnaire de files d'attente à l'aide des zones ExitResponse et ExitResponse2 .
- Les zones CompCode et Code anomalie communiquent avec l'application. Les fonctions de gestionnaire de files d'attente et d'exit peuvent définir les zones CompCode et Code anomalie.
- L'appel MQXEP renvoie de nouveaux codes anomalie aux fonctions d'exit qui appellent MQXEP. Toutefois, les fonctions de sortie peuvent convertir ces nouveaux codes raison en codes raison existants que les applications existantes et nouvelles peuvent comprendre.
- Chaque prototype de fonction d'exit possède des paramètres similaires à la fonction d'API avec un niveau supplémentaire d'adressage indirect, à l'exception de CompCode et de Reason.
- Les exits API peuvent émettre des appels MQI (à l'exception de MQDISC), mais ces appels MQI n'appellent pas eux-mêmes les exits API.

Notez que, que l'application se trouve sur un serveur ou un client, vous ne pouvez pas prévoir le séquencement des appels d'exit d'API. Un appel BEF0RE d'exit API peut ne pas être immédiatement suivi d'un appel AFTER .

L'appel BEF0RE peut être suivi d'un autre appel BEF0RE . Exemple :

AVANT MQCTL
AVANT le rappel
AVANT MQPUT
APRES MQPUT
Rappel APRES
APRÈS MQCTL

ou

AVANT XAOPEN
AVANT MQCONNX
APRES MQCONNX
APRÈS XAOPEN

Sur le client, il existe un exit qui peut modifier le comportement de l'appel MQCONN ou MQCONNX, appelé exit `PreConnect` . L'exit `PreConnect` peut modifier n'importe quel paramètre de l'appel MQCONN ou MQCONNX, y compris le nom du gestionnaire de files d'attente. Le client appelle d'abord cet exit, puis appelle l'appel MQCONN ou MQCONNX. Notez que seul l'appel MQCONN ou MQCONNX initial appelle l'exit API ; les appels de reconnexion ultérieurs n'ont aucun effet.

Environnement d'exécution

En général, toutes les erreurs des fonctions d'exit sont communiquées au gestionnaire d'exit à l'aide des zones ExitResponse et ExitResponse2 de MQAXP.

Ces erreurs sont à leur tour converties en valeurs MQCC_ * et MQRC_ * et communiquées à l'application dans les zones CompCode et Motif. Toutefois, toutes les erreurs rencontrées dans la logique du gestionnaire d'exit sont communiquées à l'application sous forme de valeurs MQCC_ * et MQRC_ * dans les zones CompCode et Motif.

Si une fonction MQ_TERM_EXIT renvoie une erreur:

- L'appel MQDISC a déjà eu lieu
- Il n'y a pas d'autre possibilité de piloter la *après la fonction d'exit* MQ_TERM_EXIT (et donc d'effectuer un nettoyage de l'environnement d'exécution d'exit)
- Le nettoyage de l'environnement d'exécution de sortie n'est *pas* effectué

L'exit ne peut pas être déchargé car il est peut-être encore en cours d'utilisation. En outre, les autres exits enregistrés plus bas dans la chaîne d'exit pour laquelle l'exit *avant* a abouti seront gérés dans l'ordre inverse.

Configuration de l'environnement d'exécution de l'exit

Lors du traitement d'un appel MQCONN ou MQCONNX explicite, la logique de traitement de l'exit configure l'environnement d'exécution de l'exit avant d'appeler la fonction d'initialisation de l'exit (MQ_INIT_EXIT). La configuration de l'environnement d'exécution d'exit implique le chargement de l'exit, l'acquisition de la mémoire et l'initialisation des structures de paramètres d'exit. Le descripteur de configuration d'exit est également alloué.

Si des erreurs se produisent au cours de cette phase, l'appel MQCONN ou MQCONNX échoue avec CompCode MQCC_FAILED et l'un des codes anomalie suivants:

MQRC_API_EXIT_LOAD_ERROR
Une tentative de chargement d'un module d'exit API a échoué.

MQRC_API_EXIT_NOT_FOUND
Une fonction d'exit d'API est introuvable dans le module d'exit d'API.

MQRC_STORAGE_NOT_AVAILABLE
Une tentative d'initialisation de l'environnement d'exécution pour une fonction d'exit d'API a échoué car la mémoire disponible était insuffisante.

MQRC_API_EXIT_INIT_ERROR
Une erreur s'est produite lors de l'initialisation de l'environnement d'exécution pour une fonction d'exit d'API.

Nettoyage de l'environnement d'exécution de l'exit

Lors du traitement d'un appel MQDISC explicite ou d'une demande de déconnexion implicite suite à l'arrêt d'une application, la logique de traitement de l'exit peut avoir besoin de nettoyer l'environnement

d'exécution de l'exit après avoir appelé la fonction d'arrêt de l'exit (MQ_TERM_EXIT), si elle est enregistrée.

Le nettoyage de l'environnement d'exécution d'exit implique la libération de la mémoire pour les structures de paramètres d'exit, éventuellement la suppression des modules précédemment chargés en mémoire.

Si des erreurs se produisent au cours de cette phase, un appel MQDISC explicite échoue avec CompCode MQCC_FAILED et le code anomalie suivant (les erreurs ne sont pas mises en évidence sur les demandes de déconnexion implicites):

MQRC_API_EXIT_TERM_ERREUR
Une erreur s'est produite lors de la fermeture de l'environnement d'exécution pour une fonction d'exit d'API. L'exit ne doit *pas* renvoyer d'échec à partir du MQDISC avant ou après les appels de la fonction d'exit d'API MQ_TERM*.

Exits d'API sur les clients

Un client utilise l'exit PreConnect pour modifier le comportement des appels MQCONN et MQCONNX et ne prend pas en charge les propriétés d'exit d'API.

Exit PreConnect

Sur un client, l'exit PreConnect peut être utilisé pour rechercher la définition de canal à partir d'un référentiel central, tel qu'un serveur LDAP.

L'exit PreConnect peut également modifier n'importe quel paramètre, ou tous les paramètres, sur un appel MQCONN ou MQCONNX lui-même, par exemple, le nom du gestionnaire de files d'attente.

Dans le cas des applications client, l'exit PreConnect doit être appelé avant l'exit API car l'exit API MQCONN ou MQCONNX est appelé uniquement une fois que le nom du gestionnaire de files d'attente est connu et que ce nom peut être modifié par l'exit PreConnect .

Notez que seul l'appel MQCONN ou MQCONNX initial appelle l'exit.

Propriétés de l'exit API

Sur un serveur, les exits API peuvent enregistrer une structure MQXEPO lors de l'initialisation. La structure MQXEPO contient la zone ExitProperties qui détaille le groupe de propriétés qui intéresse l'exit. Cela a pour effet de générer un descripteur de propriété de message distinct que l'exit peut manipuler séparément de tout descripteur de propriété de message d'application.

Sur un client, les propriétés d'exit d'API ne sont pas prises en charge. Si une tentative d'enregistrement d'un nom de groupe de propriétés est effectuée sur un client, la fonction échoue avec le code anomalie MQRC_EXIT_PROPS_NOT_SUPPORTED.

Annulation-MQ_BACK_EXIT

MQ_BACK_EXIT fournit une fonction d'exit d'annulation permettant d'exécuter *avant* et *après* le traitement d'annulation. Utilisez l'identificateur de fonction MQXF_BACK avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel d'annulation *avant* et *après* .

L'interface de cette fonction est:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Début-MQ_BEGIN_EXIT

MQ_BEGIN_EXIT fournit une fonction d'exit de début permettant d'exécuter *avant* et *après* le traitement de l'appel MQBEGIN. Utilisez l'identificateur de fonction MQXF_BEGIN avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQBEGIN.

L'interface de cette fonction est:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

Options pBegin(PMQBO)-entrée / sortie
Pointeur vers les options de début.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Rappel-MQ_CALLBACK_EXIT

MQ_CALLBACK_EXIT fournit une fonction d'exit permettant d'exécuter *avant* et *après* le traitement du rappel. Utilisez l'identificateur de fonction MQXF_CALLBACK avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel de rappel *avant* et *après* .

L'interface de cette fonction est:

```
MQ_CALLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts,
    &pBuffer, &pMQCBContext)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure de paramètre d'exit

ExitContext (MQAXC)-entrée/sortie
Structure de contexte de sortie

Hconn (MQHCONN)-entrée/sortie
Descripteur de connexion

pMsgDescription
Descripteur de message

pGetMsgOpts
Options qui contrôlent l'action de MQGET

pBuffer
Zone devant contenir les données de message

pMQCBContext
Données contextuelles pour le rappel

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts, &pBuffer,
 &pContext);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Notes d'utilisation

1. L'exit de rappel est appelé avant l'appel du consommateur et après la fin de la fonction de consommateur du consommateur. Bien que les structures MQMD et MQGMO puissent être modifiées, la modification des valeurs dans l'exit avant ne reconduit pas l'extraction d'un message de la file d'attente car le message a déjà été supprimé de la file d'attente pour être distribué à la fonction de destinataire

Gestion des fonctions de rappel-MQ_CB_EXIT

MQ_CB_EXIT fournit une fonction d'exit permettant d'exécuter *avant* et *après* l'appel MQCB. Utilisez l'identificateur de fonction MQXF_CB avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQCB.

L'interface de cette fonction est:

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &pCallbackDesc,
            &Hobj, &pMsgDesc, &pGetMsgOpts, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure de paramètre d'exit

ExitContext (MQAXC)-entrée/sortie
Structure de contexte de sortie

Hconn (MQHCONN)-entrée/sortie
Descripteur de connexion

Opération (MQLONG)-entrée / sortie
Valeur de l'opération

pCallbackDesc (PMQCBD)-entrée / sortie
Descripteur de rappel

Hobj (MQHOBJ)-entrée/sortie
Descripteur d'objet

pMsgDescription (PMQMD)-entrée/sortie
Descripteur de message

pGetMsgOpts (PMQGMO)-entrée/sortie
Options qui contrôlent l'action de MQCB

CompCode (MQLONG)-entrée/sortie
Code de fin d'exécution

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant CompCode

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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.
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
        MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &Hobj, &pMsgDesc,
                    &pGetMsgOpts, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Fermer-MQ_CLOSE_EXIT

MQ_CLOSE_EXIT fournit une fonction d'exit de fermeture permettant d'exécuter *avant* et *après* le traitement des appels MQCLOSE. Utilisez l'identificateur de fonction MQXF_CLOSE avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel MQCLOSE *avant* et *après* .

L'interface de cette fonction est:

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHobj,
               &Options, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pHobj (PMQHOBJ)-entrée
Pointeur vers le descripteur d'objet.

Options (MQLONG)-entrée / sortie
Options de fermeture.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext,&Hconn, &pHobj, &Options,
               &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Validation-MQ_CMIT_EXIT

MQ_CMIT_EXIT fournit une fonction d'exit de validation permettant d'exécuter *avant* et *après* le traitement de validation. Utilisez l'identificateur de fonction MQXF_CMIT avec les raisons d'exit

MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel de validation *avant* et *après* .

Si une opération de validation échoue et que la transaction est annulée, l'appel MQCMIT échoue avec MQCC_WARNING et MQRC_BACKED_OUT. Ces codes retour et anomalie sont transmis à toutes les fonctions d'exit *après* MQCMIT pour indiquer aux fonctions d'exit que l'unité de travail a été annulée.

L'interface de cette fonction est:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext,&Hconn, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Notes d'utilisation

1. L'interface de fonction MQ_GET_EXIT décrite ici est utilisée à la fois pour la fonction d'exit MQXF_GET et pour la fonction d'exit «MQXF_DATA_CONV_ON_GET», à la page 1133 .

   Des points d'entrée distincts sont définis pour ces deux fonctions d'exit. Par conséquent, pour intercepter *les deux* , l'appel MQXEP doit être utilisé deux fois ; pour cet appel, utilisez l'identificateur de fonction MQXF_GET.

   L'interface MQ_GET_EXIT étant la même pour MQXF_GET et MQXF_DATA_CONV_ON_GET, une seule fonction d'exit peut être utilisée pour les deux ; la zone *Function* de la structure MQAXP indique quelle fonction d'exit a été appelée. L'appel MQXEP peut également être utilisé pour enregistrer des fonctions d'exit différentes pour les deux cas.

Connexion et extension de connexion-MQ_CONNX_EXIT

MQ_CONNX_EXIT fournit:

- Fonction d'exit de connexion permettant d'exécuter *avant* et *après* le traitement MQCONN
- Fonction d'exit d'extension de connexion permettant d'exécuter *avant* et *après le traitement de* MQCONNX

La même interface, décrite ici, est appelée pour les fonctions d'exit d'appel MQCONN et MQCONNX.

Lorsque l'agent MCA répond à une connexion client entrante, il peut se connecter et effectuer un certain nombre d'appels API WebSphere MQ avant que l'état du client ne soit complètement connu. Ces appels API appellent les fonctions d'exit API avec MQAXC en fonction du programme MCA lui-même (par exemple, dans les zones UserId et ConnectionName de MQAXC).

Lorsque l'agent MCA répond aux appels API client entrants suivants, la structure MQAXC est basée sur le client entrant, en définissant les zones UserId et ConnectionName de manière appropriée.

Le nom du gestionnaire de files d'attente défini par l'application sur un appel MQCONN ou MQCONNX est transmis à l'appel de connexion sous-jacent. Toute tentative de modification du nom du gestionnaire de files d'attente par un *avant* MQ_CONNX_EXIT n'a aucun effet.

Utilisez les identificateurs de fonction MQXF_CONN et MQXF_CONNX avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQCONN et MQCONNX.

Un exit MQ_CONNX_EXIT appelé pour la raison suivante: MQXR_BEFORE *ne doit* émettre aucun appel d'API WebSphere MQ , car l'environnement approprié n'a pas été configuré pour le moment.

Un MQ_CONNX_EXIT ne peut pas appeler MQDISC à partir d'un appel d'exit API pour la connexion pour laquelle il est appelé. Cette restriction s'applique aux exits API client et serveur.

L'interface de MQCONN et de MQCONNX est identique:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOpts,
               &pHconn, &CompCode, &Reason);
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

pQMgrNom (PMQCHAR)-entrée
Pointeur vers le nom du gestionnaire de files d'attente fourni dans l'appel MQCONNX. L'exit ne doit pas modifier ce nom dans l'appel MQCONN ou MQCONNX.

pConnectOpts (PMQCNO)-entrée/sortie
Pointeur vers les options qui contrôlent l'action de l'appel MQCONNX.

Voir «MQCNO-Options de connexion», à la page 298 pour plus de détails.

Pour la fonction d'exit MQXF_CONN, pConnectOpts pointe vers la structure d'options de connexion par défaut (MQCNO_DEFAULT).

pHconn (PMQHCONN)-entrée
Pointeur vers le descripteur de connexion.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel)

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOps,
               &pHconn, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Notes d'utilisation

1. L'interface de fonction MQ_CONNX_EXIT décrite ici est utilisée à la fois pour l'appel MQCONN et pour l'appel MQCONNX. Toutefois, des points d'entrée distincts sont définis pour ces deux appels. Pour intercepter *les deux* appels, l'appel MQXEP doit être utilisé au moins deux fois-une fois avec l'identificateur de fonction MQXF_CONN, et de nouveau avec MQXF_CONNX.

   L'interface MQ_CONNX_EXIT étant la même pour MQCONN et MQCONNX, une seule fonction d'exit peut être utilisée pour les deux appels ; la zone *Function* de la structure MQAXP indique l'appel en cours. L'appel MQXEP peut également être utilisé pour enregistrer des fonctions d'exit différentes pour les deux appels.

2. Lorsqu'un agent MCA répond à une connexion client entrante, il peut émettre un certain nombre d'appels MQ avant que l'état du client ne soit complètement connu. Ces appels MQ entraînent l'appel des fonctions d'exit API avec la structure MQAXC contenant des données relatives à l'agent MCA et non au client (par exemple, l'identificateur utilisateur et le nom de connexion). Toutefois, une fois que l'état du client est entièrement connu, les appels MQ ultérieurs entraînent l'appel des fonctions d'exit API avec les données client appropriées dans la structure MQAXC.
3. Toutes les fonctions d'exit MQXR_BEFORE sont appelées avant toute validation de paramètre effectuée par le gestionnaire de files d'attente. Il se peut donc que les paramètres ne soient pas valides (y compris les pointeurs non valides pour les adresses des paramètres).

   La fonction MQ_CONNX_EXIT est appelée avant toute vérification d'autorisation effectuée par le gestionnaire de files d'attente.
4. La fonction d'exit ne doit pas modifier le nom du gestionnaire de files d'attente spécifié dans l'appel MQCONN ou MQCONNX. Si le nom est modifié par la fonction d'exit, les résultats ne sont pas définis.
5. Une fonction d'exit MQXR_BEFORE pour MQ_CONNX_EXIT ne peut pas émettre d'appels MQ autres que MQXEP.

Rappel de contrôle-MQ_CTL_EXIT

MQ_CTL_EXIT fournit une fonction d'exit de demande d'abonnement pour exécuter *avant* et *après le traitement du rappel de contrôle* . Utilisez l'identificateur de fonction MQXF_CTL avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel de rappel de contrôle.

L'interface de cette fonction est:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason)
```

où les paramètres sont :

Hconn (MQHCONN)-entrée/sortie
Descripteur de connexion.

Entrée / sortie d'opération (MQLONG)
Opération en cours de traitement sur le rappel défini pour le descripteur d'objet spécifié

Entrée / sortie ControlOpts (MQCTLO)
Options qui contrôlent l'action de MQCTL

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Déconnexion-MQ_DISC_EXIT

MQ_DISC_EXIT fournit une fonction d'exit de déconnexion permettant d'exécuter *avant* et *après le traitement de l'exit MQDISC* . Utilisez l'identificateur de fonction MQXF_DISC avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQDISC.

L'interface de cette fonction est

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
              &CompCode, &Reason);
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

pHconn (PMQHCONN)-entrée
Pointeur vers le descripteur de connexion.

Pour l'appel MQDISC précédent, la valeur de cette zone est l'une des suivantes:

- Descripteur de connexion renvoyé lors de l'appel MQCONN ou MQCONNX
- Zéro, pour les environnements dans lesquels un adaptateur spécifique à l'environnement s'est connecté au gestionnaire de files d'attente
- Valeur définie par un appel de fonction d'exit précédent

Pour l'appel MQDISC après, la valeur de cette zone est zéro ou une valeur définie par un appel de fonction d'exit précédent.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP           ExitParms;      /* Exit parameter structure */
MQAXC           ExitContext;    /* Exit context structure */
PMQHCONN        pHconn;         /* Ptr to Connection handle */
MQLONG          CompCode;       /* Completion code */
MQLONG          Reason;         /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
              &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Get-MQ_GET_EXIT

MQ_GET_EXIT fournit une fonction d'exit d'extraction permettant d'exécuter *avant* et *après* le traitement de l'appel MQGET.

Il existe deux identificateurs de fonction:

1. Utilisez MQXF_GET avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel MQGET *avant* et *après* .
2. Pour plus d'informations sur l'utilisation de l'identificateur de fonction MQXF_DATA_CONV_ON_GET, voir «MQXF_DATA_CONV_ON_GET», à la page 1133 .

L'interface de cette fonction est:

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

Hobj (MQHOBJ)-entrée/sortie
Descripteur d'objet.

pMsgDescription (PMQMD)-entrée/sortie
Pointeur vers le descripteur de message.

pGetMsgOpts (PMQGMO)-entrée/sortie
Pointeur pour obtenir les options de message.

BufferLength (MQLONG)-entrée/sortie
Longueur de la mémoire tampon de messages.

pBuffer (PMQBYTE)-entrée/sortie
Pointeur vers la mémoire tampon de messages.

pDataLength (PMQLONG)-entrée/sortie
Pointeur vers la zone de longueur de données.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Notes d'utilisation

1. L'interface de fonction MQ_GET_EXIT décrite ici est utilisée à la fois pour la fonction d'exit MQXF_GET et pour la fonction d'exit «MQXF_DATA_CONV_ON_GET», à la page 1133 .

   Des points d'entrée distincts sont définis pour ces deux fonctions d'exit. Par conséquent, pour intercepter *les deux* , l'appel MQXEP doit être utilisé deux fois ; pour cet appel, utilisez l'identificateur de fonction MQXF_GET.

   L'interface MQ_GET_EXIT étant la même pour MQXF_GET et MQXF_DATA_CONV_ON_GET, une seule fonction d'exit peut être utilisée pour les deux ; la zone *Function* de la structure MQAXP indique quelle fonction d'exit a été appelée. L'appel MQXEP peut également être utilisé pour enregistrer des fonctions d'exit différentes pour les deux cas.

MQXF_DATA_CONV_ON_GET

Voir MQ_GET_EXIT pour plus d'informations sur l'interface de cet appel et un exemple de déclaration en langage C.

Notes d'utilisation

S'il est enregistré, ce point d'entrée est appelé lorsque des messages arrivent à l'application mais avant qu'une conversion de données ne soit effectuée. Cela peut être utile si l'exit API doit effectuer un traitement, tel que le déchiffrement ou la décompression, avant que le message ne soit transmis à la conversion de données. L'exit peut, si nécessaire, provoquer le contournement de la conversion de données en renvoyant la fonction MQXCC_SUPPRESS_FUNCTION; pour plus d'informations, voir la structure MQAXP .

L'enregistrement de ce point d'entrée sur un client a pour effet d'entraîner l'exécution locale de la conversion de données sur la machine client. Pour un fonctionnement correct, il peut donc être nécessaire d'installer les exits de conversion d'application sur le client. Notez que MQXF_DATA_CONV_ON_GET est également utilisé pour la consommation asynchrone.

Lors de l'utilisation de l' appel MQ_GET_EXIT, utilisez MQXF_DATA_CONV_ON_GET, avec la raison d'exit MQXR_BEFORE, pour enregistrer une fonction d'exit de conversion de données *before* MQGET.

Il n'existe pas de fonction d'exit MQXR_AFTER pour MQXF_DATA_CONV_ON_GET ; la fonction d'exit MQXR_AFTER pour MQXF_GET fournit la fonction requise pour le traitement des exits après la conversion des données.

Des points d'entrée distincts sont définis pour l' appel MQ_GET_EXIT. Par conséquent, pour intercepter *les deux* fonctions d'exit, l'appel MQXEP doit être utilisé deux fois ; pour cet appel, utilisez l'identificateur de fonction MQXF_DATA_CONV_ON_GET.

L'interface MQ_GET_EXIT étant la même pour MQXF_GET et MQXF_DATA_CONV_ON_GET, une seule fonction d'exit peut être utilisée pour les deux ; la zone *Function* de la structure MQAXP indique quelle fonction d'exit a été appelée. L'appel MQXEP peut également être utilisé pour enregistrer des fonctions d'exit différentes pour les deux cas.

Initialisation-MQ_INIT_EXIT

MQ_INIT_EXIT fournit une initialisation de niveau de connexion, indiquée en définissant ExitReason dans MQAXP sur MQXR_CONNECTION.

Lors de l'initialisation, notez ce qui suit:

- La fonction MQ_INIT_EXIT appelle MQXEP pour enregistrer les instructions d'API WebSphere MQ et les points ENTRY et EXIT qui l'intéressent.
- Les exits n'ont pas besoin d'intercepter toutes les instructions d'API WebSphere MQ . Les fonctions d'exit sont appelées uniquement si un intérêt a été enregistré.
- La mémoire qui doit être utilisée par l'exit peut être acquise lors de son initialisation.
- Si un appel à cette fonction échoue, l'appel MQCONN ou MQCONNX qui l'a appelé échoue également avec un CompCode et une raison qui dépendent de la valeur de la zone ExitResponse dans MQAXP.

- Un exit MQ_INIT_EXIT ne doit pas émettre d'appels d'API WebSphere MQ , car l'environnement approprié n'a pas été configuré pour le moment.
- Si un exit MQ_INIT_EXIT échoue avec MQXCC_FAILED, le gestionnaire de files d'attente renvoie l'appel MQCONN ou MQCONNX qui l'a appelé avec MQCC_FAILED et MQRC_API_EXIT_ERROR.
- Si le gestionnaire de files d'attente rencontre une erreur lors de l'initialisation de l'environnement d'exécution de la fonction d'exit d'API avant d'appeler le premier exit MQ_INIT_EXIT, le gestionnaire de files d'attente renvoie l'appel MQCONN ou MQCONNX qui a appelé MQ_INIT_EXIT avec MQCC_FAILED et MQRC_API_EXIT_INIT_ERROR.

L'interface de MQ_INIT_EXIT est la suivante:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

CompCode (MQLONG)-entrée/sortie
Pointeur vers le code achèvement, dont les valeurs admises sont:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Pointeur vers le code raison qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Le CompCode et le motif renvoyés à l'application dépendent de la valeur de la zone ExitResponse dans MQAXP.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        MQAXP           ExitParms;      /* Exit parameter structure */
        MQAXC           ExitContext;    /* Exit context structure */
        MQLONG          CompCode;       /* Completion code */
        MQLONG          Reason;         /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Notes d'utilisation

1. La fonction MQ_INIT_EXIT peut émettre l'appel MQXEP pour enregistrer les adresses des fonctions d'exit pour les appels MQ à intercepter. Il n'est pas nécessaire d'intercepter tous les appels MQ , ni d'intercepter les deux appels MQXR_BEFORE et MQXR_AFTER. Par exemple, une suite d'exit peut choisir d'intercepter uniquement l'appel MQXR_BEFORE de MQPUT.
2. Le stockage qui doit être utilisé par les fonctions d'exit dans la suite d'exit peut être acquis par la fonction MQ_INIT_EXIT. Les fonctions d'exit peuvent également acquérir du stockage lorsqu'elles sont appelées, selon les besoins. Toutefois, toute la mémoire doit être libérée avant l'arrêt de la suite d'exit ; la fonction MQ_TERM_EXIT peut libérer la mémoire ou une fonction d'exit appelée précédemment.
3. Si MQ_INIT_EXIT renvoie MQXCC_FAILED dans la zone *ExitResponse* de MQAXP ou échoue d'une autre manière, l'appel MQCONN ou MQCONNX qui a provoqué l'appel de MQ_INIT_EXIT échoue également, avec les paramètres *CompCode* et *Reason* définis sur les valeurs appropriées.
4. Une fonction MQ_INIT_EXIT ne peut pas émettre d'appels MQ autres que MQXEP.

Interroger-MQ_INQ_EXIT

MQ_INQ_EXIT fournit une fonction d'exit d'interrogation permettant d'exécuter *avant* et *après le traitement de l'appel MQINQ* . Utilisez l'identificateur de fonction MQXF_INQ avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel MQINQ *avant* et *après* .

L'interface de cette fonction est:

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

Hobj (MQHOBJ)-entrée
Descripteur d'objet.

SelectorCount (MQLONG)-entrée
Nombre de sélecteurs

pSelectors (PMQLONG)-entrée/sortie
Pointeur vers un tableau de valeurs de sélecteur.

IntAttrCount (MQLONG)-entrée
Nombre d'attributs de type entier.

Attributs pInt(PMQLONG)-entrée/sortie
Pointeur vers un tableau de valeurs d'attribut entières.

CharAttrLongueur (MQLONG)-entrée/sortie
Longueur du tableau d'attributs de caractères.

pCharAttrs (PMQCHAR)-entrée / sortie
Pointeur vers un tableau d'attributs de caractères.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Ouvrir-MQ_OPEN_EXIT

MQ_OPEN_EXIT fournit une fonction d'exit ouvert permettant d'exécuter *avant* et *après* le traitement de l'appel MQOPEN. Utilisez l'identificateur de fonction MQXF_OPEN avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel MQOPEN *avant* et *après* .

L'interface de cette fonction est

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pObjDescription (PMQOD)-entrée/sortie
Pointeur vers le descripteur d'objet.

Options (MQLONG)-entrée / sortie
Options d'ouverture.

pHobj (PMQHOBJ)-entrée
Pointeur vers le descripteur d'objet.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Insertion-MQ_PUT_EXIT

MQ_PUT_EXIT fournit une fonction d'exit d'insertion pour exécuter *avant* et *après* le traitement de l'appel MQPUT. Utilisez l'identificateur de fonction MQXF_PUT avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel MQPUT *avant* et *après* .

L'interface de cette fonction est:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

Hobj (MQHOBJ)-entrée/sortie
Descripteur d'objet.

pMsgDescription (PMQMD)-entrée/sortie
Pointeur vers le descripteur de message.

pPutMsgOpts (PMQPMO)-entrée/sortie
Pointeur permettant d'insérer des options de message.

BufferLength (MQLONG)-entrée/sortie
Longueur de la mémoire tampon de messages.

pBuffer (PMQBYTE)-entrée/sortie
Pointeur vers la mémoire tampon de messages.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Notes d'utilisation

- Les messages de rapport générés par le gestionnaire de files d'attente ignorent le traitement normal des appels. Par conséquent, ces messages ne peuvent pas être interceptés par la fonction MQ_PUT_EXIT ou la fonction MQPUT1 . Toutefois, les messages de rapport générés par l'agent MCA sont traités normalement et peuvent donc être interceptés par la fonction MQ_PUT_EXIT ou la fonction MQ_PUT1_EXIT . Pour être sûr d'intercepter tous les messages de rapport générés par l'agent MCA, vous devez utiliser à la fois MQ_PUT_EXIT et MQ_PUT1_EXIT .

Put1 - MQ_PUT1_EXIT

MQ_PUT1_EXIT fournit une fonction d'exit *put one message only* pour exécuter *avant* et *après le traitement de l'appel* MQPUT1 . Utilisez l'identificateur de fonction MQXF_PUT1 avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel MQPUT1 .

L'interface de cette fonction est:

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
              &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pObjDescription (PMQOD)-entrée/sortie
Pointeur vers le descripteur d'objet.

pMsgDescription (PMQMD)-entrée/sortie
Pointeur vers le descripteur de message.

pPutMsgOpts (PMQPMO)-entrée/sortie
Pointeur permettant d'insérer des options de message.

BufferLength (MQLONG)-entrée/sortie
Longueur de la mémoire tampon de messages.

pBuffer (PMQBYTE)-entrée/sortie
Pointeur vers la mémoire tampon de messages.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
              &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Définir-MQ_SET_EXIT

MQ_SET_EXIT fournit une fonction d'exit de définition permettant d'exécuter *avant* et *après le traitement des appels MQSET* . Utilisez l'identificateur de fonction MQXF_SET avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel MQSET *avant* et *après* .

L'interface de cette fonction est:

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttr, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

Hobj (MQHOBJ)-entrée
Descripteur d'objet.

SelectorCount (MQLONG)-entrée
Nombre de sélecteurs

pSelectors (PMQLONG)-entrée/sortie
Pointeur vers un tableau de valeurs de sélecteur.

IntAttrCount (MQLONG)-entrée
Nombre d'attributs de type entier.

Attributs pInt(PMQLONG)-entrée/sortie
Pointeur vers un tableau de valeurs d'attribut entières.

CharAttrLongueur (MQLONG)-entrée/sortie
Longueur du tableau d'attributs de caractères.

pCharAttrs (PMQCHAR)-entrée / sortie
Pointeur vers les valeurs d'attribut de caractère.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Statut-MQ_STAT_EXIT

MQ_STAT_EXIT fournit une fonction d'exit de statut permettant d'exécuter *avant* et *après* le traitement des appels MQSTAT. Utilisez l'identificateur de fonction MQXF_STAT avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel MQSTAT *avant* et *après* .

L'interface de cette fonction est:

```
MQ_STAT_EXIT (&ExitParms, &ExitContext, &Hconn, &Type, &pStatus
              &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

Type (MQLONG)-entrée
Type d'informations de statut à extraire.

pStatus (PMQSTS)-sortie
Pointeur vers la mémoire tampon d'état.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Arrêt-MQ_TERM_EXIT

MQ_TERM_EXIT fournit un arrêt au niveau de la connexion, enregistré avec l'identificateur de fonction MQXF_TERM et ExitReason MQXR_CONNECTION. S'il est enregistré, MQ_TERM_EXIT est appelé une fois pour chaque demande de déconnexion.

Dans le cadre de l'arrêt, le stockage qui n'est plus requis par l'exit peut être libéré et tout nettoyage requis peut être effectué.

Si un MQ_TERM_EXIT échoue avec MQXCC_FAILED, le gestionnaire de files d'attente est renvoyé par le MQDISC qui l'a appelé avec MQCC_FAILED et MQRC_API_EXIT_ERROR.

Si le gestionnaire de files d'attente rencontre une erreur lors de l'arrêt de l'environnement d'exécution de la fonction d'exit d'API après avoir appelé le dernier MQ_TERM_EXIT, le gestionnaire de files d'attente renvoie l'appel MQDISC qui a appelé MQ_TERM_EXIT avec MQCC_FAILED et MQRC_API_EXIT_TERM_ERROR

L'interface de cette fonction est:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Le CompCode et le motif renvoyés à l'application dépendent de la valeur de la zone ExitResponse dans MQAXP.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
MQAXP           ExitParms;      /* Exit parameter structure */
MQAXC           ExitContext;    /* Exit context structure */
MQLONG          CompCode;       /* Completion code */
MQLONG          Reason;         /* Reason code */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Notes d'utilisation

1. La fonction MQ_TERM_EXIT est facultative. Il n'est pas nécessaire qu'une suite d'exit enregistre un exit d'arrêt s'il n'y a pas de traitement d'arrêt à effectuer.

   Si des fonctions appartenant à la suite d'exit acquièrent des ressources lors de la connexion, une fonction MQ_TERM_EXIT est un point pratique pour libérer ces ressources, par exemple pour libérer de la mémoire obtenue de manière dynamique.
2. Si une fonction MQ_TERM_EXIT est enregistrée lors de l'émission de l'appel MQDISC, la fonction d'exit est appelée une fois que toutes les fonctions d'exit MQDISC ont été appelées.
3. Si MQ_TERM_EXIT renvoie MQXCC_FAILED dans la zone *ExitResponse* de MQAXP, ou échoue d'une autre manière, l'appel MQDISC qui a provoqué l'appel de MQ_TERM_EXIT échoue également, avec les paramètres *CompCode* et *Reason* définis sur les valeurs appropriées.

Enregistrement de l'abonnement-MQ_SUB_EXIT

MQ_SUB_EXIT fournit une fonction d'exit permettant d'exécuter *avant* et *après le traitement de réenregistrement d'abonnement* . Utilisez l'identificateur de fonction MQXF_SUB avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel d'enregistrement d'abonnement.

L'interface de cette fonction est:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub, &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée/sortie
Descripteur de connexion.

pSubDesc-input/output
Tableau de sélecteurs d'attribut.

pHobj -entrée/sortie
Descripteur d'objet

Entrée / sortie pHsub (MQHOBJ)
Descripteur d'abonnement

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub,
 &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Demande d'abonnement-MQ_SUBRQ_EXIT

MQ_SUBRQ_EXIT fournit une fonction d'exit de demande d'abonnement permettant d'exécuter *avant* et *après* le traitement de la demande d'abonnement. Utilisez l'identificateur de fonction MQXF_SUBRQ avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer *avant* et *après* les fonctions d'exit d'appel de demande d'abonnement.

L'interface de cette fonction est:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
    &CompCode, &Reason)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée/sortie
Descripteur de connexion.

Entrée / sortie pHsub (MQHOBJ)
Descripteur d'abonnement

Entrée / sortie d'action (MQLONG)
Action

Entrée / sortie pSubRqOpts (MQSRO)

CompCode (MQLONG)-entrée/sortie
Code achèvement, valeurs valides pour lesquelles:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel

Motif (MQLONG)-entrée / sortie
Code anomalie qualifiant le code achèvement.

Si le code achèvement est MQCC_OK, la seule valeur valide est:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si le code achèvement est MQCC_FAILED ou MQCC_WARNING, la fonction d'exit peut définir la zone de code anomalie sur n'importe quelle valeur MQRC_ * valide.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
        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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
 &CompCode, &Reason);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_close à exécuter avant et après le traitement de xa_close. Utilisez l'identificateur de fonction MQXF_XACLOSE avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel xa_close avant et après.

L'interface de cette fonction est:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXa_info (PMQCHAR)-entrée/sortie
Informations sur le gestionnaire de ressources spécifique à l'instance.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_commit à exécuter avant et après le traitement de xa_commit. Utilisez l'identificateur de fonction MQXF_XACOMMIT avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel avant et après xa_commit.

L'interface de cette fonction est:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXID (MQPTR)-entrée/sortie
ID de branche de transaction.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_complete à exécuter avant et après le traitement xa_complete. Utilisez l'identificateur de fonction MQXF_XACOMPLETE avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel avant et après xa_complete.

L'interface de cette fonction est:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pHandle (PMQLONG)-entrée/sortie
Pointeur vers l'opération asynchrone.

pRetVal (PMQLONG)-entrée/sortie
Valeur de retour de l'opération asynchrone.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_end à exécuter avant et après le traitement xa_end. Utilisez l'identificateur de fonction MQXF_XAEND avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel avant et après xa_end.

L'interface de cette fonction est:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXID (MQPTR)-entrée/sortie
ID de branche de transaction.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_forget à exécuter avant et après le traitement de xa_forget. Utilisez l'identificateur de fonction MQXF_XAFORGET avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel xa_forget avant et après.

L'interface de cette fonction est:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXID (MQPTR)-entrée/sortie
ID de branche de transaction.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_open à exécuter avant et après le traitement xa_open. Utilisez l'identificateur de fonction MQXF_XAOPEN avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel xa_open avant et après.

L'interface de cette fonction est:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXa_info (PMQCHAR)-entrée/sortie
Informations sur le gestionnaire de ressources spécifique à l'instance.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_prepare à exécuter avant et après le traitement xa_prepare. Utilisez l'identificateur de fonction MQXF_XAPREPARE avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel xa_prepare avant et après.

L'interface de cette fonction est:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXID (MQPTR)-entrée/sortie
ID de branche de transaction.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_recover à exécuter avant et après le traitement xa_recover. Utilisez l'identificateur de fonction MQXF_XARECOVER avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel xa_recover avant et après.

L'interface de cette fonction est:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXID (MQPTR)-entrée/sortie
ID de branche de transaction.

Nombre (MQLONG)-entrée / sortie
Nombre maximal de XID dans le tableau XID

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_rollback à exécuter avant et après le traitement de xa_rollback. Utilisez l'identificateur de fonction MQXF_XAROLLBACK avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel xa_rollback avant et après.

L'interface de cette fonction est:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXID (MQPTR)-entrée/sortie
ID de branche de transaction.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit xa_start à exécuter avant et après le traitement xa_start. Utilisez l'identificateur de fonction MQXF_XASTART avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel avant et après xa_start.

L'interface de cette fonction est:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXID (MQPTR)-entrée/sortie
ID de branche de transaction.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit ax_reg à exécuter avant et après le traitement ax_reg. Utilisez l'identificateur de fonction MQXF_AXREG avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel ax_reg avant et après.

L'interface de cette fonction est:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode)
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Hconn (MQHCONN)-entrée
Descripteur de connexion.

pXID (MQPTR)-entrée/sortie
ID de branche de transaction.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 fournit une fonction d'exit ax_unreg à exécuter avant et après le traitement ax_unreg. Utilisez l'identificateur de fonction MQXF_AXUNREG avec les raisons d'exit MQXR_BEFORE et MQXR_AFTER pour enregistrer les fonctions d'exit d'appel ax_unreg avant et après.

L'interface de cette fonction est:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

où les paramètres sont :

ExitParms (MQAXP)-entrée/sortie
Structure des paramètres d'exit.

ExitContext (MQAXC)-entrée/sortie
Quittez la structure de contexte.

Rmid (MQLONG)-entrée / sortie
Identificateur du gestionnaire de ressources.

Indicateurs (MQLONG)-entrée/sortie
Options du gestionnaire de ressources.

XARetCode (MQLONG)-entrée/sortie
Réponse de l'appel XA.

Appel du langage C

Le gestionnaire de files d'attente définit logiquement les variables suivantes:

```
     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 */
```

Le gestionnaire de files d'attente appelle ensuite logiquement l'exit comme suit:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

Votre exit doit correspondre au prototype de fonction C suivant:

```
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 */
```

Informations générales sur l'appel des fonctions d'exit

Cette rubrique fournit des conseils généraux pour vous aider à planifier vos exits, en particulier en ce qui concerne le traitement des erreurs et des événements inattendus.

Incident de sortie

Si une fonction d'exit s'arrête de manière anormale après un appel MQGET destructif, hors point de synchronisation, mais avant que le message n'ait été transmis à l'application, le gestionnaire d'exit peut récupérer de l'échec et transmettre le contrôle à l'application.

Dans ce cas, le message peut être perdu. Il s'agit de ce qui se produit lorsqu'une application échoue immédiatement après avoir reçu un message d'une file d'attente.

L'appel MQGET peut aboutir avec MQCC_FAILED et MQRC_API_EXIT_ERROR.

Si une fonction d'exit d'appel d'API *avant* s'arrête de manière anormale, le gestionnaire d'exit peut effectuer une reprise après l'échec et transmettre le contrôle à l'application sans traiter l'appel d'API. Dans ce cas, la fonction d'exit doit récupérer les ressources qu'elle possède.

Si des exits chaînés sont utilisés, les exits d'appel d'API *après* pour les exits d'appel d'API *avant* qui ont été déclenchés avec succès peuvent eux-mêmes être déclenchés. L'appel API peut échouer avec MQCC_FAILED et MQRC_API_EXIT_ERROR.

Exemple de traitement d'erreurs pour les fonctions de sortie

Le diagramme suivant montre les points (e*N*) auxquels des erreurs peuvent se produire. Il ne s'agit que d'un exemple montrant comment les exits se comportent et doivent être lus avec le tableau suivant. Dans cet exemple, deux fonctions d'exit sont appelées avant et après chaque appel d'API pour afficher le comportement avec des exits chaînés.

```
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
```

Le tableau suivant répertorie les actions à effectuer à chaque point d'erreur. Seul un sous-ensemble des points d'erreur a été couvert, car les règles affichées ici peuvent s'appliquer à tous les autres. Il s'agit des actions qui spécifient le comportement prévu dans chaque cas.

Tableau 595. Erreurs de sortie d'API et actions appropriées à effectuer

Err Pt	Description	Actions
e1	Erreur lors de la configuration de l'environnement.	1. Annuler la configuration de l'environnement si nécessaire 2. Aucune fonction d'exit de l'unité 3. Echec de MQCONN avec MQCC_FAILED, MQRC_API_EXIT_LOAD_ERROR

Tableau 595. Erreurs de sortie d'API et actions appropriées à effectuer (suite)

Err Pt	Description	Actions
e2	La fonction MQ_INIT_EXIT se termine par: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED: 1. Nettoyer l'environnement 2. Echec de MQCONN avec MQCC_FAILED, MQRC_API_EXIT_INIT_ERROR • Pour MQXCC_ * 1. Agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1] 2. Nettoyer l'environnement
e3	*Avant* que la fonction MQ_CONNX_EXIT 1 se termine par: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED: 1. Fonction MQ_TERM_EXIT de l'unité 2. Nettoyer l'environnement 3. Echec de l'appel MQCONN avec MQCC_FAILED, MQRC_API_EXIT_ERROR • Pour MQXCC_ * 1. Agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1] 2. Fonction MQ_TERM_EXIT de l'unité, si nécessaire 3. Nettoyer l'environnement si nécessaire
e4	*Avant* que la fonction MQ_CONNX_EXIT 2 se termine par: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED: 1. Unité *après la fonction* MQ_CONNX_EXIT 1 2. Fonction MQ_TERM_EXIT de l'unité 3. Nettoyer l'environnement 4. Echec de l'appel MQCONN avec MQCC_FAILED, MQRC_API_EXIT_ERROR • Pour MQXCC_ * 1. Agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1] 2. Unité *après la fonction* MQ_CONNX_EXIT 1 si l'exit n'est pas supprimé 3. Fonction MQ_TERM_EXIT de l'unité, si nécessaire 4. Nettoyer l'environnement si nécessaire
e5	Echec de l'appel MQCONN.	1. Transmission de MQCONN CompCode et de la raison 2. Lecteur *après* la fonction MQ_CONNX_EXIT 2 si le *antérieur à* MQ_CONNX_EXIT 2 a abouti et que l'exit n'est pas supprimé 3. Unité *après la fonction* MQ_CONNX_EXIT 1 si la valeur *antérieure à* MQ_CONNX_EXIT 1 a abouti et que l'exit n'est pas supprimé 4. Fonction MQ_TERM_EXIT de l'unité 5. Nettoyer l'environnement

Tableau 595. Erreurs de sortie d'API et actions appropriées à effectuer (suite)

Err Pt	Description	Actions
e6	*Après* l'exécution de la fonction MQ_CONNX_EXIT 2 avec: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED: 1. Unité *après la fonction* MQ_CONNX_EXIT 1 2. Terminez l'appel MQCONN avec MQCC_FAILED, MQRC_API_EXIT_ERROR • Pour MQXCC_ * 1. Agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1] 2. Unité *après la fonction* MQ_CONNX_EXIT 1, si nécessaire
e7	*Après* l'exécution de la fonction MQ_CONNX_EXIT 1 avec: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED, effectuez l'appel MQCONN avec MQCC_FAILED, MQRC_API_EXIT_ERROR • Pour MQXCC_ *, agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1]
e8	*Avant* , la fonction MQ_OPEN_EXIT 1 se termine par: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED, effectuez l'appel MQOPEN avec MQCC_FAILED, MQRC_API_EXIT_ERROR • Pour MQXCC_ *, agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1]
e9	*Avant* , la fonction MQ_OPEN_EXIT 2 se termine par: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED: 1. Unité *après la fonction* MQ_OPEN_EXIT 1 2. Terminer l'appel MQOPEN avec MQCC_FAILED, MQRC_API_EXIT_ERROR • Pour MQXCC_ *, agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1]
e10	Echec de l'appel MQOPEN	1. Transmission de MQOPEN CompCode et raison 2. Unité *après* la fonction MQ_OPEN_EXIT 2 si l'exit n'est pas supprimé 3. Unité *après la fonction* MQ_OPEN_EXIT 1 si l'exit n'est pas supprimé et si les exits chaînés ne sont pas supprimés
e11	*Après* l'exécution de la fonction MQ_OPEN_EXIT 2 avec: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED: 1. Unité *après la fonction* MQ_OPEN_EXIT 1 2. Terminer l'appel MQOPEN avec MQCC_FAILED, MQRC_API_EXIT_ERROR • Pour MQXCC_ * 1. Agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1] 2. Unité *après* la fonction MQ_OPEN_EXIT 1 si l'exit n'est pas supprimé

Tableau 595. Erreurs de sortie d'API et actions appropriées à effectuer (suite)		
Err Pt	**Description**	**Actions**
e25	*Après* l'exécution de la fonction MQ_DISC_EXIT 2 avec: • MQXCC_ECHEC • MQXCC_ *	• Pour MQXCC_FAILED: 1. Unité *après la fonction* MQ_DISC_EXIT 1 2. Fonction MQ_TERM_EXIT de l'unité 3. Nettoyer l'environnement d'exécution de l'exit 4. Terminer l'appel MQDISC avec MQCC_FAILED, MQRC_API_EXIT_ERROR • Pour MQXCC_ * 1. Agir comme pour les valeurs de MQXCC_ * et MQXR2_ *[1] 2. Fonction MQ_TERM_EXIT de l'unité 3. Nettoyer l'environnement d'exécution de l'exit

Remarque :

1. Les valeurs de MQXCC_ * et MQXR2_* et les actions correspondantes sont définies dans Fonctionnement des exits des gestionnaires de files d'attente.

Les champs ExitResponse ne sont pas définis correctement

Cette rubrique fournit des informations sur ce qui se passerait lorsque la zone ExitResponse est définie sur une valeur autre que les valeurs prises en charge.

Si la zone ExitResponse est définie sur une valeur autre que l'une des valeurs prises en charge, les actions suivantes s'appliquent:

- Pour une fonction d'exit API *avant* MQCONN ou MQDISC:
  - La valeur ExitResponse2 est ignorée.
  - Aucune autre fonction d'exit *avant* dans la chaîne d'exit (le cas échéant) n'est appelée ; l'appel d'API lui-même n'est pas émis.
  - Pour les exits *avant* qui ont été appelés avec succès, les exits *après* sont appelés dans l'ordre inverse.
  - Si elles sont enregistrées, les fonctions d'exit de fin pour les fonctions d'exit *avant* MQCONN ou MQDISC de la chaîne qui ont été appelées avec succès sont déclenchées pour le nettoyage après ces fonctions d'exit.
  - L'appel MQCONN ou MQDISC échoue avec MQRC_API_EXIT_ERROR.
- Pour une fonction d'exit API *avant* WebSphere MQ autre que MQCONN ou MQDISC:
  - La valeur ExitResponse2 est ignorée.
  - Aucune autre fonction de conversion de données *avant* ou *après* dans la chaîne d'exit (le cas échéant) n'est appelée.
  - Pour les exits *avant* qui ont été appelés avec succès, les exits *après* sont appelés dans l'ordre inverse.
  - L'appel d'API WebSphere MQ lui-même n'est pas émis.
  - L'appel API WebSphere MQ échoue avec MQRC_API_EXIT_ERROR.
- Pour une fonction d'exit API MQCONN ou MQDISC *après* :
  - La valeur ExitResponse2 est ignorée.
  - Les autres fonctions d'exit qui ont été appelées avec succès avant l'appel d'API sont appelées dans l'ordre inverse.

- Si elles sont enregistrées, les fonctions d'exit d'arrêt pour les fonctions d'exit *avant* ou *après* MQCONN ou MQDISC de la chaîne qui ont été appelées avec succès sont exécutées pour être nettoyées après l'exit.
- Un CompCode plus grave de MQCC_WARNING et le CompCode renvoyé par l'exit est renvoyé à l'application.
- Une raison de MQRC_API_EXIT_ERROR est renvoyée à l'application.
- L'appel API WebSphere MQ a été émis.

- Pour une fonction d'exit d'appel d'API *après* WebSphere MQ autre que MQCONN ou MQDISC:
  - La valeur ExitResponse2 est ignorée.
  - Les autres fonctions d'exit qui ont été appelées avec succès avant l'appel d'API sont appelées dans l'ordre inverse.
  - Un CompCode plus grave de MQCC_WARNING et le CompCode renvoyé par l'exit est renvoyé à l'application.
  - Une raison de MQRC_API_EXIT_ERROR est renvoyée à l'application.
  - L'appel API WebSphere MQ a été émis.
- Pour la fonction *before* data conversion on get exit:
  - La valeur ExitResponse2 est ignorée.
  - Les autres fonctions d'exit qui ont été appelées avec succès avant l'appel d'API sont appelées dans l'ordre inverse.
  - Le message n'est pas converti et le message non converti est renvoyé à l'application.
  - Un CompCode plus grave de MQCC_WARNING et le CompCode renvoyé par l'exit est renvoyé à l'application.
  - Une raison de MQRC_API_EXIT_ERROR est renvoyée à l'application.
  - L'appel API WebSphere MQ a été émis.

  Remarque : Comme l'erreur est liée à l'exit, il est préférable de renvoyer MQRC_API_EXIT_ERROR que de renvoyer MQRC_NOT_CONVERTIS.

Si une fonction d'exit définit la zone ExitResponse2 sur une valeur autre que l'une des valeurs prises en charge, la valeur MQXR2_DEFAULT_CONTINUATION est utilisée à la place.

Informations de référence de l'interface des services installables

Cette collection de rubriques fournit des informations de référence pour les services installables.

Les fonctions et les types de données sont répertoriés par ordre alphabétique au sein du groupe pour chaque type de service.

Affichage des fonctions

La façon dont les fonctions des services installables sont documentées.

Pour chaque fonction, il existe une description, y compris l'identificateur de fonction (pour MQZEP).

Les *paramètres* sont répertoriés dans l'ordre dans lequel ils doivent apparaître. Ils doivent tous être présents.

Chaque nom de paramètre est suivi de son type de données. Il s'agit des types de données élémentaires décrits dans le «Types de données élémentaires», à la page 218.

L'appel de langage C est également fourni, après la description des paramètres.

MQZ_AUTHENTICATE_USER-Authentification de l'utilisateur

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_5 et est appelée par le gestionnaire de files d'attente pour authentifier un utilisateur ou pour définir des zones de contexte d'identité. Il est appelé lorsque le contexte d'application utilisateur de WebSphere MQest établi.

Le contexte d'application est établi lors des appels de connexion au point où le contexte utilisateur de l'application est initialisé et à chaque point où le contexte utilisateur de l'application est modifié. Chaque fois qu'un appel de connexion est effectué, les informations de contexte utilisateur de l'application sont réacquises dans la zone *IdentityContext* .

L'identificateur de cette fonction (pour MQZEP) est MQZID_AUTHENTICATE_USER.

Syntaxe

MQZ_AUTHENTICATE_USER (*QMgrName*, *SecurityParms*, *ApplicationContext*, *IdentityContext*, *CorrelationPtr*, *ComponentData*, *Continuation*, *CompCode*, *Raison*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

SecurityParms
Type: MQCSP-entrée

Paramètres de sécurité. Données relatives à l'ID utilisateur, au mot de passe et au type d'authentification. Si l'attribut AuthenticationType de la structure MQCSP est spécifié comme MQCSP_AUTH_USER_ID_AND_PWD, l'ID utilisateur et le mot de passe sont comparés aux zones équivalentes dans le paramètre IdentityContext (MQZIC) pour déterminer s'ils correspondent à. Pour plus d'informations, voir «MQCSP-Paramètres de sécurité», à la page 314.

Lors d'un appel MQCONN MQI, ce paramètre contient des valeurs null ou des valeurs par défaut.

ApplicationContext
Type: MQZAC-entrée

Contexte d'application. Données relatives à l'application appelante. Voir MQZAC-Contexte d'application pour plus de détails.

Lors de chaque appel MQCONN ou MQCONNX MQI, les informations de contexte utilisateur de la structure MQZAC sont réacquises.

IdentityContext
Type: MQZIC-entrée / sortie

Contexte d'identité. Dans l' entrée de la fonction d'authentification de l'utilisateur, this identifie le contexte d'identité en cours. La fonction d'authentification de l'utilisateur peut modifier ce, auquel cas le gestionnaire de files d'attente adopte le nouveau contexte d'identité. Voir Contexte MQZIC-Identity pour plus de détails sur la structure MQZIC.

CorrelationPtr
Type: MQPTR-sortie

Pointeur de corrélation. Indique l'adresse des données de corrélation. Ce pointeur est ensuite transmis à d'autres appels OAM.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. This data is kept by the queue manager on behalf of this particular component; any changes made to it by any of the functions provided by this component are preserved, and presented the next time one of this component's functions is called.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation. Vous pouvez spécifier les valeurs suivantes:

MQZCI_DEFAULT
Continuation dépendante des autres composants.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d' informations sur ces codes anomalie, voir Codes anomalie.

Appel C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                  IdentityContext, &CorrelationPtr, ComponentData,
                  &Continuation, &CompCode, &Reason);
```

Déclarez les paramètres transmis au service comme suit:

```
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-Vérification des droits

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_1 et est démarrée par le gestionnaire de files d'attente pour vérifier si une entité est autorisée à effectuer une ou plusieurs actions particulières sur un objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID_CHECK_AUTHORITY.

Syntaxe

MQZ_CHECK_AUTHORITY(*QMgrName*, *EntityName*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityName
Type: MQCHAR12 -entrée

Nom de l'entité. Nom de l'entité dont l'autorisation sur l'objet doit être vérifiée. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

Il n'est pas essentiel que cette entité soit connue du service de sécurité sous-jacent. S'il n'est pas connu, les autorisations du groupe spécial **nobody** (auquel toutes les entités sont supposées appartenir) sont utilisées pour la vérification. Un nom vide est valide et peut être utilisé de cette manière.

EntityType
Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par EntityName. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL
Entité.

GROUPE MQZAET_GROUP
.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet auquel l'accès est requis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

authority
Type : MQLONG - entrée

Droits à vérifier. Si une autorisation est vérifiée, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO_ *). Si plusieurs autorisations sont vérifiées, il s'agit du OU bit à bit des constantes MQZAO_ * correspondantes.

Les autorisations suivantes s'appliquent à l'utilisation des appels MQI:

MQZAO_CONNECT
Possibilité d'utiliser l'appel MQCONN.

MQZAO_PARCOURIR
Possibilité d'utiliser l'appel MQGET avec une option de navigation.

Cela permet de spécifier l'option MQGMO_BROWSE_FIRST, MQGMO_BROWSE_MSG_UNDER_CURSOR ou MQGMO_BROWSE_NEXT sur l'appel MQGET.

MQZAO_ENTREE
Entité. Possibilité d'utiliser l'appel MQGET avec une option d'entrée.

Cela permet à l'option MQOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE ou MQOO_INPUT_AS_Q_DEF d'être spécifiée sur l'appel MQOPEN.

MQZAO_OUTPUT
Possibilité d'utiliser l'appel MQPUT.

Cela permet à l'option MQOO_OUTPUT d'être spécifiée sur l'appel MQOPEN.

MQZAO_INQUIRE
Possibilité d'utiliser l'appel MQINQ.

Cela permet de spécifier l'option MQOO_INQUIRE sur l'appel MQOPEN.

MQZAO_SET
Possibilité d'utiliser l'appel MQSET.

Cela permet de spécifier l'option MQOO_SET sur l'appel MQOPEN.

MQZAO_PASS_IDENTITY_CONTEXT
Possibilité de transmettre le contexte d'identité.

Cela permet de spécifier l'option MQOO_PASS_IDENTITY_CONTEXT sur l'appel MQOPEN et l'option MQPMO_PASS_IDENTITY_CONTEXT sur les appels MQPUT et MQPUT1 .

MQZAO_PASS_ALL_CONTEXT
Possibilité de transmettre tout le contexte.

Cela permet de spécifier l'option MQOO_PASS_ALL_CONTEXT sur l'appel MQOPEN et l'option MQPMO_PASS_ALL_CONTEXT sur les appels MQPUT et MQPUT1 .

MQZAO_SET_IDENTITY_CONTEXT
Possibilité de définir le contexte d'identité.

Cela permet de spécifier l'option MQOO_SET_IDENTITY_CONTEXT sur l'appel MQOPEN et l'option MQPMO_SET_IDENTITY_CONTEXT sur les appels MQPUT et MQPUT1 .

MQZAO_SET_ALL_CONTEXT
Possibilité de définir tous les contextes.

Cela permet de spécifier l'option MQOO_SET_ALL_CONTEXT sur l'appel MQOPEN et l'option MQPMO_SET_ALL_CONTEXT sur les appels MQPUT et MQPUT1 .

MQZAO_ALTERNATE_USER_AUTHORITY
Possibilité d'utiliser des droits utilisateur de remplacement.

Cela permet de spécifier l'option MQOO_ALTERNATE_USER_AUTHORITY sur l'appel MQOPEN et l'option MQPMO_ALTERNATE_USER_AUTHORITY sur l'appel MQPUT1 .

MQZAO_ALL_MQI
Toutes les autorisations MQI.

Cela active toutes les autorisations.

Les autorisations suivantes s'appliquent à l'administration d'un gestionnaire de files d'attente:

MQZAO_CREER
Possibilité de créer des objets d'un type spécifié.

MQZAO_DELETE
Possibilité de supprimer un objet spécifié.

MQZAO_DISPLAY
Possibilité d'afficher les attributs d'un objet spécifié.

MODIFICATION MQZAO_DE
Possibilité de modifier les attributs d'un objet spécifié.

MQZAO_CLEAR
Possibilité de supprimer tous les messages d'une file d'attente spécifiée.

MQZAO_AUTORISATION
Possibilité d'autoriser d'autres utilisateurs pour un objet spécifié.

CONTROLE MQZ
Possibilité de démarrer ou d'arrêter un programme d'écoute, un service ou un objet de canal non client, et possibilité de lancer une commande ping sur un objet de canal non client.

MQZAO_CONTRÔLE_ÉTENDU
Possibilité de réinitialiser un numéro de séquence ou de résoudre un message en attente de validation sur un objet canal non client.

MQZAO_ALL_ADMIN
Possibilité de définir le contexte d'identité.

Toutes les autorisations d'administration, autres que MQZAO_CREATE.

Les autorisations suivantes s'appliquent à l'utilisation de l'interface MQI et à l'administration d'un gestionnaire de files d'attente:

MQZAO_ALL
Toutes les autorisations, autres que MQZAO_CREATE.

MQZAO_AUCUN
Aucune autorisation.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des

fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

Si l'appel à un composant échoue (c'est-à-dire que *CompCode* renvoie MQCC_FAILED) et que le paramètre *Continuation* est MQZCI_DEFAULT ou MQZCI_CONTINUE, le gestionnaire de files d'attente continue d'appeler d'autres composants s'il en existe.

Si l'appel aboutit (c'est-à-dire si *CompCode* renvoie MQCC_OK), aucun autre composant n'est appelé, quelle que soit la valeur de *Continuation* .

Si l'appel échoue et que le paramètre *Continuation* est MQZCI_STOP, aucun autre composant n'est appelé et l'erreur est renvoyée au gestionnaire de files d'attente. Les composants ne connaissant pas les appels précédents, le paramètre *Continuation* est toujours défini sur MQZCI_DEFAULT avant l'appel.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                     ObjectType, Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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 -Vérification des droits (étendu)

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_2 et est démarrée par le gestionnaire de files d'attente pour vérifier si une entité est autorisée à effectuer une ou plusieurs actions sur un objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID_CHECK_AUTHORITY.

MQZ_CHECK_AUTHORITY_2 est similaire à MQZ_CHECK_AUTHORITY, mais avec le paramètre *EntityName* remplacé par le paramètre *EntityData* .

Syntaxe

MQZ_CHECK_AUTHORITY_2(*QMgrName*, *EntityData*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityData
Type: MQZED-entrée

Données d'entité. Données relatives à l'entité avec autorisation sur l'objet à vérifier. Voir «MQZED-Descripteur d'entité», à la page 1221 pour plus de détails.

Il n'est pas essentiel que cette entité soit connue du service de sécurité sous-jacent. S'il n'est pas connu, les autorisations du groupe spécial **nobody** (auquel toutes les entités sont supposées appartenir) sont utilisées pour la vérification. Un nom vide est valide et peut être utilisé de cette manière.

EntityType
Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityData*. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL
Entité.

GROUPE MQZAET_GROUP
.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet auquel l'accès est requis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

authority
Type : MQLONG - entrée

Droits à vérifier. Si une autorisation est vérifiée, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO_ *). Si plusieurs autorisations sont vérifiées, il s'agit du OU bit à bit des constantes MQZAO_ * correspondantes.

Les autorisations suivantes s'appliquent à l'utilisation des appels MQI:

MQZAO_CONNECT
Possibilité d'utiliser l'appel MQCONN.

MQZAO_PARCOURIR
Possibilité d'utiliser l'appel MQGET avec une option de navigation.

Cela permet de spécifier l'option MQGMO_BROWSE_FIRST, MQGMO_BROWSE_MSG_UNDER_CURSOR ou MQGMO_BROWSE_NEXT sur l'appel MQGET.

MQZAO_ENTREE
Entité. Possibilité d'utiliser l'appel MQGET avec une option d'entrée.

Cela permet à l'option MQOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE ou MQOO_INPUT_AS_Q_DEF d'être spécifiée sur l'appel MQOPEN.

MQZAO_OUTPUT
Possibilité d'utiliser l'appel MQPUT.

Cela permet à l'option MQOO_OUTPUT d'être spécifiée sur l'appel MQOPEN.

MQZAO_INQUIRE
Possibilité d'utiliser l'appel MQINQ.

Cela permet de spécifier l'option MQOO_INQUIRE sur l'appel MQOPEN.

MQZAO_SET
Possibilité d'utiliser l'appel MQSET.

Cela permet de spécifier l'option MQOO_SET sur l'appel MQOPEN.

MQZAO_PASS_IDENTITY_CONTEXT
Possibilité de transmettre le contexte d'identité.

Cela permet de spécifier l'option MQOO_PASS_IDENTITY_CONTEXT sur l'appel MQOPEN et l'option MQPMO_PASS_IDENTITY_CONTEXT sur les appels MQPUT et MQPUT1 .

MQZAO_PASS_ALL_CONTEXT
Possibilité de transmettre tout le contexte.

Cela permet de spécifier l'option MQOO_PASS_ALL_CONTEXT sur l'appel MQOPEN et l'option MQPMO_PASS_ALL_CONTEXT sur les appels MQPUT et MQPUT1 .

MQZAO_SET_IDENTITY_CONTEXT
Possibilité de définir le contexte d'identité.

Cela permet de spécifier l'option MQOO_SET_IDENTITY_CONTEXT sur l'appel MQOPEN et l'option MQPMO_SET_IDENTITY_CONTEXT sur les appels MQPUT et MQPUT1 .

MQZAO_SET_ALL_CONTEXT
Possibilité de définir tous les contextes.

Cela permet de spécifier l'option MQOO_SET_ALL_CONTEXT sur l'appel MQOPEN et l'option MQPMO_SET_ALL_CONTEXT sur les appels MQPUT et MQPUT1 .

MQZAO_ALTERNATE_USER_AUTHORITY
Possibilité d'utiliser des droits utilisateur de remplacement.

Cela permet de spécifier l'option MQOO_ALTERNATE_USER_AUTHORITY sur l'appel MQOPEN et l'option MQPMO_ALTERNATE_USER_AUTHORITY sur l'appel MQPUT1 .

MQZAO_ALL_MQI
Toutes les autorisations MQI.

Cela active toutes les autorisations.

Les autorisations suivantes s'appliquent à l'administration d'un gestionnaire de files d'attente:

MQZAO_CREER
Possibilité de créer des objets d'un type spécifié.

MQZAO_DELETE
Possibilité de supprimer un objet spécifié.

MQZAO_DISPLAY
Possibilité d'afficher les attributs d'un objet spécifié.

MODIFICATION MQZAO_DE
Possibilité de modifier les attributs d'un objet spécifié.

MQZAO_CLEAR
Possibilité de supprimer tous les messages d'une file d'attente spécifiée.

MQZAO_AUTORISATION
Possibilité d'autoriser d'autres utilisateurs pour un objet spécifié.

CONTROLE MQZ
Possibilité de démarrer ou d'arrêter un programme d'écoute, un service ou un objet de canal non client, et possibilité de lancer une commande ping sur un objet de canal non client.

MQZAO_CONTRÔLE_ÉTENDU
Possibilité de réinitialiser un numéro de séquence ou de résoudre un message en attente de validation sur un objet canal non client.

MQZAO_ALL_ADMIN
Possibilité de définir le contexte d'identité.

Toutes les autorisations d'administration, autres que MQZAO_CREATE.

Les autorisations suivantes s'appliquent à l'utilisation de l'interface MQI et à l'administration d'un gestionnaire de files d'attente:

MQZAO_ALL
Toutes les autorisations, autres que MQZAO_CREATE.

MQZAO_AUCUN
Aucune autorisation.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_CHECK_AUTHORITY_2 (QMgrName, &EntityData, EntityType,
                       ObjectName, ObjectType, Authority, ComponentData,
                       &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-vérifie si l'utilisateur est privilégié

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_6 et est appelée par le gestionnaire de files d'attente pour déterminer si un utilisateur spécifié est un utilisateur privilégié.

L'identificateur de cette fonction (pour MQZEP) est MQZID_CHECK_PRIVILEGED.

Syntaxe

MQZ_CHECK_PRIVILEGED(*QMgrName*, *EntityData*, *EntityType*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityData
Type: MQZED-entrée

Données d'entité. Données relatives à l'entité à vérifier. Pour plus d'informations, voir «MQZED-Descripteur d'entité», à la page 1221.

EntityType
Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par EntityData. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL
Entité.

GROUPE MQZAET_GROUP
.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

Si l'appel à un composant échoue (c'est-à-dire que *CompCode* renvoie MQCC_FAILED) et que le paramètre *Continuation* est MQZCI_DEFAULT ou MQZCI_CONTINUE, le gestionnaire de files d'attente continue d'appeler d'autres composants s'il en existe.

Si l'appel aboutit (c'est-à-dire si *CompCode* renvoie MQCC_OK), aucun autre composant n'est appelé, quelle que soit la valeur de *Continuation* .

Si l'appel échoue et que le paramètre *Continuation* est MQZCI_STOP, aucun autre composant n'est appelé et l'erreur est renvoyée au gestionnaire de files d'attente. Les composants ne connaissant pas les appels précédents, le paramètre *Continuation* est toujours défini sur MQZCI_DEFAULT avant l'appel.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_PRIVILEGED
(2584, X'A18') Cet utilisateur n'est pas un ID utilisateur privilégié.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entité inconnue du service.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_CHECK_PRIVILEGED (QMgrName, &EntityData, EntityType,
                      ComponentData, &Continuation,
                      &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Copie de tous les droits

Cette fonction est fournie par un composant de service d'autorisation. Il est démarré par le gestionnaire de files d'attente pour copier toutes les autorisations actuellement en vigueur pour un objet de référence dans un autre objet.

L'identificateur de cette fonction (pour MQZEP) est MQZID_COPY_ALL_AUTHORITY.

Syntaxe

MQZ_COPY_ALL_AUTHORITY(*QMgrName*, *RefObjectName*, *ObjectName*, *ObjectType*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

RefObjectNom
Type: MQCHAR48 -entrée

Nom de l'objet de référence. Nom de l'objet de référence dont les autorisations doivent être copiées. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel les accès doivent être définis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *RefObjectName* et *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

MQRC_UNKNOWN_REF_OBJECT
(2294, X'8F6') Objet de référence inconnu.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName, ObjectType,
                        ComponentData, &Continuation, &CompCode,
                        &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Suppression des droits

Cette fonction est fournie par un composant de service d'autorisation et démarrée par le gestionnaire de files d'attente pour supprimer toutes les autorisations associées à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID_DELETE_AUTHORITY.

Syntaxe

MQZ_DELETE_AUTHORITY(*QMgrName*, *ObjectName*, *ObjectType*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel les accès doivent être supprimés. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_DELETE_AUTHORITY (QMgrName, ObjectName, ObjectType, ComponentData,
                      &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Enumération des données de droits d'accès

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_4 et est démarrée à plusieurs reprises par le gestionnaire de files d'attente pour extraire toutes les données de droits d'accès qui correspondent aux critères de sélection spécifiés lors du premier appel.

L'identificateur de cette fonction (pour MQZEP) est MQZID_ENUMERATE_AUTHORITY_DATA.

Syntaxe

MQZ_ENUMERATE_AUTHORITY_DATA(*QMgrName*, *StartEnumeration*, *Filter*, *AuthorityBufferLength*, *AuthorityBuffer*, *AuthorityDataLength*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

StartEnumeration
Type : MQLONG - entrée

Indicateur signalant si l'appel peut démarrer l'énumération. Indique si l'appel peut démarrer l'énumération des données de droits d'accès ou continuer l'énumération des données de droits d'accès démarrée par un appel précédent à MQZ_ENUMERATE_AUTHORITY_DATA. La valeur est l'une des suivantes:

DEMARRAGE MQZ
Démarrer l'énumération. L'appel est démarré avec cette valeur pour démarrer l'énumération des données de droits d'accès. Le paramètre *Filter* indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès renvoyées par cet appel et les appels successifs.

MQZSE_CONTINUE
Continuez l'énumération. L'appel est lancé avec cette valeur pour continuer l'énumération des données de droits. Le paramètre *Filter* est ignoré dans ce cas et peut être spécifié en tant que pointeur null (les critères de sélection sont déterminés par le paramètre *Filter*spécifié par l'appel qui a défini *StartEnumeration* sur MQZSE_START).

Filter
Type: MQZAD-entrée

filtrer. Si *StartEnumeration* est MQZSE_START, *Filter* indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès à renvoyer. Si *Filter* est le pointeur null, aucun critère de sélection n'est utilisé, c'est-à-dire que toutes les données de droits d'accès sont renvoyées. Pour plus de détails sur les critères de sélection pouvant être utilisés, voir «MQZAD-Données de droits d'accès», à la page 1218 .

Si *StartEnumeration* est MQZSE_CONTINUE, *Filter* est ignoré et peut être spécifié en tant que pointeur null.

AuthorityBufferLongueur
Type : MQLONG - entrée

Longueur de *AuthorityBuffer*. Il s'agit de la longueur en octets du paramètre *AuthorityBuffer* . La mémoire tampon des droits d'accès doit être suffisamment grande pour contenir les données à renvoyer.

AuthorityBuffer
Type: MQZAD-sortie

Données de droits d'accès. Il s'agit de la mémoire tampon dans laquelle les données de droits d'accès sont renvoyées. La mémoire tampon doit être suffisamment grande pour accueillir une structure MQZAD, une structure MQZED, plus le nom d'entité le plus long et le nom de domaine le plus long définis.

Remarque : Remarque: Ce paramètre est défini en tant que MQZAD, car MQZAD se produit toujours au début de la mémoire tampon. Toutefois, si la mémoire tampon est déclarée en tant que MQZAD, elle sera trop petite-elle doit être plus grande qu'une MQZAD pour pouvoir accueillir les noms d'entité et de domaine MQZAD, MQZED.

AuthorityDataLongueur
Type : MQLONG - sortie

Longueur des données renvoyées dans *AuthorityBuffer*. Si la mémoire tampon des droits est trop petite, *AuthorityDataLength* est défini sur la longueur de la mémoire tampon requise et l'appel renvoie le code achèvement MQCC_FAILED et le code anomalie MQRC_BUFFER_LENGTH_ERROR.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_ENUMERATE_AUTHORITY_DATA, cela a le même effet que MQZCI_CONTINUE.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Paramètre de capacité de mémoire tampon non valide.

MQRC_NO_DATA_XX_ENCODE_CASE_ONE disponible
(2379, X'94B') Aucune donnée disponible.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration, &Filter,
                              AuthorityBufferLength,
```

```
                              &AuthorityBuffer,
                              &AuthorityDataLength, ComponentData,
                              &Continuation, &CompCode,
                              &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Utilisateur libre

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_5 et est démarrée par le gestionnaire de files d'attente pour libérer la ressource allouée associée.

Elle est démarrée lorsqu'une application a fini de s'exécuter dans tous les contextes utilisateur, par exemple lors d'un appel MQDISC MQI.

L'identificateur de cette fonction (pour MQZEP) est MQZID_FREE_USER.

Syntaxe

MQZ_FREE_USER(*QMgrName*, *FreeParms*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

FreeParms

Type: MQZFP-entrée

Paramètres libres. Structure contenant des données relatives à la ressource à libérer. Pour plus d'informations, voir «MQZFP-Paramètres libres», à la page 1224.

ComponentData

Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ_INIT_AUTHORITY.

Continuation

Type : MQLONG - sortie

Indicateur de continuation. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendante des autres composants.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                   IdentityContext, CorrelationPtr, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Obtention des droits

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_1 et est démarrée par le gestionnaire de files d'attente pour extraire les droits dont une entité dispose pour accéder à l'objet spécifié, y compris (si l'entité est un principal) les droits détenus par les groupes dont le principal est membre. Les droits des profils génériques sont inclus dans le jeu de droits renvoyé.

L'identificateur de cette fonction (pour MQZEP) est MQZID_GET_AUTHORITY.

Syntaxe

MQZ_GET_AUTHORITY(*QMgrName*, *EntityName*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityName
Type: MQCHAR12 -entrée

Nom de l'entité. Nom de l'entité dont l'accès à l'objet doit être extrait. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

EntityType
Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityName*. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL
Entité.

GROUPE MQZAET_GROUP
.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet auquel l'accès doit être extrait. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

authority
Type : MQLONG - entrée

Autorité de l'entité. Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO_ *). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO_ * correspondantes.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_GET_AUTHORITY, cela a le même effet que MQZCI_CONTINUE.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_GET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, &Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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 -Obtention des droits (étendu)

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_2 et est démarrée par le gestionnaire de files d'attente pour extraire les droits dont dispose une entité pour accéder à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID_GET_AUTHORITY.

MQZ_GET_AUTHORITY_2 est similaire à MQZ_GET_AUTHORITY, mais avec le paramètre *EntityName* remplacé par le paramètre *EntityData* .

Syntaxe

MQZ_GET_AUTHORITY_2(*QMgrName*, *EntityData*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityData

Type: MQZED-entrée

Données d'entité. Données relatives à l'entité pour laquelle l'autorisation sur l'objet doit être extraite. Voir «MQZED-Descripteur d'entité», à la page 1221 pour plus de détails.

EntityType

Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityData*. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL

Entité.

GROUPE MQZAET_GROUP

.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel le droit d'entité doit être extrait. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

authority
Type : MQLONG - entrée

Autorité de l'entité. Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO_ *). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO_ * correspondantes.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Syntaxe

MQZ_GET_AUTHORITY_2*(QMgrName, EntityData, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)*

Appel C

```
MQZ_GET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                     ObjectType, &Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Obtention des droits explicites

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_1 et est démarrée par le gestionnaire de files d'attente pour extraire les droits dont dispose un groupe nommé pour accéder à un objet spécifié (mais sans les droits supplémentaires du groupe **nobody**) ou les droits dont dispose le groupe principal du principal nommé pour accéder à un objet spécifié.

Sur les plateformes UNIX, pour le gestionnaire des droits d'accès aux objets (OAM) WebSphere MQ intégré, les droits d'accès renvoyés sont ceux détenus uniquement par le groupe principal du principal.

L'identificateur de cette fonction (pour MQZEP) est MQZID_GET_EXPLICIT_AUTHORITY.

Syntaxe

```
MQZ_GET_EXPLICIT_AUTHORITY( QMgrName, EntityName, EntityType, ObjectName,
ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)
```

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityName
Type: MQCHAR12 -entrée

Nom de l'entité. Nom de l'entité pour laquelle l'accès à l'objet doit être extrait. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

EntityType
Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityName*. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL
Entité.

GROUPE MQZAET_GROUP
.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel le droit d'entité doit être extrait. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

authority
Type : MQLONG - entrée

Autorité de l'entité. Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO_ *). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO_ * correspondantes.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_GET_AUTHORITY, cela a le même effet que MQZCI_CONTINUE.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,
                            ObjectName, ObjectType, &Authority,
                            ComponentData, &Continuation,
                            &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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 -Permet d'obtenir des droits explicites (étendu)

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_2 et est démarrée par le gestionnaire de files d'attente pour extraire les droits dont dispose un groupe nommé pour accéder à un objet spécifié (mais sans les droits supplémentaires du groupe **personne**), ou les droits dont dispose le groupe principal du principal nommé pour accéder à un objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID_GET_EXPLICIT_AUTHORITY.

MQZ_GET_EXPLICIT_AUTHORITY_2 est similaire à MQZ_GET_EXPLICIT_AUTHORITY, mais avec le paramètre *EntityName* remplacé par le paramètre *EntityData* .

Syntaxe

MQZ_GET_EXPLICIT_AUTHORITY_2(*QMgrName*, *EntityData*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityData
Type: MQZED-entrée

Données d'entité. Données relatives à l'entité dont l'autorisation sur l'objet doit être extraite. Voir «MQZED-Descripteur d'entité», à la page 1221 pour plus de détails.

EntityType
Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityData*. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL
Entité.

GROUPE MQZAET_GROUP
.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet pour lequel le droit d'entité doit être extrait. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

authority
Type : MQLONG - entrée

Autorité de l'entité. Si l'entité dispose d'un droit d'accès, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO_ *). S'il possède plusieurs droits, cette zone correspond au OU bit à bit des constantes MQZAO_ * correspondantes.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_GET_EXPLICIT_AUTHORITY_2 (QMgrName, &EntityData, EntityType,
                              ObjectName, ObjectType, &Authority,
                              ComponentData, &Continuation,
                              &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Initialisation du service d'autorisation

Cette fonction est fournie par un composant de service d'autorisation et est démarrée par le gestionnaire de files d'attente lors de la configuration du composant. Il est prévu d'appeler MQZEP afin de fournir des informations au gestionnaire de files d'attente.

L'identificateur de cette fonction (pour MQZEP) est MQZID_INIT_AUTHORITY.

Syntaxe

MQZ_INIT_AUTHORITY(*Hconfig*, *Options*, *QMgrName*, *ComponentDataLength*, *ComponentData*, *Version*, *CompCode*, *Reason*)

Paramètres

Configuration Hconfig
Type: MQHCONFIG-entrée

Descripteur de configuration. Cette poignée représente le composant particulier en cours d'initialisation. Il doit être utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

Options
Type : MQLONG - entrée

Options d'initialisation. Il doit s'agir de l'une des valeurs suivantes:

MQZIO_PRIMAIRE
Initialisation principale.

MQZIO_SECONDAIRE
Initialisation secondaire.

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

ComponentDataLongueur
Type : MQLONG - entrée

Longueur des données de composant. Longueur en octets de la zone *ComponentData* . Cette longueur est définie dans les données de configuration du composant.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Il est initialisé à zéro avant d'appeler la fonction d'initialisation principale du composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

version
Type: MQLONG-entrée / sortie

Numéro de version. En entrée de la fonction d'initialisation, identifie le numéro de version le plus élevé pris en charge par le gestionnaire de files d'attente. La fonction d'initialisation doit modifier, si nécessaire, la version de l'interface qu'elle prend en charge. Si, à son retour, le gestionnaire de files d'attente ne prend pas en charge la version renvoyée par le composant, il appelle la fonction MQZ_TERM_AUTHORITY du composant et n'utilise plus ce composant.

Les valeurs suivantes sont prises en charge :

MQZAS_VERSION_1
Version 1.

MQZAS_VERSION_2
Version 2.

MQZAS_VERSION_3
Version 3.

MQZAS_VERSION_4
Version 4.

MQZAS_VERSION_5
Version 5.

MQZAS_VERSION_6
Version 6.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_INITIALIZATION_FAILED
(2286, X'8EE') L'initialisation a échoué pour une raison non définie.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,
                    ComponentData, &Version, &CompCode,
                    &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Demande du service d'autorisation

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_5 et est démarrée par le gestionnaire de files d'attente pour interroger la fonctionnalité prise en charge.

Lorsque plusieurs composants de service sont utilisés, les composants de service sont appelés dans l'ordre inverse de l'ordre dans lequel ils ont été installés.

L'identificateur de cette fonction (pour MQZEP) est MQZID_INQUIRE.

Syntaxe

MQZ_INQUIRE(*QMgrName*, *SelectorCount* , *Selectors* , *IntAttrCount* , *IntAttrs* , *CharAttrLength* , *CharAttrs* , *SelectorReturned* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

SelectorCount
Type : MQLONG - entrée

Nombre de sélecteurs. Nombre de sélecteurs fournis dans le paramètre *Selectors* .

La valeur doit être comprise entre 0 et 256.

Sélecteurs
Type: MQLONGxSelectorNombre-entrée

Tableau de sélecteurs. Chaque sélecteur identifie un attribut obligatoire et doit être l'un des suivants:

- MQIACF_INTERFACE_VERSION (entier)
- MQIACF_USER_ID_SUPPORT (entier)

- MQCACF_SERVICE_COMPONENT (caractère)

Les sélecteurs peuvent être spécifiés dans n'importe quel ordre. Le nombre de sélecteurs dans le tableau est indiqué par le paramètre *SelectorCount* .

Les attributs entiers identifiés par les sélecteurs sont renvoyés dans le paramètre *IntAttrs* dans l'ordre dans lequel ils apparaissent dans *Selectors* .

Les attributs de caractères identifiés par les sélecteurs sont renvoyés dans le paramètre *CharAttrs* dans l'ordre dans lequel ils apparaissent *Selectors* .

IntAttrCount
Type : MQLONG - entrée

Nombre d'attributs de type entier fournis dans le paramètre IntAttrs .

La valeur doit être comprise entre 0 et 256.

IntAttrs
Type: MQLONG ×IntAttrCount-output

Attributs de type entier. Tableau d'attributs de type entier. Les attributs d'entier sont renvoyés dans le même ordre que les sélecteurs d'entier correspondants dans le tableau *Selectors* .

CharAttr
Type : MQLONG - entrée

Longueur de la mémoire tampon des attributs de caractères. Longueur en octets du paramètre *CharAttrs* .

La valeur doit être au moins égale à la somme des longueurs des attributs de caractères demandés. Si aucun attribut de caractère n'est demandé, zéro est une valeur valide.

CharAttrs
Type: MQLONG ×CharAttrCount-output

Mémoire tampon des attributs de caractères. Mémoire tampon contenant des attributs de caractères, concaténés ensemble. Les attributs de caractères sont renvoyés dans le même ordre que les sélecteurs de caractères correspondants dans le tableau *Selectors* .

La longueur de la mémoire tampon est indiquée par le paramètre CharAttrCount.

SelectorReturned
Type: MQLONG ×SelectorCount -entrée

Sélecteur renvoyé. Tableau de valeurs identifiant les attributs qui ont été renvoyés à partir de l'ensemble demandé par les sélecteurs dans le paramètre Sélecteurs. Le nombre de valeurs de ce tableau est indiqué par le paramètre *SelectorCount* . Chaque valeur du tableau est liée au sélecteur à partir de la position correspondante dans le tableau Sélecteurs. Chaque valeur est l'une des suivantes:

MQZSL_RENVOYE
L'attribut demandé par le sélecteur correspondant dans le paramètre *Selectors* a été renvoyé.

MQZSL_NOT_RETURNED
L'attribut demandé par le sélecteur correspondant dans le paramètre *Selectors* n'a pas été renvoyé.

Le tableau est initialisé avec toutes les valeurs *MQZSL_NOT_RETURNED*. Lorsqu'un composant de service d'autorisation renvoie un attribut, il définit la valeur appropriée dans le tableau sur *MQZSL_NOT_RETURNED* . Cela permet à tout autre composant de service d'autorisation, auquel l'appel d'interrogation est envoyé, d'identifier les attributs qui ont déjà été renvoyés.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des

fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Achèvement partiel.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode* .

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
Espace insuffisant pour les attributs de caractères.

MQRC_INT_COUNT_TOO_SMALL
Espace insuffisant pour les attributs de type entier.

Si *CompCode* est MQCC_FAILED:

MQRC_SELECTOR_COUNT_ERREUR
Le nombre de sélecteurs n'est pas valide.

MQRC_SELECTOR_ERREUR
Sélecteur d'attribut non valide.

MQRC_SELECTOR_LIMIT_EXCEEDED
Trop de sélecteurs spécifiés.

MQRC_INT_ATTR_COUNT_ERROR
Le nombre d'attributs de type entier n'est pas valide.

MQRC_INT_ATTRS_ARRAY_ERREUR
Tableau d'attributs d'entier non valide.

MQRC_CHAR_ATTR_LENGTH_ERROR
Le nombre d'attributs de caractères n'est pas valide.

MQRC_CHAR_ATTRS_ERREUR
La chaîne d'attributs de caractères n'est pas valide.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_INQUIRE (QMgrName, SelectorCount, Selectors, IntAttrCount,
                  &IntAttrs, CharAttrLength, &CharAttrs,
                  SelectorReturned, ComponentData, &Continuation,
                  &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Actualise toutes les autorisations

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_3 et est appelée par le gestionnaire de files d'attente pour actualiser la liste des autorisations détenues en interne par le composant.

L'identificateur de cette fonction (pour MQZEP) est MQZID_REFRESH_CACHE (8L).

Syntaxe

MQZ_REFRESH_CACHE(*QMgrName*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors du prochain appel de l'une des fonctions de ce composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_WARNING:

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

Appel C

```
MQZ_REFRESH_CACHE (QMgrName, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
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-Définition des droits

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_1 et est démarrée par le gestionnaire de files d'attente pour définir les droits d'accès d'une entité à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID_SET_AUTHORITY.

Remarque : Cette fonction remplace les droits existants. Pour conserver les droits existants, vous devez les définir à nouveau à l'aide de cette fonction.

Syntaxe

MQZ_SET_AUTHORITY(*QMgrName*, *EntityName*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityName
Type: MQCHAR12 -entrée

Nom de l'entité. Nom de l'entité pour laquelle l'accès à l'objet doit être extrait. La longueur maximale de la chaîne est de 12 caractères ; si elle est plus courte, elle est remplie à droite avec des blancs. Le nom ne se termine pas par un caractère NULL.

EntityType
Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityName*. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL
Entité.

GROUPE MQZAET_GROUP
.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet auquel l'accès est requis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

authority
Type : MQLONG - entrée

Autorité de l'entité. Si un droit est défini, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO_ *). Si plusieurs droits sont définis, cette zone correspond au OU bit à bit des constantes MQZAO_ * correspondantes.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_GET_AUTHORITY, cela a le même effet que MQZCI_CONTINUE.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_SET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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 -Définition des droits (étendu)

Cette fonction est fournie par un composant de service d'autorisation MQZAS_VERSION_2 et est démarrée par le gestionnaire de files d'attente pour définir les droits d'accès d'une entité à l'objet spécifié.

L'identificateur de cette fonction (pour MQZEP) est MQZID_SET_AUTHORITY.

Remarque : Cette fonction remplace les droits existants. Pour conserver les droits existants, vous devez les définir à nouveau à l'aide de cette fonction.

MQZ_SET_AUTHORITY_2 est similaire à MQZ_SET_AUTHORITY, mais avec le paramètre *EntityName* remplacé par le paramètre *EntityData* .

Syntaxe

MQZ_SET_AUTHORITY_2(*QMgrName*, *EntityData*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

EntityData
Type: MQZED-entrée

Données d'entité. Données relatives à l'entité dont l'autorisation sur l'objet doit être définie. Voir «MQZED-Descripteur d'entité», à la page 1221 pour plus de détails.

EntityType
Type : MQLONG - entrée

Type d'entité. Type d'entité spécifié par *EntityData*. Il doit s'agir de l'une des valeurs suivantes:

MQZAET_PRINCIPAL
Entité.

GROUPE MQZAET_GROUP
.

ObjectName
Type: MQCHAR48 -entrée

Nom de l'objet. Nom de l'objet sur lequel les droits d'entité doivent être définis. La longueur maximale de la chaîne est de 48 caractères ; si elle est plus courte, elle est complétée à droite par des blancs. Le nom ne se termine pas par un caractère NULL.

Si *ObjectType* est MQOT_Q_MGR, ce nom est identique à *QMgrName*.

ObjectType
Type : MQLONG - entrée

Types d'objet. Type d'entité spécifié par *ObjectName*. Il doit s'agir de l'une des valeurs suivantes:

INFO MQOT_AUTH_INFO
Informations d'authentification.

CANAL_MQTON
Canal.

MQOT_CLNTCONN_CHANNEL
Canal de connexion client.

MQOT_PROGRAMME d'écoute
Programme d'écoute.

MQOT_NAMELIST
NAMELIST.

PROCESSUS MQ
Définition de processus.

MQOT_Q
File d'attente.

MQOT_Q_DIR
Gestionnaire de files d'attente.

SERVICE MQOT
Service.

MQOT_TOPIC
:NONE.

authority
Type : MQLONG - entrée

Autorité de l'entité. Si un droit est défini, cette zone est égale à l'opération d'autorisation appropriée (constante MQZAO_ *). Si plusieurs droits sont définis, cette zone correspond au OU bit à bit des constantes MQZAO_ * correspondantes.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Les valeurs suivantes peuvent être spécifiées :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

Pour MQZ_CHECK_AUTHORITY, cela a le même effet que MQZCI_STOP.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Non autorisé pour l'accès.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entité inconnue du service.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_SET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                     ObjectType, Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Arrêt du service d'autorisation

Cette fonction est fournie par un composant de service d'autorisation et est démarrée par le gestionnaire de files d'attente lorsqu'il n'a plus besoin des services de ce composant. La fonction doit effectuer tout nettoyage requis par le composant.

L'identificateur de cette fonction (pour MQZEP) est MQZID_TERM_AUTHORITY.

Syntaxe

MQZ_TERM_AUTHORITY(*Hconfig*, *Options*, *QMgrName*, *ComponentData*, *CompCode*, *Reason*)

Paramètres

Configuration Hconfig
Type: MQHCONFIG-entrée

Descripteur de configuration. Cette poignée représente le composant particulier en cours d'arrêt. Il doit être utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

Options
Type : MQLONG - entrée

Options de fin. Il doit s'agir de l'une des valeurs suivantes:

MQZTO_PRIMAIRE
Arrêt principal.

MQZTO_SECONDAIRE
Arrêt secondaire.

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ_INIT_AUTHORITY.

Lorsque l'appel MQZ_TERM_AUTHORITY est terminé, le gestionnaire de files d'attente supprime ces données.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Echec de la commande MQRC_TERMINATION_FAILED
(2287, X'8FF') L'arrêt a échoué pour une raison non définie.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_TERM_AUTHORITY (Hconfig, Options, QMgrName, ComponentData,
                    &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Nom de suppression

Cette fonction est fournie par un composant de service annuaire et démarrée par le gestionnaire de files d'attente pour supprimer une entrée de la file d'attente spécifiée.

L'identificateur de cette fonction (pour MQZEP) est MQZID_DELETE_NAME.

Syntaxe

MQZ_DELETE_NAME(*QMgrName*, *QName*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

nomQ
Type: MQCHAR48 -entrée

Nom de file d'attente. Nom de la file d'attente pour laquelle un poste doit être supprimé. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre de longueur ComponentDatade l'appel MQZ_INIT_NAME.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Il doit s'agir de l'une des valeurs suivantes:

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

MQZCI_ARRET
Ne passez pas au composant suivant.

Pour la commande **MQZ_DELETE_NAME** , le gestionnaire de files d'attente ne tente pas de démarrer un autre composant, peu importe ce qui est renvoyé dans le paramètre **Continuation** .

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_WARNING
Avertissement (achèvement partiel).

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_WARNING:

MQRC_UNKNOWN_NAME
(2288, X'8F0') Nom de file d'attente introuvable.

Remarque : Il se peut qu'il ne soit pas possible de renvoyer ce code si le service sous-jacent répond avec succès dans ce cas.

Si *CompCode* est MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_DELETE_NAME (QMgrName, QName, ComponentData, &Continuation,
                 &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Initialisation du service annuaire

Cette fonction est fournie par un composant de service annuaire et est démarrée par le gestionnaire de files d'attente lors de la configuration du composant. Il est prévu d'appeler MQZEP afin de fournir des informations au gestionnaire de files d'attente.

L'identificateur de cette fonction (pour MQZEP) est MQZID_INIT_NAME.

Syntaxe

MQZ_INIT_NAME(*Hconfig*, *Options*, *QMgrName*, *ComponentDataLength*, *ComponentData*, *Version*, *CompCode*, *Reason*)

Paramètres

Configuration Hconfig
Type: MQHCONFIG-entrée

Descripteur de configuration. Cette poignée représente le composant particulier en cours d'initialisation. Il doit être utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

Options
Type : MQLONG - entrée

Options d'initialisation. Il doit s'agir de l'une des valeurs suivantes:

MQZIO_PRIMAIRE
Initialisation principale.

MQZIO_SECONDAIRE
Initialisation secondaire.

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

ComponentDataLongueur
Type : MQLONG - entrée

Longueur des données de composant. Longueur en octets de la zone *ComponentData* . Cette longueur est définie dans les données de configuration du composant.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Il est initialisé à zéro avant d'appeler la fonction d'initialisation principale du composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_AUTHORITY.

version
Type: MQLONG-entrée / sortie

Numéro de version. En entrée de la fonction d'initialisation, identifie le numéro de version le plus élevé pris en charge par le gestionnaire de files d'attente. La fonction d'initialisation doit modifier, si nécessaire, la version de l'interface qu'elle prend en charge. Si le gestionnaire de files d'attente ne prend pas en charge la version renvoyée par le composant, il appelle la fonction MQZ_TERM_NAME du composant et n'utilise plus ce composant.

Les valeurs suivantes sont prises en charge :

MQZAS_VERSION_1
Version 1.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_INITIALIZATION_FAILED
(2286, X'8EE') L'initialisation a échoué pour une raison non définie.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_INIT_NAME (Hconfig, Options, QMgrName, ComponentDataLength,
               ComponentData, &Version, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Nom d'insertion

Cette fonction est fournie par un composant de service annuaire et est démarrée par le gestionnaire de files d'attente pour insérer une entrée pour la file d'attente spécifiée, contenant le nom du gestionnaire de files d'attente propriétaire de la file d'attente. Si la file d'attente est déjà définie dans le service, l'appel échoue.

L'identificateur de cette fonction (pour MQZEP) est MQZID_INSERT_NAME.

Syntaxe

MQZ_INSERT_NAME(*QMgrName*, *QName*, *ResolvedQMgrName*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

nomQ

Type: MQCHAR48 -entrée

Nom de file d'attente. Nom de la file d'attente pour laquelle un poste doit être inséré. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

ResolvedQMgrNom

Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente résolu. Nom du gestionnaire de files d'attente dans lequel la file d'attente est résolue. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

ComponentData

Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_NAME.

Continuation

Type: MQLONG-entrée / sortie

Indicateur de continuation défini par composant. Pour MQZ_INSERT_NAME, le gestionnaire de files d'attente ne tente pas de démarrer un autre composant, quel qu'il soit renvoyé dans le paramètre *Continuation* .

Les valeurs suivantes sont prises en charge :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_Q_ALREADY_EXISTS
(2290, X'8F2') L'objet de file d'attente existe déjà.

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_INSERT_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,
                 &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Nom de recherche

Cette fonction est fournie par un composant de service annuaire et démarrée par le gestionnaire de files d'attente pour extraire le nom du gestionnaire de files d'attente propriétaire, pour une file d'attente spécifiée.

L'identificateur de cette fonction (pour MQZEP) est MQZID_LOOKUP_NAME.

Syntaxe

MQZ_LOOKUP_NAME(*QMgrName*, *QName*, *ResolvedQMgrName*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

Paramètres

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

nomQ
Type: MQCHAR48 -entrée

Nom de file d'attente. Nom de la file d'attente pour laquelle un poste doit être résolu. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

ResolvedQMgrNom
Type: MQCHAR48 -sortie

Nom du gestionnaire de files d'attente résolu. Si la fonction aboutit, il s'agit du nom du gestionnaire de files d'attente propriétaire de la file d'attente.

Le nom renvoyé par le composant de service doit être rempli à droite avec des blancs sur toute la longueur du paramètre ; le nom ne doit pas se terminer par un caractère NULL ou contenir des blancs de début ou imbriqués.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_NAME.

Continuation
Type : MQLONG - sortie

Indicateur de continuation défini par composant. Pour MQZ_LOOKUP_NAME, le gestionnaire de files d'attente indique s'il doit démarrer un autre composant de service annuaire, comme suit:

- Si *CompCode* est MQCC_OK, aucun autre composant n'est démarré, quelle que soit la valeur renvoyée dans *Continuation*.
- Si *CompCode* n'est pas MQCC_OK, un autre composant est démarré, sauf si *Continuation* est MQZCI_STOP.

Les valeurs suivantes sont prises en charge :

MQZCI_DEFAULT
Continuation dépendant du gestionnaire de files d'attente.

MQZCI_CONTINUER
Passez au composant suivant.

MQZCI_ARRET
Ne passez pas au composant suivant.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Une erreur inattendue s'est produite lors de l'accès au service.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

MQRC_UNKNOWN_NOM_QQ
(2288, X'8F0') Nom de file d'attente introuvable.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_LOOKUP_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,
                 &Continuation, &CompCode, &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Arrêt du service de nom

Cette fonction est fournie par un composant de service annuaire et est démarrée par le gestionnaire de files d'attente lorsqu'il n'a plus besoin des services de ce composant. La fonction doit effectuer tout nettoyage requis par le composant.

L'identificateur de cette fonction (pour MQZEP) est MQZID_TERM_NAME.

Syntaxe

MQZ_TERM_NAME(*Hconfig*, *Options*, *QMgrName*, *ComponentData*, *CompCode*, *Reason*)

Paramètres

Configuration Hconfig
Type: MQHCONFIG-entrée

Descripteur de configuration. Cette poignée représente le composant particulier en cours d'arrêt. Il est utilisé par le composant lors de l'appel du gestionnaire de files d'attente avec la fonction MQZEP.

Options
Type : MQLONG - entrée

Options de fin. Il doit s'agir de l'une des valeurs suivantes:

MQZTO_PRIMAIRE
Arrêt principal.

MQZTO_SECONDAIRE
Arrêt secondaire.

QMgrName
Type: MQCHAR48 -entrée

Nom du gestionnaire de files d'attente Nom du gestionnaire de files d'attente appelant le composant. Ce nom est rempli avec des blancs sur toute la longueur du paramètre ; il ne se termine pas par un caractère indéfini.

Le nom du gestionnaire de files d'attente est transmis au composant pour information ; l'interface du service d'autorisation ne requiert pas que le composant l'utilise de manière définie.

ComponentData
Type: MQBYTE ×ComponentDataLongueur-entrée/sortie

Données de composant. Ces données sont conservées par le gestionnaire de files d'attente pour le compte de ce composant particulier ; toutes les modifications qui lui sont apportées par l'une des fonctions (y compris la fonction d'initialisation) fournies par ce composant sont conservées et présentées lors de l'appel suivant de l'une de ces fonctions de composant.

Les données de composant sont dans la mémoire partagée accessible à tous les processus.

La longueur de cette zone de données est transmise par le gestionnaire de files d'attente dans le paramètre *ComponentDataLength* de l'appel MQZ_INIT_NAME.

Lorsque l'appel MQZ_TERM_NAME est terminé, le gestionnaire de files d'attente supprime ces données.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode*.

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

Echec de la commande MQRC_TERMINATION_FAILED
(2287, X'8FF') L'arrêt a échoué pour une raison non définie.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Service sous-jacent non disponible.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZ_TERM_NAME (Hconfig, Options, QMgrName, ComponentData, &CompCode,
               &Reason);
```

Les paramètres transmis au service sont déclarés comme suit:

```
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-Contexte d'application

La structure MQZAC est utilisée dans l'appel MQZ_AUTHENTICATE_USER pour le paramètre *ApplicationContext* . Ce paramètre spécifie les données relatives à l'application appelante.

Le tableau 1 récapitule les zones de la structure.

Tableau 596. Zones dans MQZAC

Zone	Description
StrucId	Identificateur de structure
Version	Numéro de version de structure
ProcessId	Identificateur de processus
ThreadId	Identificateur d'unité d'exécution
ApplName	Nom d'application
UserID	Identificateur utilisateur
IDEffectiveUser	ID utilisateur effectif
Environnement	Environnement
CallerType	Type d'appelant
AuthenticationType	Type d'authentification
BindType	Type de liaison

Zones

StrucId
Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

ID_STRUC_MQZAC
Identificateur de la structure de contexte d'application.

Pour le langage de programmation C, la constante MQZAC_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQZAC_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

version
Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

MQZAC_VERSION_1
Structure de contexte d'application Version-1 . La constante MQZAC_CURRENT_VERSION indique le numéro de version de la version en cours.

ProcessId
Type: MQPID-entrée

Identificateur de processus de l'application.

ThreadId
Type: MQTID-entrée

Identificateur d'unité d'exécution de l'application.

ApplName
Type: MQCHAR28 -entrée

Nom de l'application.

UserID
Type: MQCHAR12 -entrée

ID utilisateur. Sur les systèmes UNIX , cette zone indique l'ID utilisateur réel de l'application. Sous Windows , cette zone indique l'ID utilisateur de l'application.

IDEffectiveUser
Type: MQCHAR12 -entrée

Identificateur utilisateur effectif. Sur les systèmes UNIX , cette zone indique l'ID utilisateur effectif de l'application. Sous Windows , cette zone est vide.

environnement
Type : MQLONG - entrée

Environnement. Cette zone indique l'environnement à partir duquel l'appel a été effectué. La zone est l'une des valeurs suivantes:

SERVEUR_COMMAND_MQ
Serveur de commandes

MQXE_MQSC
Interpréteur de commandes **runmqsc**

MQXE_MCA
Agent MCA MQXE_OTHER

MQXE_AUTRES
Environnement non défini

CallerType
Type : MQLONG - entrée

Type d'appelant. Cette zone indique le type de programme qui a effectué l'appel. La zone est l'une des valeurs suivantes:

MQXACT_EXTERNAL
L'appel est externe au gestionnaire de files d'attente.

MQXACT_INTERNAL
L'appel est interne au gestionnaire de files d'attente.

AuthenticationType
Type : MQLONG - entrée

Type d'authentification. Cette zone indique le type d'authentification en cours d'exécution. La zone est l'une des valeurs suivantes:

MQZAT_INITIAL_CONTEXT
L'appel d'authentification est dû à l'initialisation du contexte utilisateur. Cette valeur est utilisée lors d'un appel MQCONN ou MQCONNX.

MQZAT_CHANGE_CONTEXT
L'appel d'authentification est dû à la modification du contexte de l'utilisateur. Cette valeur est utilisée lorsque l'agent MCA modifie le contexte utilisateur. Rubrique parent: MQZAC-

BindType
Type : MQLONG - entrée

Type de liaison. Cette zone indique le type de liaison en cours d'utilisation. La zone est l'une des valeurs suivantes:

MQCNO_FASTPATH_BINDING
Liaison Fastpath.

MQCNO_LIEN_PARTAGE
Liaison partagée.

MQCNO_LIEN_ISOLÉ_LIAISON
Liaison isolée.

Déclaration C

Déclarez les zones de la structure comme suit:

```
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-Données de droits d'accès

La structure MQZAD est utilisée dans l'appel MQZ_ENUMERATE_AUTHORITY_DATA pour deux paramètres, une entrée et une sortie.

- MQZAD est utilisé pour le paramètre *Filter* qui est l'entrée de l'appel. Ce paramètre indique les critères de sélection à utiliser pour sélectionner les données de droits d'accès renvoyées par l'appel.
- MQZAD est également utilisé pour le paramètre *AuthorityBuffer* qui est généré à partir de l'appel. Ce paramètre indique les autorisations pour une combinaison de nom de profil, de type d'objet et d'entité.

Tableau 1. récapitule les zones de la structure.

Tableau 597. Zones dans MQZAD

Zone	Description
StrucId	Identificateur de structure
Version	Numéro de version de structure
ProfileName	Identificateur de processus
ObjectType	Identificateur d'unité d'exécution
Droits d'accès	Nom d'application
EntityDataPtr	Identificateur utilisateur
EntityType	Environnement

Tableau 597. Zones dans MQZAD (suite)	
Zone	**Description**
Options	Type d'appelant

Zones

StrucId
Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

ID_STRUC_MQZAC
Identificateur de la structure de contexte d'application.

Pour le langage de programmation C, la constante MQZAC_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQZAC_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

version
Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

MQZAC_VERSION_1
Structure de contexte d'application Version-1 . La constante MQZAC_CURRENT_VERSION indique le numéro de version de la version en cours.

La constante suivante indique le numéro de version de la version en cours:

MQZAD_CURRENT_VERSION
Version actuelle de la structure de données de droits d'accès.

ProfileName
Type: MQCHAR48 -entrée

Nom du profil.

Pour le paramètre *Filtre* , cette zone correspond au nom de profil pour lequel des données de droits d'accès sont requises. Si le nom est entièrement à blanc jusqu'à la fin de la zone ou jusqu'au premier caractère NULL, les données de droits pour tous les noms de profil sont renvoyées.

Pour le paramètre *AuthorityBuffer* , cette zone est le nom d'un profil qui correspond aux critères de sélection spécifiés.

ObjectType
Type : MQLONG - entrée

Types d'objet.

Pour le paramètre *Filtre* , cette zone correspond au type d'objet pour lequel des données de droits d'accès sont requises. Si la valeur est MQOT_ALL, les données de droits d'accès pour tous les types d'objet sont renvoyées.

Pour le paramètre *AuthorityBuffer* , cette zone correspond au type d'objet auquel s'applique le profil identifié par le paramètre *ProfileName* .

La valeur est l'une des suivantes ; pour le paramètre *Filter* , la valeur MQOT_ALL est également valide:

INFO MQOT_AUTH_INFO
Informations d'authentification

CANAL_MQTON
Canal

MQOT_CLNTCONN_CHANNEL
Canal de connexion client

MQOT_PROGRAMME d'écoute
Programme d'écoute

MQOT_NAMELIST
Liste de noms

PROCESSUS MQ
Définition de processus

MQOT_Q
File d'attente

MQOT_Q_DIR
Gestionnaire de files d'attente

SERVICE MQOT
Service

Droits d'accès
Type : MQLONG - entrée

Droits d'accès.

Pour le paramètre *Filtre* , cette zone est ignorée.

Pour le paramètre *AuthorityBuffer* , cette zone représente les autorisations dont dispose l'entité pour les objets identifiés par *ProfileName* et *ObjectType*. Si l'entité ne possède qu'un seul droit d'accès, la zone est égale à la valeur d'autorisation appropriée (constante MQZAO_ *). Si l'entité possède plusieurs droits d'accès, la zone est le OU bit à bit des constantes MQZAO_ * correspondantes.

EntityDataPtr
Type: PMQZED-entrée

Adresse de la structure MQZED identifiant une entité.

Pour le paramètre *Filtre* , cette zone pointe vers une structure MQZED qui identifie l'entité pour laquelle des données de droits d'accès sont requises. Si *EntityDataPtr* est le pointeur null, les données de droits d'accès pour toutes les entités sont renvoyées.

Pour le paramètre *AuthorityBuffer* , cette zone pointe vers une structure MQZED qui identifie l'entité pour laquelle des données de droits d'accès ont été renvoyées.

EntityType
Type : MQLONG - entrée

Type d'entité.

Pour le paramètre *Filtre* , cette zone indique le type d'entité pour lequel des données de droits d'accès sont requises. Si la valeur est MQZAET_NONE, les données de droits d'accès pour tous les types d'entité sont renvoyées.

Pour le paramètre *AuthorityBuffer* , cette zone indique le type de l'entité identifiée par la structure MQZED désignée par le paramètre *EntityDataPtr* .

La valeur est l'une des suivantes ; pour le paramètre *Filtre* , la valeur MQZAET_NONE est également valide:

MQZAET_PRINCIPAL
Principale

GROUPE MQZAET_GROUP
Groupe

Options
Type: MQAUTHOPT-entrée

Options. Cette zone indique les options qui permettent de contrôler les profils affichés. L'une des valeurs suivantes doit être spécifiée:

MQAUTHOPT_NOM_ALL_MATCHING
Affiche tous les profils

MQAUTHOPT_NOM_EXPLICITE
Affiche les profils ayant exactement le même nom que celui spécifié dans la zone *ProfileName* .

En outre, l'un des éléments suivants doit également être spécifié:

MQAUTHOPT_ENTITY_SET
Affiche tous les profils utilisés pour calculer les droits cumulés de l'entité sur l'objet spécifié par le paramètre *ProfileName* . Le paramètre *ProfileName* ne doit pas contenir de caractères génériques.

Si l'entité spécifiée est un principal, pour chaque membre de l'ensemble {entity, groups} , le profil le plus applicable qui s'applique à l'objet est affiché.

Si l'entité spécifiée est un groupe, le profil le plus applicable du groupe qui s'applique à l'objet est affiché.

Si cette valeur est spécifiée, les valeurs de *ProfileName*, *ObjectType*, *EntityType*et le nom d'entité spécifié dans la structure *EntityDataPtr* MQZED ne doivent pas être vides.

Si vous avez spécifié MQAUTHOPT_NAME_ALL_MATCHING, vous pouvez également spécifier la valeur suivante:

MQAUTHOPT_ENTITY_EXPLICIT
Affiche les profils qui ont exactement le même nom d'entité que le nom d'entité spécifié dans la structure *EntityDataPtr* MQZED.

Déclaration C

```
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 */
};
```

Zones

MQZED-Descripteur d'entité

La structure MQZED est utilisée dans un certain nombre d'appels de service d'autorisation pour spécifier l'entité pour laquelle l'autorisation doit être vérifiée.

Tableau 1. récapitule les zones de la structure.

Tableau 598. Zones dans MQZED

Zone	Description
StrucId	Identificateur de structure
Version	Version
EntityName Ptr	Nom d'entité
EntityDomainPtr	Pointeur de domaine d'entité
SecurityId	Identificateur de sécurité
CorrelationPtr	Pointeur de corrélation

Zones

StrucId
Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

ID_STRUC_MQZ
Identificateur de la structure de descripteur d'entité.

Pour le langage de programmation C, la constante MQZED_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQZED_STRUC_ID, mais il s'agit d'un tableau de caractères au lieu d'une chaîne.

version
Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

MQZED_VERSION_1
Structure de descripteur d'entité Version-1 .

La constante suivante indique le numéro de version de la version en cours:

VERSION_MQZED_CURRENT_VERSION
Version actuelle de la structure de descripteur d'entité.

EntityNamePtr
Type: PMQCHAR-input

Nom du profil.

Adresse du nom de l'entité. Il s'agit d'un pointeur vers le nom de l'entité dont l'autorisation doit être vérifiée.

EntityDomainPtr
Type: PMQCHAR-input

Adresse du nom de domaine de l'entité. Il s'agit d'un pointeur vers le nom du domaine contenant la définition de l'entité dont l'autorisation doit être vérifiée.

SecurityId
Type: MQBYTE40 -entrée

Droits d'accès.

Identificateur de sécurité. Il s'agit de l'identificateur de sécurité dont l'autorisation doit être vérifiée.

CorrelationPtr
Type: MQPTR-entrée

Pointeur de corrélation. Cela facilite la transmission de données correlationnelles entre la fonction d'authentification de l'utilisateur et d'autres fonctions OAM appropriées.

Déclaration C

```
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 */
```

Zones

MQZEP-Ajout d'un point d'entrée de composant

Un composant de service démarre cette fonction, lors de l'initialisation, pour ajouter un point d'entrée au vecteur de point d'entrée de ce composant de service.

Syntaxe

MQZEP (*Hconfig*, *Fonction*, *EntryPoint*, *CompCode*, *Motif*)

Paramètres

Configuration Hconfig
Type: MQHCONFIG-entrée

Descripteur de configuration. Ce descripteur représente le composant en cours de configuration pour ce service installable particulier. Il doit être identique au composant transmis à la fonction de configuration de composant par le gestionnaire de files d'attente lors de l'appel d'initialisation de composant.

Fonction
Type : MQLONG - entrée

Identificateur de fonction. Les valeurs valides sont définies pour chaque service installable.

Si MQZEP est appelé plusieurs fois pour la même fonction, le dernier appel effectué fournit le point d'entrée utilisé.

EntryPoint
Type: PMQFUNC-entrée

Point d'entrée de fonction. Il s'agit de l'adresse du point d'entrée fourni par le composant pour exécuter la fonction.

La valeur NULL est valide et indique que la fonction n'est pas fournie par ce composant. La valeur NULL est prise en compte pour les points d'entrée qui ne sont pas définis à l'aide de MQZEP.

CompCode
Type : MQLONG - sortie

Code achèvement. Il doit s'agir de l'une des valeurs suivantes:

MQCC_OK
Achèvement réussi.

MQCC_FAILED
Echec de l'appel.

raison
Type : MQLONG - sortie

Code anomalie qualifiant *CompCode* .

Si *CompCode* est MQCC_OK:

MQRC_NONE
(0, X'000') Aucun code anomalie à signaler.

Si *CompCode* est MQCC_FAILED:

Erreur de fonction MQRC_FUNCTION_ERROR
(2281, X'8E9') Identificateur de fonction incorrect.

ERREUR MQRC_HCONFIG_ERROR
(2280, X'8E8') Descripteur de configuration non valide.

Pour plus d'informations sur ces codes anomalie, voir Codes anomalie d'API.

Appel C

```
MQZEP (Hconfig, Function, EntryPoint, &CompCode, &Reason);
```

Déclarez les paramètres comme suit :

```
MQHCONFIG  Hconfig;     /* Configuration handle */
MQLONG     Function;    /* Function identifier */
PMQFUNC    EntryPoint;  /* Function entry point */
MQLONG     CompCode;    /* Completion code */
MQLONG     Reason;      /* Reason code qualifying CompCode */
```

MQZFP-Paramètres libres

La structure MQZFP est utilisée dans l'appel MQZ_FREE_USER pour le paramètre *FreeParms* . Ce paramètre indique les données relatives à la ressource à libérer.

Tableau 1. récapitule les zones de la structure.

Tableau 599. Zones dans MQZFP

Zone	Description
StrucId	Identificateur de structure
Version	Version
Reserved	Réservé, zone
CorrelationPtr	Pointeur de corrélation

Zones

StrucId
Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

ID_STRUC_MQZ
Identificateur de la structure de contexte d'identité. Pour le langage de programmation C, la constante MQZIC_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQZIC_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

version
Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

MQZFP_VERSION_1
Structure des paramètres libres Version-1 .

La constante suivante indique le numéro de version de la version en cours:

VERSION_MQZFP_CURRENT_VERSION
Version actuelle de la structure des paramètres libres.

Reserved
Type: MQBYTE8 -entrée

Zone réservée. La valeur initiale est null.

CorrelationPtr
Type: MQPTR-entrée

Pointeur de corrélation. Adresse des données de corrélation relatives à la ressource à libérer.

Déclaration C

```
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 */
};
```

Zones

MQZIC-Contexte d'identité

La structure MQZIC est utilisée dans l'appel MQZ_AUTHENTICATE_USER pour le paramètre *IdentityContext* .

La structure MQZIC contient des informations de contexte d'identité, qui identifient l'utilisateur de l'application qui a placé le message dans une file d'attente pour la première fois:

- Le gestionnaire de files d'attente remplit la zone *UserIdentifier* avec un nom qui identifie l'utilisateur, la manière dont le gestionnaire de files d'attente peut effectuer cette opération dépend de l'environnement dans lequel l'application s'exécute.
- Le gestionnaire de files d'attente remplit la zone *AccountingToken* avec un jeton ou un numéro qu'il a déterminé à partir de l'application qui a inséré le message.
- Les applications peuvent utiliser la zone *DonnéesApplIdentity* pour toute information supplémentaire qu'elles souhaitent inclure sur l'utilisateur (par exemple, un mot de passe chiffré).

Les applications disposant des droits appropriés peuvent définir le contexte d'identité à l'aide de la fonction MQZ_AUTHENTICATE_USER.

Un identificateur de sécurité des systèmes Windows est stocké dans la zone *AccountingToken* lorsqu'un message est créé sous WebSphere MQ for Windows. Le SID peut être utilisé pour compléter la zone *UserIdentifier* et pour établir les données d'identification d'un utilisateur.

Tableau 1. récapitule les zones de la structure.

Tableau 600. Zones dans MQZIC

Zone	Description
StrucId	Identificateur de structure
Version	Version
UserIdentifier	Identificateur utilisateur
AccountingToken	Jeton de comptabilité
ApplIdentityData	Données sur l'identité de l'application

Zones

StrucId
Type: MQCHAR4 -entrée

Identificateur de structure. La valeur est la suivante:

ID_STRUC_MQZ
Identificateur de la structure de contexte d'identité. Pour le langage de programmation C, la constante MQZIC_STRUC_ID_ARRAY est également définie ; elle a la même valeur que MQZIC_STRUC_ID, mais il s'agit d'un tableau de caractères à la place d'une chaîne.

version
Type : MQLONG - entrée

Numéro de version de la structure. La valeur est la suivante:

MQZIC_VERSION_1
Structure de contexte d'identité Version-1 .

La constante suivante indique le numéro de version de la version en cours:

MQZIC_CURRENT_VERSION
Version actuelle de la structure de contexte d'identité.

UserIdentifier
Type: MQCHAR12 -entrée

ID utilisateur. Il fait partie du contexte d'identité du message. *UserIdentifier* indique l'ID utilisateur de l'application à l'origine du message. Le gestionnaire de files d'attente traite ces informations comme des données de type caractères, mais ne définit pas leur format. Pour plus d'informations sur la zone *UserIdentifier* , voir «UserIdentifier (MQCHAR12)», à la page 441.

AccountingToken
Type: MQBYTE32 -entrée

Jeton de comptabilité. Il fait partie du contexte d'identité du message. *AccountingToken* permet à une application de provoquer une facturation appropriée du travail effectué à la suite du message. Le gestionnaire de files d'attente traite ces informations comme une chaîne de bits et ne vérifie pas leur contenu. Pour plus d'informations sur la zone *AccountingToken* , voir «AccountingToken (MQBYTE32)», à la page 398.

ApplIdentityData
Type: MQCHAR32 -entrée

Données d'application relatives à l'identité. Il fait partie du contexte d'identité du message. ApplIdentityLes données sont des informations définies par la suite d'applications qui peuvent être utilisées pour fournir des informations supplémentaires sur l'origine du message. Par exemple, elle peut être définie par des applications s'exécutant avec des droits utilisateur appropriés pour indiquer si les données d'identité sont dignes de confiance. Pour plus d'informations sur la zone de données ApplIdentity, voir «Données ApplIdentity(MQCHAR32)», à la page 400.

Déclaration C

```
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 */
};
```

Zones

Informations de référence pour le pont IBM WebSphere MQ pour HTTP

Rubriques de référence pour IBM WebSphere MQ bridge for HTTP, classées par ordre alphabétique

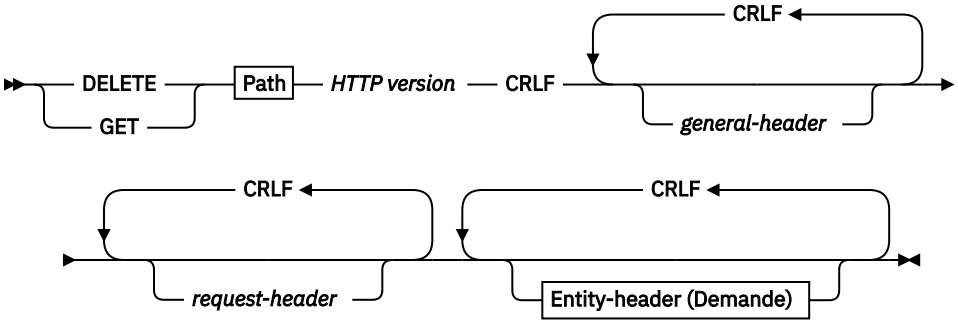
HTTP DELETE: WebSphere MQ Bridge for HTTP command

L'opération HTTP **DELETE** extrait un message d'une file d'attente WebSphere MQ ou extrait une publication d'une rubrique. Le message est supprimé de la file d'attente. Si la publication est conservée,

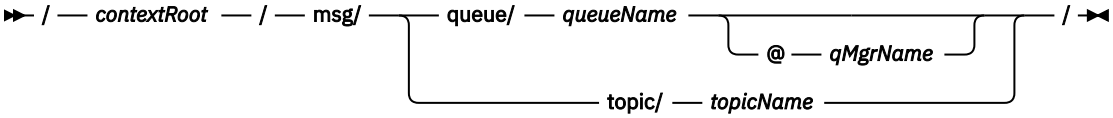
elle n'est pas supprimée. Un message de réponse est renvoyé au client avec des informations sur le message.

Syntaxe

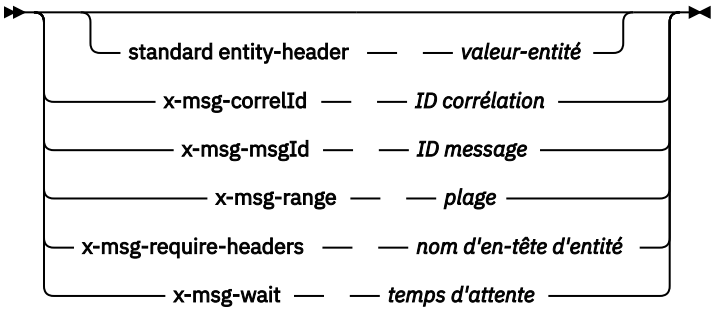
Demande



Path



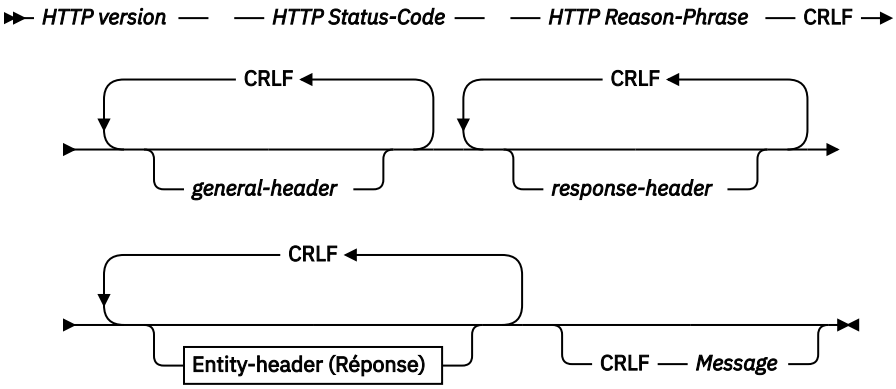
entity-header (Demande)



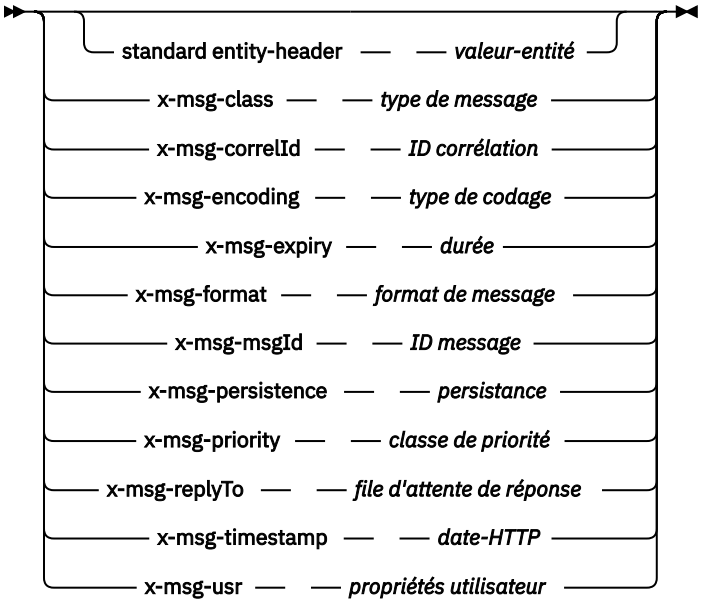
Remarque :

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**

Réponse



en-tête d'entité (réponse)



Paramètres de la demande

Chemin d'accès
Voir «Format d'URI», à la page 1261.

Version HTTP
Version HTTP ; par exemple, HTTP/1.1

général-en-tête
Voir HTTP/1.1 - 4.5 General Header Fields.

en-tête de demande
Voir HTTP/1.1 - 5.3 Request Header Fields. La zone Hôte est obligatoire dans une demande HTTP/1.1 . Il est souvent inséré automatiquement par l'outil que vous utilisez pour créer une demande client.

entity-header (Demande)
Voir HTTP/1.1 - 7.1 Entity Header Fields. L'un des en-têtes d'entité répertoriés dans le diagramme de syntaxe de la demande.

Paramètres de réponse

Chemin d'accès
Voir «Format d'URI», à la page 1261.

Version HTTP
Version HTTP ; par exemple, HTTP/1.1

général-en-tête
Voir HTTP/1.1 - 4.5 General Header Fields.

en-tête de réponse
Voir HTTP/1.1 - 6.2 Response Header Fields.

entity-header (réponse)
Voir HTTP/1.1 - 7.1 Entity Header Fields. L'un des en-têtes d'entité ou de réponse répertoriés dans le diagramme de syntaxe de réponse. La zone `Content-Length` est toujours présente dans une réponse. Elle est définie sur zéro s'il n'y a pas de corps de message.

message
Corps du message.

Description

Si la demande HTTP **DELETE** aboutit, le message de réponse contient les données extraites de la file d'attente WebSphere MQ . Le nombre d'octets dans le corps du message est renvoyé dans l'en-tête HTTP `Content-Length` . Le code de statut de la réponse HTTP est défini sur `200 OK`. Si `x-msg-range` est spécifié en tant que `0`ou `0-0`, le code de statut de la réponse HTTP est `204 No Content`.

Si la demande HTTP **DELETE** échoue, la réponse inclut un pont WebSphere MQ pour le message d'erreur HTTP et un code de statut HTTP.

Exemple HTTP DELETE

HTTP **DELETE** obtient un message d'une file d'attente et supprime le message, ou extrait et supprime une publication. L'exemple **HTTPDELETE** Java est un exemple de demande HTTP **DELETE** qui lit un message à partir d'une file d'attente. Au lieu d'utiliser Java, vous pouvez créer une demande HTTP **DELETE** à l'aide d'un formulaire de navigateur ou d'un kit d'outils AJAX.

Figure 37, à la page 1229 est une demande HTTP de suppression du message suivant dans la file d'attente appelée `myQueue`. En réponse, le corps du message est renvoyé au client. Dans les termes WebSphere MQ , HTTP **DELETE** est un get destructeur.

La demande contient l'en-tête de demande HTTP `x-msg-wait`, qui indique à WebSphere MQ le temps d'attente d'un message dans la file d'attente. Elle contient également l'en-tête de demande `x-msg-require-headers`, qui spécifie que le client doit recevoir l'ID de corrélation du message dans la réponse.

```
DELETE /msg/queue/myQueue/ HTTP/1.1
Host: www.example.org
x-msg-wait: 10
x-msg-require-headers: correlID
```

*Figure 37. Exemple d'une demande HTTP **DELETE***

Figure 38, à la page 1229est la réponse renvoyée au client. L'ID de corrélation est renvoyé au client, comme demandé dans l'en-tête `x-msg-require-headers` de la demande.

```
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.
```

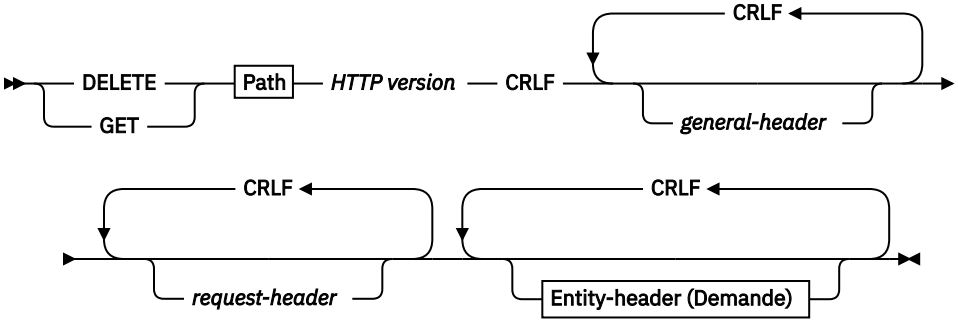
*Figure 38. Exemple de réponse HTTP **DELETE***

HTTP GET: WebSphere MQ Bridge for HTTP command

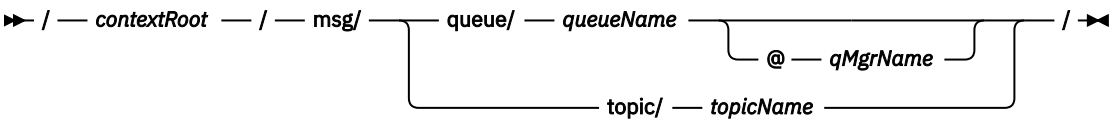
L'opération HTTP **GET** obtient un message d'une file d'attente WebSphere MQ . Le message reste dans la file d'attente. L'opération HTTP **GET** équivaut à parcourir une file d'attente WebSphere MQ .

Syntaxe

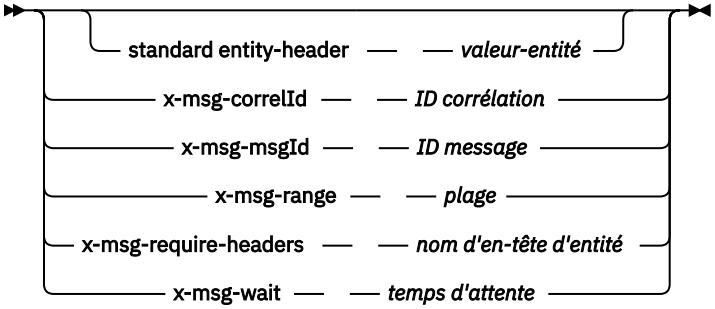
Demande



Path



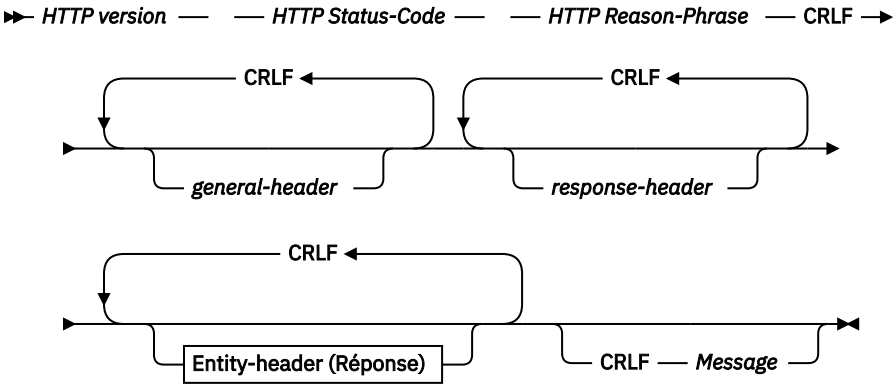
entity-header (Demande)



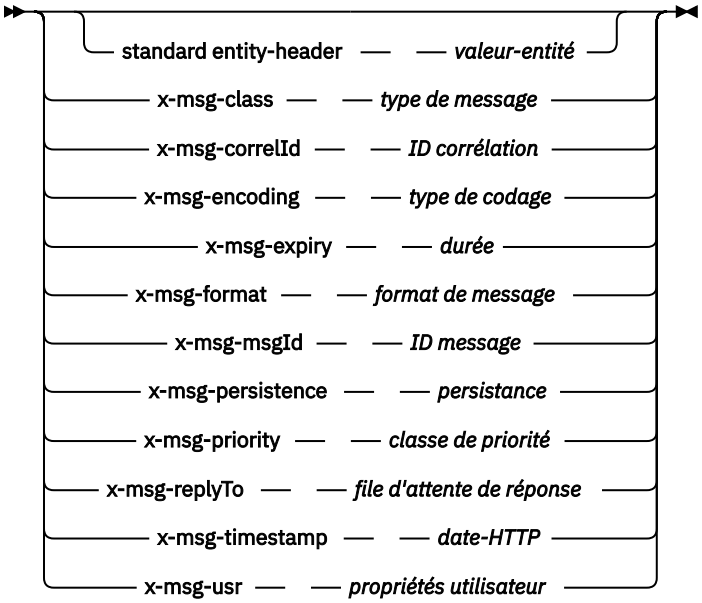
Remarque :

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**

Réponse



en-tête d'entité (réponse)



Paramètres de la demande

Chemin d'accès
Voir «Format d'URI», à la page 1261.

Version HTTP
Version HTTP ; par exemple, HTTP/1.1

général-en-tête
Voir HTTP/1.1 - 4.5 General Header Fields.

en-tête de demande
Voir HTTP/1.1 - 5.3 Request Header Fields. La zone Hôte est obligatoire dans une demande HTTP/1.1 . Il est souvent inséré automatiquement par l'outil que vous utilisez pour créer une demande client.

entity-header (Demande)
Voir HTTP/1.1 - 7.1 Entity Header Fields. L'un des en-têtes d'entité répertoriés dans le diagramme de syntaxe de la demande.

Paramètres de réponse

Chemin d'accès
Voir «Format d'URI», à la page 1261.

Version HTTP
Version HTTP ; par exemple, HTTP/1.1

général-en-tête
Voir HTTP/1.1 - 4.5 General Header Fields.

en-tête de réponse
Voir HTTP/1.1 - 6.2 Response Header Fields.

entity-header (réponse)
Voir HTTP/1.1 - 7.1 Entity Header Fields. L'un des en-têtes d'entité ou de réponse répertoriés dans le diagramme de syntaxe de réponse. La zone `Content-Length` est toujours présente dans une réponse. Elle est définie sur zéro s'il n'y a pas de corps de message.

message
Corps du message.

Description

Si la demande HTTP **GET** aboutit, le message de réponse contient les données extraites de la file d'attente WebSphere MQ . Le nombre d'octets dans le corps du message est renvoyé dans l'en-tête HTTP `Content-Length` . Le code de statut de la réponse HTTP est défini sur `200 OK`. Si `x-msg-range` est spécifié en tant que `0`ou `0-0`, le code de statut de la réponse HTTP est `204 No Content`.

Si la demande HTTP **GET** échoue, la réponse inclut un pont WebSphere MQ pour le message d'erreur HTTP et un code de statut HTTP.

Exemple HTTP GET

HTTP **GET** envoie un message depuis une file d'attente. Le message reste dans la file d'attente. Dans les termes WebSphere MQ , HTTP **GET** est une demande de navigation. Vous pouvez créer une demande HTTP **GET** avec un client Java, un formulaire de navigateur ou un kit d'outils AJAX.

Figure 39, à la page 1232 est une demande HTTP permettant de parcourir le message suivant dans la file d'attente appelée `myQueue`.

La demande contient l'en-tête de demande HTTP `x-msg-wait`, qui indique à WebSphere MQ le temps d'attente d'un message dans la file d'attente. Elle contient également l'en-tête de demande `x-msg-require-headers`, qui spécifie que le client doit recevoir l'ID de corrélation du message dans la réponse.

```
GET /msg/queue/myQueue/ HTTP/1.1
Host: www.example.org
x-msg-wait: 10
x-msg-require-headers: correlID
```

*Figure 39. Exemple de demande HTTP **GET***

Figure 40, à la page 1232 est la réponse renvoyée au client. L'ID de corrélation est renvoyé au client, comme demandé dans l'en-tête `x-msg-require-headers` de la demande.

```
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.
```

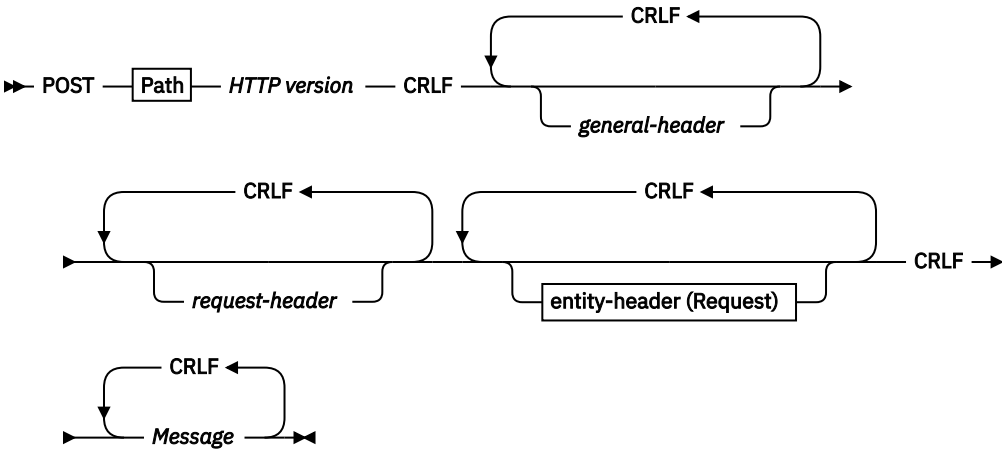
*Figure 40. Exemple de réponse HTTP **GET***

HTTP POST: WebSphere MQ Bridge for HTTP command

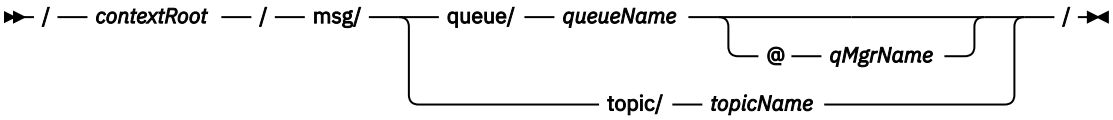
L'opération HTTP **POST** place un message dans une file d'attente WebSphere MQ ou publie un message dans une rubrique.

Syntax

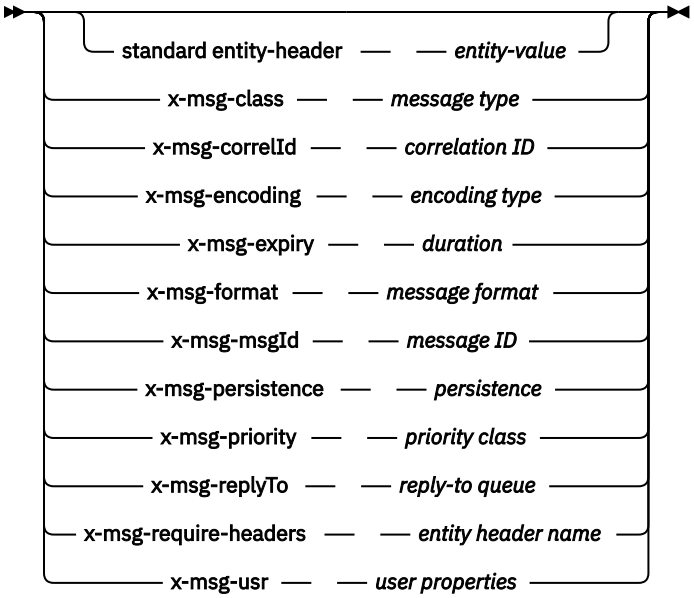
Request



Path



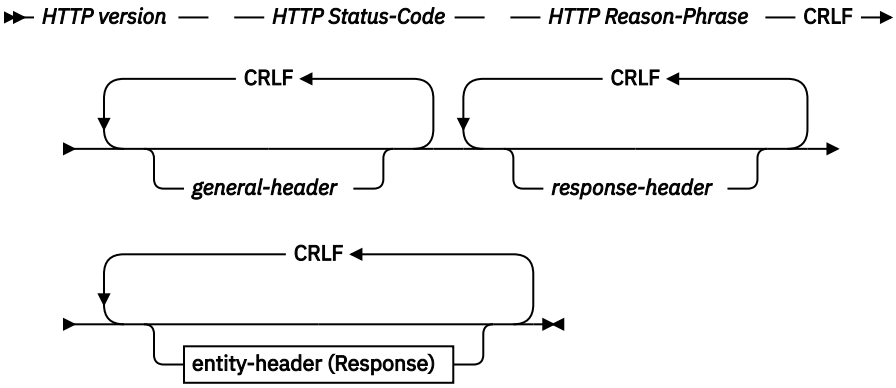
entity-header (Request)



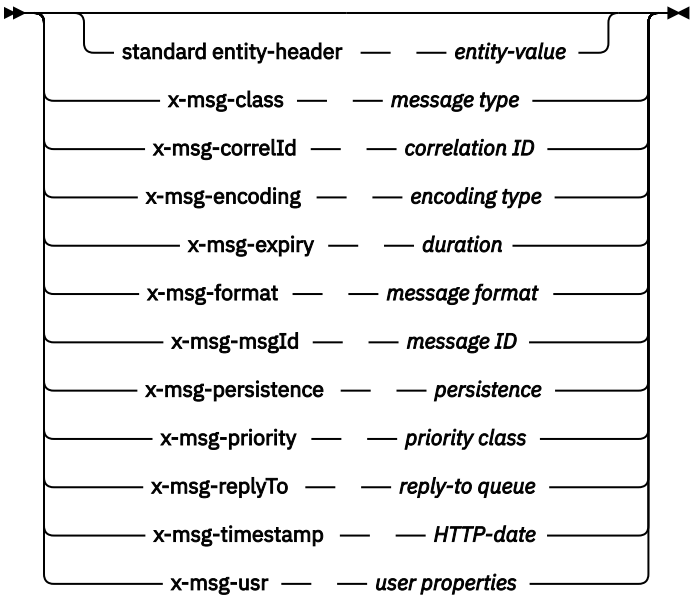
Remarque :

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)



Paramètres de la demande

Chemin d'accès
Voir «Format d'URI», à la page 1261.

Version HTTP
Version HTTP ; par exemple, `HTTP/1.1`

général-en-tête
Voir HTTP/1.1 - 4.5 General Header Fields.

en-tête de demande
Voir HTTP/1.1 - 5.3 Request Header Fields. La zone Hôte est obligatoire dans une demande HTTP/1.1 . Il est souvent inséré automatiquement par l'outil que vous utilisez pour créer une demande client.

entity-header (Demande)
Voir HTTP/1.1 - 7.1 Entity Header Fields. L'un des en-têtes d'entité répertoriés dans le diagramme de syntaxe de la demande. Les éléments `Content-Length` et `Content-Type` doivent être insérés dans une demande et sont souvent insérés automatiquement par l'outil que vous utilisez pour créer une demande client. Le `Content-Type` doit correspondre au type défini dans l'en-tête d'entité personnalisé `x-msg-class` , s'il est spécifié.

message
Message à insérer dans la file d'attente ou publication à publier dans une rubrique.

Paramètres de réponse

Chemin d'accès
Voir «Format d'URI», à la page 1261.

Version HTTP
Version HTTP ; par exemple, `HTTP/1.1`

général-en-tête
Voir HTTP/1.1 - 4.5 General Header Fields.

en-tête de réponse
Voir HTTP/1.1 - 6.2 Response Header Fields.

entity-header (réponse)
Voir HTTP/1.1 - 7.1 Entity Header Fields. L'un des en-têtes d'entité ou de réponse répertoriés dans le diagramme de syntaxe de réponse. La zone `Content-Length` est toujours présente dans une réponse. Elle est définie sur zéro s'il n'y a pas de corps de message.

Description

Si aucun en-tête `x-msg-usr` n'est inclus et que la classe de message est BYTES ou TEXT, le message inséré dans la file d'attente ne comporte pas de `MQRFH2`.

Utilisez l'entité HTTP et les en-têtes de demande dans la demande HTTP **POST** pour définir les propriétés du message inséré dans la file d'attente. Vous pouvez également utiliser `x-msg-require-headers` pour demander quels en-têtes sont renvoyés dans le message de réponse.

Si la demande HTTP **POST** aboutit, l'entité du message de réponse est vide et sa longueur de contenu est égale à zéro. Le code de statut HTTP est `200 OK`.

Si la demande HTTP **POST** échoue, la réponse inclut un pont WebSphere MQ pour le message d'erreur HTTP et un code de statut HTTP. Le message WebSphere MQ n'est pas inséré dans la file d'attente ou la rubrique.

Exemple HTTP POST

HTTP **POST** place un message dans une file d'attente ou une publication sur dans une rubrique. L'exemple Java **HTTPPOST** est un exemple de demande HTTP **POST** d'un message dans une file d'attente. Au lieu d'utiliser Java, vous pouvez créer une requête HTTP **POST** à l'aide d'un formulaire de navigateur ou d'un kit d'outils AJAX.

La Figure 41, à la page 1235 illustre une demande HTTP d'insertion d'un message dans une file d'attente appelée `myQueue`. Cette demande contient l'en-tête HTTP `x-msg-correlId` pour définir l'ID de corrélation du message WebSphere MQ .

```
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.
```

*Figure 41. Exemple de requête HTTP **POST** dans une file d'attente*

Figure 42, à la page 1236 montre la réponse renvoyée au client. Le contenu de la réponse est inexistant.

```
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
```

Figure 42. Exemple de réponse HTTP POST

En-têtes HTTP

Le pont WebSphere MQ pour HTTP prend en charge les en-têtes HTTP de demande personnalisés, les en-têtes HTTP d'entité personnalisés et un sous-ensemble d'en-têtes HTTP standard.

La pratique HTTP consiste à préfixer tous les en-têtes personnalisés avec `x-`, le pont WebSphere MQ pour les en-têtes HTTP est préfixé avec `x-msg-`. Par exemple, pour définir l'en-tête de priorité, utilisez `x-msg-priority`.

Remarque :

- La plupart des valeurs d'en-tête sont sensibles à la casse. Par exemple, lorsque vous utilisez l'en-tête `msgId` , `NONE` est un mot clé, alors que `none` est un `msgID`.
- Les en-têtes mal orthographiés sont ignorés.

En-têtes HTTP d'entité personnalisée

Les en-têtes HTTP d'entité personnalisée contiennent des informations sur les messages WebSphere MQ . A l'aide des en-têtes d'entité, vous pouvez définir des valeurs dans le descripteur de message (`MQMD`) ou des valeurs de requête dans `MQMD`. Un en-tête d'entité supplémentaire, `x-msg-usr`, définit et renvoie les informations de propriété utilisateur que vous souhaitez associer à une demande.

Vous pouvez utiliser des en-têtes d'entité dans différents contextes de demande HTTP:

DELETE
Vous pouvez uniquement utiliser les en-têtes d'entité `x-msg-correlId`et/ou `x-msg-msgId`avec une demande HTTP **DELETE** . Les en-têtes ont pour effet de sélectionner un message particulier par `MsgId` et `CorrelId` dans un `MQGET`, et de supprimer le message de sa file d'attente.

GET
Vous pouvez uniquement utiliser les en-têtes d'entité `x-msg-correlId`et/ou `x-msg-msgId`avec une demande HTTP **GET** . Les en-têtes ont pour effet de sélectionner un message particulier par `MsgId` et `CorrelId` dans un `MQGET` pour l'exploration.

POST
Vous pouvez utiliser n'importe quel en-tête d'entité dans une demande HTTP **POST** , à l'exception de `x-msg-timestamp`.

x-msg-require-headers
Sur toute demande HTTP **GET**, **POST** ou **DELETE** , vous pouvez ajouter plusieurs en-têtes d'entité dans l'en-tête de demande `x-msg-require-headers` , séparés par des virgules. L'effet est de renvoyer les en-têtes d'entité spécifiés dans le message de réponse HTTP, contenant la valeur de la propriété de message associée.

La description de chaque en-tête répertorie les contextes dans lesquels l'en-tête est traité par le pont WebSphere MQ pour HTTP. Par exemple, dans l'en-tête **POST**, `x-msg-require-headers`, l'en-tête est traité par WebSphere MQ bridge for HTTP dans une demande HTTP **POST** , ou dans l'en-tête de demande `x-msg-require-headers` dans une demande HTTP **POST**, **GET**ou **DELETE** . Si l'en-tête est inclus dans un contexte dans lequel il n'est pas autorisé, il est ignoré. Aucune erreur n'est signalée.

Vous pouvez placer des en-têtes HTTP standard dans des demandes à traiter par le serveur Web ou d'autres gestionnaires de demandes. De même, la réponse peut contenir d'autres en-têtes HTTP standard insérés par le serveur Web ou d'autres gestionnaires de réponse.

En-têtes HTTP de demande personnalisée

Les trois en-têtes HTTP de demande personnalisée, `x-msg-range`, `x-msg-require-headerset` `x-msg-wait`, transmettent des informations supplémentaires sur la demande HTTP au serveur. Ils agissent en tant que modificateurs de demande. Avec `x-msg-range`, vous pouvez limiter la quantité de données de message renvoyées dans une réponse. Avec `x-msg-require-headers`, vous pouvez demander à la réponse de contenir des informations sur le résultat de la demande. Avec `x-msg-wait`, vous pouvez modifier le temps pendant lequel le client attend une réponse HTTP.

En-têtes HTTP standard

L'en-tête de demande HTTP standard `Host` doit être spécifié dans une demande HTTP/1.1 .

Les en-têtes d'entité HTTP standard `Content-Length` et `Content-Type` peuvent être spécifiés dans une demande.

Les en-têtes d'entité HTTP standard `Content-Length`, `Content-Location`, `Content-Range`, `Content-Typeet Server` peuvent être renvoyés en réponse à une demande. Spécifiez un ou plusieurs des en-têtes HTTP standard dans l'en-tête `x-msg-request-header` du message de demande.

Liste alphabétique des en-têtes HTTP

`class`: HTTP `x-msg-class` entity-header

Définissez ou renvoyez le type de message.

Type	Description
Nom d'en-tête HTTP	`classe-msg-x`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	POST, `x-msg-require-headers`
Valeurs autorisées	**BYTES** **MAP** **STREAM** **TEXT**
Valeur par défaut	BYTES

Description

- Dans une demande HTTP **POST** , définit le type du message créé.
- La spécification de l'en-tête de classe sur un **GET** ou **DELETE** renvoie un `400 Bad Request` avec le corps d'entité `MQHTTP40007`.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-class` dans le message de réponse HTTP sur le type d'un message.
- Si une valeur non valide est spécifiée pour cet en-tête, un message `MQHTTP40005` est renvoyé.
- Si l'en-tête `x-msg-class` n'est pas spécifié et que le type de contenu du message est `application/x-www-form-urlencoded`, les données sont supposées être un objet de mappe JMS.

`Content-Length`: en-tête d'entité HTTP

Définissez ou renvoyez la longueur, en octets, du corps du message.

Type	Description
Nom d'en-tête HTTP	`Longueur de contenu`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	`x-msg-require-headers`
Valeur autorisée et renvoyée	***`Integer value`*** Longueur en octets du corps du message.

Description

- `Content-Length` est facultatif dans une demande HTTP. Pour un **GET** ou un **DELETE** , la longueur doit être égale à zéro. Pour **POST**, si `Content-Length` est spécifié et qu'il ne correspond pas à la longueur de la ligne de message, le message est tronqué ou rempli avec des valeurs nulles à la longueur spécifiée.
- La valeur `Content-Length` est toujours renvoyée dans la réponse HTTP même en l'absence de contenu, auquel cas la valeur est zéro.

`Content-Location`: en-tête d'entité HTTP

Renvoie la file d'attente ou la rubrique référencée dans la demande, dans l'en-tête `Content-Location` standard du message de réponse HTTP.

Type	Description
Nom d'en-tête HTTP	`Emplacement de contenu`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	`x-msg-require-headers`

Valeur renvoyée	URI au format,
	`/msg/queue/queuename`
	ou
	`/msg/topic/topicname`

Description

- Lorsqu'il est demandé dans `x-msg-require-headers`, l'en-tête d'entité `Content-Location` renvoie la file d'attente ou la rubrique référencée dans la demande HTTP.

`Content-Range`: en-tête d'entité HTTP

Renvoie la plage d'octets sélectionnée dans un message WebSphere MQ dans l'en-tête `Content-Range` d'une réponse HTTP.

Type	Description
Nom d'en-tête HTTP	`Plage de contenu`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	`x-msg-require-headers`

Type	Description
Valeur renvoyée	***String*** Renvoie la limite inférieure, *m* et la limite supérieure, *n* de la sous-chaîne renvoyée et *length* de l'ensemble du message. Par exemple : `m-n/length`

Description

•

- `Content-Range` est renvoyé uniquement dans la réponse HTTP lorsque `Content-Range` est spécifié dans une demande **GET** ou **DELETE** qui contient un en-tête de demande `x-msg-range` .
- Si `x-msg-range` est spécifié dans une demande **GET** ou **DELETE** , la plage d'octets spécifiée dans l'en-tête Content-Range est renvoyée dans la réponse. Par exemple, si `x-msg-range: 0-60` est utilisé dans une demande pour un message contenant `100` octets, l'en-tête content-range contient la chaîne `0-60/100`
- Une demande `x-msg-range` renvoie également la plage de contenu dans l'en-tête `x-msg-range` de la réponse HTTP.

`Content-Type`: en-tête d'entité HTTP

Définissez ou renvoyez la classe du message JMS dans un message WebSphere MQ en fonction du type de contenu HTTP.

Type	Description
Nom d'en-tête HTTP	`Content-Type`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`
Valeur autorisée ou renvoyée	***media-type*** Pour les types de support pris en charge, voir Tableau 601, à la page 1239.

Tableau 601. Mappage entre `x-msg-class` et HTTP `Content-Type`

classe-msg-x	**HTTP Content-Type**
BYTES	`application/octet-stream` `application/xml`
TEXT	`text/*`
MAP	`application/x-www-form-urlencoded` `application/xml` (facultatif).
STREAM	`application/xml` (facultatif)

Description

- Dans une demande HTTP **POST** , indiquez `Content-Type` ou `x-msg-class`. Si vous spécifiez les deux, ils doivent être cohérents ou une exception HTTP `Bad Request` , `Status code 400` est renvoyée. Si vous omettez les deux, `Content-Type` et `x-msg-class`, un `Content-Type` de `text/*` est utilisé.
- Le type de contenu (`Content-Type`) est toujours défini dans la réponse à un **GET** ou à un **DELETE** HTTP comportant un corps de message. `Content-Type` est défini conformément aux règles de Tableau 602, à la page 1240.

Tableau 602. Mappage des types de message à `x-msg-class` et `Content-Type`

Format de message	Type de message JMS	`classe-msg-x`	`Content-Type`
Tout sauf `MQFMT_STRING`	Aucun	`BYTES`	`application/octet-stream`
`MQFMT_STRING`	Aucun	`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

Définissez ou renvoyez l'identificateur de corrélation.

Type	Description
Nom d'en-tête HTTP	`x-msg-correlId`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**DELETE**, **GET**, **POST**, `x-msg-require-headers`

Valeurs autorisées	***`String value`***
	Exemple :
	`x-msg-correlId: mycorrelationid`
	Les chaînes placées entre guillemets sont autorisées, par exemple:
	`x-msg-correlId: "my id"`
	`Hex value`
	Une valeur hexadécimale préfixée avec `0x:`; par exemple:
	`x-msg-correlId: 0x:43c1d23a`
	La valeur hexadécimale 0x: est limitée à 48 caractères représentant 24 octets. Les données supplémentaires sont ignorées.

Valeur par défaut	Non applicable

Description

- Sur une demande HTTP **POST** , définit l'ID de corrélation du message créé.

- Dans une demande HTTP **GET** ou **DELETE** , sélectionne le message dans la file d'attente ou la rubrique. Si aucun message n'existe avec l'ID de corrélation spécifié, une réponse `HTTP 504 Gateway Timeout` est renvoyée. `x-msg-correlId` peut être utilisé avec `x-msg-msgID` pour sélectionner un message dans une file d'attente ou une rubrique qui correspond aux deux sélecteurs.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-corelId` dans le message de réponse HTTP sur l'ID de corrélation d'un message.
- Les espaces horizontaux sont autorisés après le préfixe `0x:` .

Remarque :

- La spécification de `x-msg-correlId` sans valeur sur une demande HTTP **GET** ou **DELETE** ; par exemple, `"x-msg-correlId:"`, renvoie le message suivant dans la file d'attente ou le sujet quel que soit son ID de corrélation.
- Si vous spécifiez un sélecteur de 24 caractères ou moins, ou 0x: suivi de 48 caractères ou moins, le pont WebSphere MQ pour HTTP utilise un sélecteur optimisé pour améliorer les performances.
- Un sélecteur de message JMS contenant JMSCorrelationID est utilisé lors de la sélection de messages dans la file d'attente. Ce sélecteur se comporte comme décrit dans Comportement de la sélection.

encoding: en-tête d'entité HTTP `x-msg-encoding`

Définissez ou renvoyez le codage du message.

Type	Description
Nom d'en-tête HTTP	`codage-msg-x`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`
Valeurs autorisées	Liste séparée par des virgules des valeurs suivantes: **DECIMAL_NORMAL** **DECIMAL_REVERSED** **FLOAT_IEEE_NORMAL** **FLOAT_IEEE_REVERSED** **FLOAT_S390** **INTEGER_NORMAL** **INTEGER_REVERSED** Par exemple : `x-msg-encoding: INTEGER_NORMAL,DECIMAL_NORMAL,FLOAT_IEEE_NORMAL` **Remarque :** La valeur est sensible à la casse
Valeur par défaut	`DECIMAL_NORMAL, FLOAT_IEEE_NORMAL, INTEGER_NORMAL`

Description

- Dans une demande HTTP **POST** , indique le codage du message créé.
- Sur une demande HTTP **GET** ou **DELETE** , l'en-tête `x-msg-encoding` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-encoding` dans le message de réponse HTTP sur la propriété de codage d'un message.

`expiration`: HTTP `x-msg-expiration` entity-header

Définissez ou renvoyez la durée d'expiration du message.

Type	Description
Nom d'en-tête HTTP	`x-msg-expiration`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`

Valeurs autorisées	**UNLIMITED**
	Par exemple,
	`x-msg-expiry: UNLIMITED`
	Integer value
	Millisecondes avant expiration.
	Par exemple,
	`x-msg-expiry: 10000`

Valeur par défaut	`UNLIMITED`

Description

- Lorsqu'il est défini sur une demande HTTP **POST** , le message de demande expire dans le délai spécifié.
- Sur une demande HTTP **GET** ou **DELETE** , l'en-tête `x-msg-expiration` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-expiration` dans le message de réponse HTTP sur l'heure d'expiration d'un message.
- `UNLIMITED` indique que le message n'expire jamais.
- L'expiration d'un message commence à partir du moment où le message arrive dans la file d'attente, de sorte que le temps d'attente du réseau est ignoré.
- La valeur maximale est limitée par WebSphere MQ à 214748364700 millisecondes. Si une valeur supérieure à cette valeur est spécifiée, l'heure d'expiration maximale possible est prise en compte.

`format`: en-tête d'entité HTTP `x-msg-format`

Définissez ou renvoyez le format de message WebSphere MQ .

Type	Description
Nom d'en-tête HTTP	`format-msg-x`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`
Valeurs autorisées	**NONE** Par exemple : `x-msg-format: NONE` ***String value*** Toute valeur définie par l'utilisateur pouvant comporter jusqu'à huit caractères. Par exemple :

Type	Description
	`x-msg-format: myformat`
Valeur par défaut	None

Description

- Lorsqu'il est défini sur une demande HTTP **POST** , définissez le format du message de demande.
- Sur une demande HTTP **GET** ou **DELETE** , l'en-tête `x-msg-format` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-format` dans le message de réponse HTTP au format d'un message.
- `NONE` est sensible à la casse et indique que le format de message est vide.
- La valeur de `x-msg-format` est utilisée, même si elle est en contradiction avec le type de support de la demande HTTP. Voir Tableau 603, à la page 1243.

Tableau 603. Mappage de content-type et `x-msg-class` au format de message

classe-msg-x	**Spécification**	**Format de message dans la file d'attente/ rubrique**
BYTES	• `application/octet-stream` • application/xml	Message WebSphere MQ : `MQFMT` défini sur `MQC.MQFMT_NONE`
TEXT	• `text/*`	Message WebSphere MQ : `MQFMT` défini sur `MQC.MQFMT_STRING`
MAP	• `application/x-www-form-urlencoded` • `application/xml` (facultatif)	`JMSMap`
STREAM	• `application/xml` (facultatif).	`JMSStream`

msgId: HTTP x-msg-msgId entity-header

Définissez ou renvoyez l'identificateur de message.

Type	Description
Nom d'en-tête HTTP	`x-msg-msgId`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**DELETE**, **GET**, **POST**, `x-msg-require-headers`

Valeurs autorisées	***String value***
	Par exemple :
	`x-msg-msgId: mymsgid`
	Chaînes placées entre guillemets, par exemple, `x-msg-msgId: "my id"`
	Hex value
	Une valeur hexadécimale préfixée avec `0x:`; par exemple,
	`x-msg-msgId: 0x:43c1d23a`

Type	Description
Valeur par défaut	Non applicable

Description

- Dans une demande HTTP **POST** , définit l'ID du message créé.
- Dans une demande HTTP **GET** ou **DELETE** , sélectionne le message dans la file d'attente ou la rubrique. Si aucun message n'existe avec l'ID de message spécifié, une réponse `HTTP 504 Gateway Timeout` est renvoyée. `x-msg-msgId` peut être utilisé avec `x-msg-correlID` pour sélectionner un message dans une file d'attente ou une rubrique qui correspond aux deux sélecteurs.
- Spécifié dans `x-msg-require-headers`, renvoie `x-msg-msgId` dans la réponse HTTP à l'ID d'un message.
- Les espaces horizontaux sont autorisés après le préfixe `0x:`.

Remarque : La spécification de `x-msg-msgId` sans valeur sur une demande HTTP **GET** ou **DELETE** ; par exemple, `"x-msg-msgId:"`, renvoie le message suivant dans la file d'attente ou la rubrique quel que soit son ID message.

Un sélecteur de message JMS contenant JMSMessageID est utilisé lors de la sélection de messages dans la file d'attente. Ce sélecteur se comporte comme décrit dans Comportement de la sélection .

`persistence`: en-tête d'entité HTTP `x-msg-persistence`

Définissez ou renvoyez la persistance des messages.

Type	Description
Nom d'en-tête HTTP	`x-msg-persistence`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`

Valeurs autorisées	**NON_PERSISTENT**
	Le message ne survit pas aux défaillances du système ou aux redémarrages du gestionnaire de files d'attente.
	Par exemple :
	`x-msg-persistence: NON_PERSISTENT`
	PERSISTENT
	Le message survit aux échecs du système et aux redémarrages du gestionnaire de files d'attente.
	Par exemple :
	`x-msg-persistence: PERSISTENT`
	AS_DESTINATION
	S'applique à **POST** uniquement.
	Utilisez la persistance par défaut de la destination déterminée par le fournisseur de messages.
	Remarque : Sensible à la casse

Valeur par défaut	NON_PERSISTENT

Description

- Lorsqu'il est défini sur une demande HTTP **POST** , définissez la persistance du message de demande.
- Sur une demande HTTP **GET** ou **DELETE** , l'en-tête `x-msg-persistence` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-persistence` dans le message de réponse HTTP sur la persistance d'un message.

`priority`: en-tête d'entité HTTP `x-msg-priority`

Définissez ou renvoyez la priorité du message.

Type	Description
Nom d'en-tête HTTP	`x-msg-priorité`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`

Valeurs autorisées	**LOW**
	Par exemple :
	`x-msg-priority: LOW`
	MEDIUM
	Cette priorité est égale au niveau de priorité WebSphere MQ 4. Par exemple :
	`x-msg-priority: MEDIUM`
	HIGH
	Par exemple :
	`x-msg-priority: HIGH`
	Integer value
	Représentation sous forme de chaîne d'un entier compris entre 0 et 9 ; par exemple,
	`x-msg-priority: 3`
	AS_DESTINATION
	S'applique à **POST** uniquement.
	Utilisez la priorité par défaut de la destination déterminée par le fournisseur de messages.
	Remarque : Sensible à la casse

Valeur par défaut	`MEDIUM`

Description

- Lorsqu'elle est définie sur une demande HTTP **POST** , définissez la priorité du message de demande.
- Sur une demande HTTP **GET** ou **DELETE** , l'en-tête `x-msg-priority` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-priority` dans le message de réponse HTTP sur la priorité d'un message.

`priority`: en-tête d'entité HTTP `x-msg-priority`

Définissez ou renvoyez la priorité du message.

Type	Description
Nom d'en-tête HTTP	`x-msg-priorité`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`

Valeurs autorisées	**LOW**
	Par exemple :
	`x-msg-priority: LOW`
	MEDIUM
	Cette priorité est égale au niveau de priorité WebSphere MQ 4. Par exemple :
	`x-msg-priority: MEDIUM`
	HIGH
	Par exemple :
	`x-msg-priority: HIGH`
	Integer value
	Représentation sous forme de chaîne d'un entier compris entre 0 et 9 ; par exemple,
	`x-msg-priority: 3`
	AS_DESTINATION
	S'applique à **POST** uniquement.
	Utilisez la priorité par défaut de la destination déterminée par le fournisseur de messages.
	Remarque : Sensible à la casse

Valeur par défaut	`MEDIUM`

Description

- Lorsqu'elle est définie sur une demande HTTP **POST** , définissez la priorité du message de demande.
- Sur une demande HTTP **GET** ou **DELETE** , l'en-tête `x-msg-priority` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-priority` dans le message de réponse HTTP sur la priorité d'un message.

`replyTo`: HTTP `x-msg-replyTo` entity-header

Définissez ou renvoyez la file d'attente de réponse aux messages et le nom du gestionnaire de files d'attente.

Type	Description
Nom d'en-tête HTTP	`x-msg-replyTo`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`

Type	Description

Valeurs autorisées	***URI***
	Un URI point à point ; par exemple,
	`x-msg-replyTo: /msg/queue/myReplyQueue`
	`x-msg-replyTo: /msg/queue/myReplyQueue@myReplyQueueManager`
	Remarque : Sensible à la casse

Valeur par défaut	`MEDIUM`

Description

- Lorsqu'il est défini sur une demande HTTP **POST** , définissez la destination replyTo du message de demande.
- Sur une demande HTTP **GET** ou **DELETE** , l'en-tête `x-msg-replyTo` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-replyTo` dans le message de réponse HTTP sur la file d'attente de réponse et le nom de gestionnaire de files d'attente d'un message .

Remarque : L'URI dans la réponse HTTP peut inclure le nom du gestionnaire de files d'attente auquel le pont WebSphere MQ pour HTTP est connecté.

`Serveur`: en-tête de réponse HTTP

Renvoie des informations sur le serveur et le protocole auxquels le client est connecté.

Type	Description
Nom d'en-tête HTTP	SERVER
Type d'en-tête HTTP	En-tête de réponse
Valide dans le message de demande HTTP	`x-msg-require-headers`

Valeur renvoyée	`WMQ-HTTP/1.1 JEE-Bridge/1.1`
	ou
	`Server: Product-token WMQ-HTTP/1.1 JEE-Bridge/1.1`

Description

- Si WebSphere MQ Bridge for HTTP est déployé sur un serveur d'applications, le pont WebSphere MQ pour les détails HTTP est ajouté à l'en-tête de réponse du serveur. Par exemple, le pont WebSphere MQ pour HTTP déployé sur WebSphere Application Server Community Edition, appelé Apache-Coyote, donne la réponse suivante:

```
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
```

`require-headers`: en-tête de requête HTTP `x-msg-require-headers`

Définissez les en-têtes à renvoyer dans le message de réponse HTTP.

Type	Description
Nom d'en-tête HTTP	`x-msg-require-headers`
Type d'en-tête HTTP	En-tête de demande

Type	Description
Valide dans le message de demande HTTP	**POST**, **GET**, **DELETE**

Valeurs autorisées	Liste séparée par des virgules des noms d'en-tête d'entité:
	ALL
	ALL-USR
	class
	content-location
	correlId
	encoding
	expiry
	format
	msgId
	NO_require-headers
	persistence
	priority
	replyTo
	server
	timestamp
	usr-*property name*
	Par exemple :
	`x-msg-require-headers: msgId`
	ou,
	`x-msg-require-headers: expiry,correlId,timestamp`
	Pour demander une propriété spécifique:
	`x-msg-require-headers: usr-myCustomProperty`
	Pour demander toutes les propriétés:
	`x-msg-require-headers: ALL-USR, ALL`

Valeur par défaut	`NO_require-headers`

Description

- La valeur de `x-msg-require-headers` n'est pas sensible à la casse, sauf dans les cas des constantes `ALL`, `NO_require-headerset` `ALL-USR` et de la variable *`property-name`* .

`timestamp`: en-tête d'entité HTTP `x-msg-timestamp`

Renvoie l'horodatage du message.

Type	Description
Nom d'en-tête HTTP	`horodatage-msg-x`
Type d'en-tête HTTP	En-tête d'entité

Type	Description
Valide dans le message de demande HTTP	`x-msg-require-headers`
Valeur renvoyée	***HTTP-date*** Date au format ; jour, date mois année heure-zone ; par exemple, `Sun, 06 Nov 1994 08:49:37 GMT` Défini par RFC 822et mis à jour dans RFC 1123.
Valeur par défaut	Non applicable

Description

- Sur une demande HTTP **POST**, **GET** ou **DELETE** , l'en-tête `x-msg-timestamp` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-timestamp` dans le message de réponse HTTP sur l'horodatage d'un message.

`usr`: en-tête d'entité HTTP `x-msg-usr`

Définissez ou renvoyez les propriétés de l'utilisateur.

Type	Description
Nom d'en-tête HTTP	`x-msg-usr`
Type d'en-tête HTTP	En-tête d'entité
Valide dans le message de demande HTTP	**POST**, `x-msg-require-headers`

Valeurs autorisées	Voir «Syntaxe», à la page 1250; par exemple,
	`x-msg-usr: myProp1;5;i1,`
	`x-msg-usr: myProp2;"My String";string`

Valeur par défaut	Non applicable

Description

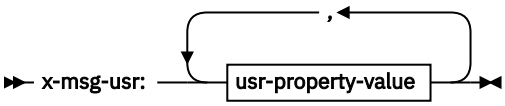
- Lorsqu'il est défini sur une demande HTTP **POST** , définissez les propriétés utilisateur du message de demande.
- Sur une demande HTTP **GET** ou **DELETE** , l'en-tête `x-msg-usr` est ignoré.
- Spécifié dans `x-msg-require-headers`, définit `x-msg-usr` dans le message de réponse HTTP sur les propriétés utilisateur d'un message.
- Plusieurs propriétés peuvent être définies sur un message. Spécifiez plusieurs propriétés séparées par des virgules dans un même en-tête `x-msg-usr` ou en utilisant deux ou plusieurs instances distinctes de l'en-tête `x-msg-usr` .
- Vous pouvez demander qu'une propriété spécifique soit renvoyée dans la réponse à une demande **GET** ou **DELETE** . Indiquez le nom de la propriété dans l'en-tête `x-msg-require-headers` de la demande, en utilisant le préfixe `usr-`. Par exemple :

```
x-msg-require-headers: usr-myProp1
```

- Pour demander que toutes les propriétés utilisateur soient renvoyées dans une réponse, utilisez la constante ALL-USR . Par exemple :

```
x-msg-require-headers: ALL-USR
```

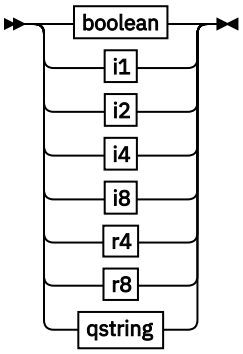
Syntaxe



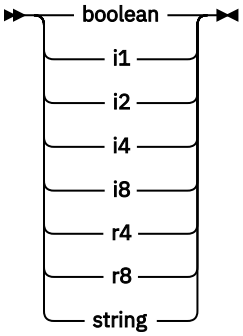
usr-property-value

property-name

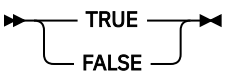
usr-value



usr-type



boolean

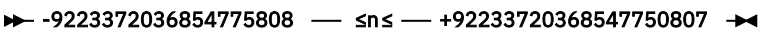


i1

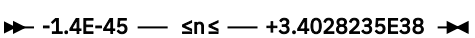
i2

i4

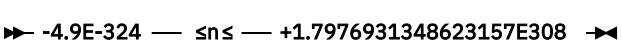
i8



r4



r8



qstring

`wait`: en-tête de demande HTTP `x-msg-wait`

Définissez le délai d'attente d'un message avant de renvoyer un message de réponse `HTTP 504 Gateway Timeout`.

Type	Description
Nom d'en-tête HTTP	`attente-msg-x`
Type d'en-tête HTTP	En-tête de demande
Valide dans le message de demande HTTP	**GET**, **DELETE**

Valeur autorisée	**NO_WAIT**
	Par exemple :
	`x-msg-wait: NO_WAIT`
	Integer value
	Nombre de millisecondes pendant lesquelles le pont WebSphere MQ pour HTTP attend l'arrivée d'un message, par exemple,
	`x-msg-wait: 8`

Valeur par défaut	`NO_WAIT`

Description

- Dans une demande HTTP **POST** , l'en-tête `x-msg-wait` est ignoré.
- Sur une demande HTTP **GET** ou **DELETE** , `x-msg-wait` indique le temps d'attente d'un message avant de renvoyer une réponse `HTTP 504 Gateway Timeout`.
- `NO_WAIT` est sensible à la casse.
- Le temps d'attente maximal par défaut est de 35000. Vous pouvez modifier la valeur par défaut en définissant le paramètre `maximum_wait_time` du servlet. Pour plus d'informations, voir la section Installation, configuration et vérification du pont WebSphere MQ pour HTTP .
- Si vous définissez une valeur supérieure à `maximum_wait_time`, `maximum_wait_time` est utilisé à la place.

Codes retour HTTP

Liste des codes retour du pont WebSphere MQ pour HTTP

Le pont WebSphere MQ pour HTTP renvoie quatre types d'erreur:

Erreurs de servlet
MQHTTP0001 et MQHTTP0002 sont des erreurs de servlet. Ils sont consignés, mais ne sont pas renvoyés au client HTTP.

Opérations réussies
Un code d'état HTTP compris entre 200 et 299 indique une opération réussie.

Erreurs client
Un code d'état HTTP compris entre 400 et 499 indique une erreur client. WebSphere MQ Bridge for HTTP codes retour dans la plage MQHTTP40001 - MQHTTP49999 correspondent à des erreurs client.

Erreurs du serveur

Un code de statut HTTP compris entre 500 et 599 indique une erreur client. WebSphere MQ Bridge for HTTP codes retour dans la plage MQHTTP50001 - MQHTTP59999 correspondent à des erreurs de serveur.

Si une erreur de serveur se produit, une trace de pile complète est générée dans le journal des erreurs du serveur d'applications. La trace de pile est également renvoyée au client HTTP dans la réponse HTTP. Gérez la trace de pile dans l'application client ou adressez-vous à l'administrateur du serveur d'applications pour résoudre le problème.

Si la trace de pile contient des erreurs d'adaptateur de ressources, consultez la documentation de votre adaptateur de ressources.

Liste alphabétique des codes retour

HTTP 200: OK

Cette classe de code de statut indique que la demande a été reçue, comprise et acceptée avec succès.

Code de statut HTTP
`200 OK`

HTTP 204: Aucun contenu

Envoyés suite à une demande HTTP **GET** ou **DELETE** et `x-msg-range: 0` ont été envoyés dans la demande.

Code de statut HTTP
`204 No Content`

MQHTTP0001: Aucune fabrique de connexions spécifiée dans le contexte de servlet

Erreur de servlet

Explication
Erreur de servlet

Code de statut HTTP
Aucun

Réponse du programmeur
L'emplacement où ces erreurs sont consignées est spécifique à votre serveur d'applications. Reportez-vous à la documentation de votre serveur d'applications.

MQHTTP0002: Impossible d'obtenir le gestionnaire de connexions pour *queueOrTopic* à l'aide du nom JNDI *jndiNameTried*

Erreur de servlet

Explication
Erreur de servlet

Code de statut HTTP
Aucun

Réponse du programmeur
L'emplacement où ces erreurs sont consignées est spécifique à votre serveur d'applications. Reportez-vous à la documentation de votre serveur d'applications.

MQHTTP40001: réservé

Réservé

Code de statut HTTP
`400 Bad Request`

MQHTTP40002: n'est pas valide pour le transport WebSphere MQ pour HTTP

L'URI spécifié dans la demande HTTP n'est pas valide.

Explication
L'URI spécifié dans la demande HTTP n'est pas valide.

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Vérifiez que le format et la syntaxe de l'URI spécifié sont corrects.

MQHTTP40003: n'est pas valide. `@qmgr` n'est valide que sur POST

L'option d'URI `@qmgr` a été spécifiée dans un URI pour une demande HTTP qui n'est pas une demande **POST** .

Explication
L'option d'URI `@qmgr` a été spécifiée dans un URI pour une demande HTTP qui n'est pas une demande **POST** .

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Si vous tentez d'insérer un message à l'aide de l'instruction **POST** , remplacez la demande HTTP par une demande **POST** . Si vous tentez d'obtenir un message à l'aide des instructions **DELETE** ou **GET** , supprimez `@qmgr` de l'URI.

MQHTTP40004: Type de contenu non `valide` spécifié

La zone d'en-tête `Content-Type` spécifiée dans une demande **POST** n'est pas compatible avec la valeur d'en-tête `x-msg-class` .

Explication
La zone d'en-tête `Content-Type` spécifiée dans une demande **POST** n'est pas compatible avec la valeur d'en-tête `x-msg-class` .

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Remplacez la zone d'en-tête `Content-Type` par une zone prise en charge. L'en-tête `Content-Type` doit être compatible avec la zone d'en-tête `x-msg-class` spécifiée.

MQHTTP40005: Valeur d'en-tête de message incorrecte

Une zone d'en-tête prise en charge a été spécifiée avec une valeur qui n'est pas valide pour la demande spécifiée.

Explication
Une zone d'en-tête prise en charge a été spécifiée avec une valeur qui n'est pas valide pour la demande spécifiée.

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Remplacez la valeur indiquée pour la zone d'en-tête donnée par une valeur correcte. Vérifiez la casse de la valeur spécifiée, car certaines zones d'en-tête ont des valeurs sensibles à la casse.

MQHTTP40006: *`Header_name`* n'est pas un en-tête de demande valide

Un en-tête valide uniquement dans un message de réponse HTTP a été spécifié dans un message de demande HTTP.

Explication
Un en-tête valide uniquement dans un message de réponse HTTP a été spécifié dans un message de demande HTTP.

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Supprimez de la demande HTTP les en-têtes qui ne sont valides que dans une réponse HTTP ; par exemple, `x-msg-timestamp`.

MQHTTP40007: *`Header_name`* est uniquement valide sur ...

Un en-tête a été spécifié dans une demande HTTP, mais la zone d'en-tête n'est pas valide pour l'instruction de demande indiquée.

Explication
Un en-tête a été spécifié dans une demande HTTP, mais la zone d'en-tête n'est pas valide pour l'instruction de demande indiquée.

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Supprimez de la demande HTTP les en-têtes qui ne sont pas valides pour l'instruction de demande donnée. Par exemple, `x-msg-encoding` est valide pour les demandes HTTP **POST** , mais non pour les demandes HTTP **GET** ou HTTP **DELETE** .

MQHTTP40008: *`Header_name`* est ...

La longueur maximale de la zone d'en-tête indiquée a été dépassée.

Explication
La longueur maximale de la zone d'en-tête indiquée a été dépassée.

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Remplacez la valeur de la zone d'en-tête par une valeur comprise dans la plage autorisée pour la zone d'en-tête.

MQHTTP40009: La zone d'en-tête *`header_field`* n'est pas valide pour ...

Une zone d'en-tête spécifiée dans une demande HTTP n'est pas prise en charge par le fournisseur de messagerie auquel le pont WebSphere MQ pour HTTP est connecté.

Explication
Une zone d'en-tête spécifiée dans une demande HTTP n'est pas prise en charge par le fournisseur de messagerie auquel le pont WebSphere MQ pour HTTP est connecté. L'erreur se produit si un fournisseur de messagerie qui ne prend pas en charge toutes les fonctions du pont WebSphere MQ pour HTTP est utilisé.

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Supprimez l'en-tête non pris en charge de la demande HTTP.

MQHTTP40010: Le message avec `Content-Type` *`content_type`* n'a pas pu être analysé

Le contenu de la demande HTTP n'est pas compatible avec le `Content-Type` de la demande.

Explication
Le contenu de la demande HTTP n'est pas compatible avec le `Content-Type` de la demande. La syntaxe de la cause courante est incorrecte: `application/x-www-form-urlencoded` ou `application/xml` `data`.

Code de statut HTTP
`400 Bad Request`

Réponse du programmeur
Corrigez le contenu de la demande HTTP de sorte qu'il soit au format correct pour le `Content-Type` de la demande.

MQHTTP40301: Vous ne pouvez pas accéder à ...

Le pont WebSphere MQ pour HTTP n'a pas pu s'authentifier pour la destination spécifiée.

Explication
Le pont WebSphere MQ pour HTTP n'a pas pu s'authentifier pour la destination spécifiée.

Code de statut HTTP
`403 Forbidden`

Réponse du programmeur
Modifiez les propriétés d'authentification de la destination de sorte que WebSphere MQ Bridge for HTTP soit autorisé à s'y connecter. Vous pouvez également spécifier une destination à laquelle le pont WebSphere MQ pour HTTP est autorisé à se connecter.

MQHTTP40302: Vous n'êtes pas autorisé à ...

Le pont WebSphere MQ pour HTTP n'a pas pu se connecter au gestionnaire de files d'attente.

Explication
Le pont WebSphere MQ pour HTTP n'a pas pu se connecter au gestionnaire de files d'attente. Le pont WebSphere MQ pour la configuration de la sécurité HTTP est incorrect.

Code de statut HTTP
`403 Forbidden`

Réponse du programmeur
Modifiez la configuration d'authentification du gestionnaire de files d'attente de sorte que WebSphere MQ Bridge for HTTP soit autorisé à s'y connecter. Vous pouvez également configurer le pont WebSphere MQ pour HTTP afin qu'il se connecte à un gestionnaire de files d'attente auquel il est autorisé à se connecter.

MQHTTP40401: La destination *destination_name* est introuvable

La destination spécifiée dans l'URI de demande HTTP est introuvable par le pont WebSphere MQ pour HTTP.

Explication
La destination spécifiée dans l'URI de demande HTTP est introuvable par le pont WebSphere MQ pour HTTP.

Code de statut HTTP
`404 Not found`

Réponse du programmeur
Vérifiez que la destination spécifiée dans l'URI de demande HTTP existe ou indiquez une autre destination.

MQHTTP40501: Méthode *method_name* non autorisée

La méthode spécifiée dans la demande HTTP n'est pas prise en charge par le pont WebSphere MQ pour HTTP.

Explication

La méthode spécifiée dans la demande HTTP n'est pas prise en charge par le pont WebSphere MQ pour HTTP.

Code de statut HTTP

`405 Method not allowed`

Réponse du programmeur

Remplacez la méthode spécifiée dans la demande HTTP par une méthode prise en charge par le pont WebSphere MQ pour HTTP.

MQHTTP41301: Le message envoyé était trop volumineux pour la destination

La destination spécifiée dans l'URI de la demande HTTP POST ne peut pas accepter les messages qui sont aussi longs que le message spécifié dans la demande HTTP.

Explication

La destination spécifiée dans l'URI de la demande HTTP POST ne peut pas accepter les messages qui sont aussi longs que le message spécifié dans la demande HTTP.

Code de statut HTTP

`413 Request entity too large`

Réponse du programmeur

Réduisez la taille du message spécifié dans la demande HTTP. Vous pouvez également spécifier une destination pouvant prendre en charge les messages de la longueur souhaitée.

MQHTTP41501: Le jeu de caractères du type de support n'est pas pris en charge

Le jeu de caractères spécifié dans la zone d'en-tête `Content-Type` n'est pas pris en charge par le pont WebSphere MQ pour HTTP.

Explication

Le jeu de caractères spécifié dans la zone d'en-tête `Content-Type` n'est pas pris en charge par le pont WebSphere MQ pour HTTP.

Code de statut HTTP

`415 Unsupported media type`

Réponse du programmeur

Remplacez le jeu de caractères de la zone d'en-tête `Content-Type` par un jeu de caractères pris en charge par le pont WebSphere MQ pour HTTP.

MQHTTP41502: Media-type *media-type* n'est pas pris en charge ...

Le type de support spécifié dans la demande HTTP n'est pas pris en charge par le pont WebSphere MQ pour HTTP pour l'instruction HTTP spécifiée.

Explication
Le type de support spécifié dans la demande HTTP n'est pas pris en charge par le pont WebSphere MQ pour HTTP pour l'instruction HTTP spécifiée.

Code de statut HTTP
`415 Unsupported media type`

Réponse du programmeur
Remplacez le type de support spécifié dans la demande HTTP par un type pris en charge par WebSphere MQ Bridge for HTTP pour l'instruction HTTP spécifiée.

MQHTTP41503: Le type de support *media-type* n'est pas pris en charge ...

Le type de support spécifié dans la demande HTTP n'est pas pris en charge par le pont WebSphere MQ pour HTTP pour la zone d'en-tête `x-msg-class` spécifiée.

Explication
Le type de support spécifié dans la demande HTTP n'est pas pris en charge par le pont WebSphere MQ pour HTTP pour la zone d'en-tête `x-msg-class` spécifiée.

Code de statut HTTP
`415 Unsupported media type`

Réponse du programmeur
Remplacez le type de support spécifié dans la demande HTTP par un type pris en charge par WebSphere MQ Bridge for HTTP pour la zone d'en-tête `x-msg-class` spécifiée.

MQHTTP41701: L'en-tête HTTP `Expect` n'est pas pris en charge

Le pont WebSphere MQ pour HTTP ne prend pas en charge la zone d'en-tête `Expect` .

Explication
L'en-tête `Expect` a été spécifié dans une demande HTTP. Le pont WebSphere MQ pour HTTP ne prend pas en charge la zone d'en-tête `Expect` .

Code de statut HTTP
`417 Expectation failed`

Réponse du programmeur
Supprimez l'en-tête `Expect` de la demande HTTP.

MQHTTP50001: Un problème inattendu s'est produit ...

Une erreur s'est produite dans le pont WebSphere MQ pour HTTP.

Explication
Une erreur s'est produite dans le pont WebSphere MQ pour HTTP.

Code de statut HTTP
`500 Internal server error`

Réponse du programmeur
Contactez l'administrateur système de WebSphere MQ Bridge for HTTP.

MQHTTP50201: Une erreur s'est produite entre le pont WebSphere MQ pour HTTP et le gestionnaire de files d'attente

Une erreur s'est produite entre le pont WebSphere MQ pour HTTP et le gestionnaire de files d'attente

Explication
Une erreur s'est produite entre le pont WebSphere MQ pour HTTP et le gestionnaire de files d'attente

Code de statut HTTP
`502 Bad Gateway`

Réponse du programmeur
Contactez l'administrateur système de WebSphere MQ Bridge for HTTP.

MQHTTP50401: Expiration de l'extraction de message

Aucun message correspondant aux paramètres de demande spécifiés dans HTTP **GET** ou HTTP **DELETE** n'a été renvoyé dans le délai imparti.

Explication
Aucun message correspondant aux paramètres de demande spécifiés dans HTTP **GET** ou HTTP **DELETE** n'a été renvoyé dans le délai imparti. Le code retour indique qu'aucun message approprié n'était disponible à tout moment pendant la durée de vie de la demande HTTP.

Code de statut HTTP
`504 Gateway timeout`

Réponse du programmeur
Si un message est attendu, vérifiez les zones d'en-tête de la demande HTTP, telles que `x-msg-correlId` et `x-msg-msgid`. Vérifiez que la destination spécifiée dans l'URI de demande HTTP est correcte. Essayez d'étendre le temps d'attente de la demande HTTP à l'aide de la zone d'en-tête `x-msg-wait` .

MQHTTP50501: HTTP 1.1 et versions suivantes ...

Le protocole HTTP utilisé dans la demande HTTP n'est pas pris en charge par le pont WebSphere MQ pour HTTP.

Explication
Le protocole HTTP utilisé dans la demande HTTP n'est pas pris en charge par le pont WebSphere MQ pour HTTP.

Code de statut HTTP
`505 HTTP version not supported`

Réponse du programmeur
Modifiez la demande HTTP pour utiliser le protocole HTTP V1.1 ou version ultérieure.

Types de message et mappages de message pour WebSphere Bridge for HTTP

WebSphere MQ Bridge for HTTP prend en charge quatre classes de message, TEXT, BYTES, STREAM et MAP. Les classes de message sont mappées vers les types de message JMS et HTTP `Content-Type`.

HTTP POST

Le type de message qui arrive à la destination dépend de la valeur de l'en-tête `x-msg-class` ou du `Content-Type` de la demande HTTP. Tableau 604, à la page 1260 affiche le type HTTP `Content-Type` qui correspond à chaque `x-msg-class`. Vous pouvez utiliser l'une ou l'autre de ces zones pour définir le type et le format du message. Si les deux champs sont définis et qu'ils sont définis de manière incohérente, un `Bad Request exception` est renvoyé (`HTTP 400, MQHTTP20004`).

Tableau 604. Mappage entre `x-msg-class` et HTTP `Content-Type`

classe-msg-x	**HTTP Content-Type**
BYTES	`application/octet-stream` `application/xml`
TEXT	`text/*`
MAP	`application/x-www-form-urlencoded` `application/xml` (facultatif).
STREAM	`application/xml` (facultatif)

Si le type de message JMS est défini dans l'en-tête MQRFH2 , il est mappé en fonction de Tableau 605, à la page 1260.

Tableau 605. Mappage entre les types de message `x-msg-class` et JMS.

classe-msg-x	**Type de message JMS**
BYTES	`jms_bytes`
TEXT	`jms_text`
MAP	`jms_map`
STREAM	`jms_stream`

Le type de message JMS est toujours défini pour une classe de message MAP ou STREAM. Il n'est pas toujours défini pour une classe de message BYTES ou TEXT. Si un MQRFH2 doit être créé pour la demande, le type de message JMS est toujours défini. Sinon, si aucun MQRFH2 n'est créé, aucun type de message JMS n'est défini. Un MQRFH2 est créé si des propriétés utilisateur sont définies dans la demande, à l'aide de l'en-tête `x-msg-usr` .

Si le type de message JMS est défini, le format de message est défini sur MQFMT_NONE. Voir Tableau 607, à la page 1261:

Tableau 606. Mappage entre `x-msg-class` et le format de message WebSphere MQ

classe-msg-x	**Format de message avec MQRFH2 présent dans le message**	**Format de message sans *aucun* MQRFH2 présent dans le message**
BYTES	MQFMT_NONE	MQFMT_NONE
TEXT	MQFMT_NONE	MQFMT_STRING
MAP	MQFMT_NONE	Impossible
STREAM	MQFMT_NONE	Impossible

HTTP GET ou DELETE

Le type ou le format de message extrait détermine la valeur de l'en-tête `x-msg-class` et le `Content-Type` de la réponse HTTP. L'en-tête `x-msg-class` est renvoyé uniquement s'il est demandé dans une demande `x-msg-headers` .

Le Tableau 607, à la page 1261 décrit les mappages entre `x-msg-class` et `Content-Type`, ainsi que le type de message extrait de la file d'attente ou de la rubrique.

Tableau 607. Mappage des types de message à `x-msg-class` et `Content-Type`

Format de message	Type de message JMS	classe-msg-x	Content-Type
Tout sauf MQFMT_STRING	Aucun	BYTES	application/octet-stream
MQFMT_STRING	Aucun	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

Sérialisation des classes de message MAP et STREAM

Les classes de message MAP et STREAM sont sérialisées vers le client dans la réponse HTTP de la même manière qu'un message est sérialisé dans une file d'attente.

Pour MAP, les triplets de nom, de type et de valeur XML sont codés comme suit:

```
<map>
    <elt name="elementname1" dt="datatype1">value1</elt>
    <elt name="elementname2" dt="datatype2">value2</elt>
    ...
</map>
```

STREAM est similaire à MAP, mais il n'a pas de noms d'élément:

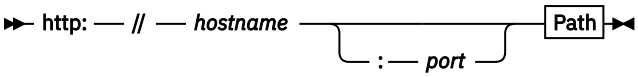
```
<stream>
    <elt dt="datatype1">value1</elt>
    <elt dt="datatype2">value2</elt>
    ...
</stream>
```

Remarque : `datatype` est l'un des types de données définis pour la définition des propriétés définies par l'utilisateur et répertoriés dans le «usr: en-tête d'entité HTTP x-msg-usr», à la page 1249. L'attribut `dt="string"` est omis pour les éléments de chaîne car le type de données par défaut est `string`.

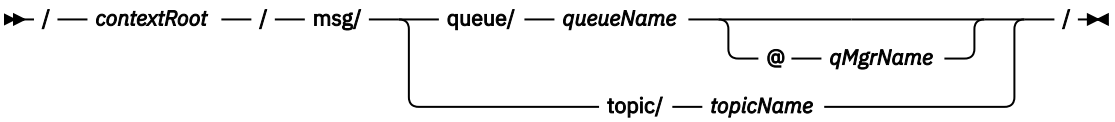
Format d'URI

URI interceptés par le pont WebSphere MQ pour HTTP.

Syntax



Path



Remarque :

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**

Description

Déployez le pont WebSphere MQ pour le servlet HTTP sur votre serveur d'applications JEE avec la racine de contexte *contextRoot*. Demandes à

```
http://hostname:port/context_root/msg/queue/queueName@qMgrName
```

et

```
http://hostname:port/context_root/msg/topic/topicString
```

sont interceptés par le pont WebSphere MQ pour HTTP.

Les classes et interfacesIBM WebSphere MQ .NET

Les classes et interfaces IBM WebSphere MQ .NET sont répertoriées par ordre alphabétique. Les propriétés, les méthodes et les constructeurs sont décrits.

Classe MQAsyncStatus .NET

Utilisez `MQAsyncStatus` pour demander le statut de l'activité MQI précédente ; par exemple, demander la réussite des opérations d'insertion asynchrone précédentes. `MQAsyncStatus` encapsule les fonctions de la structure de données MQSTS .

Classe

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQAsyncStatus
```

```
public class IBM.WMQ.MQAsyncStatus extends IBM.WMQ.MQBaseObject;
```

- «Propriétés», à la page 1262
- «Constructeurs», à la page 1263

Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

`public static int CompCode {get;}`

Code achèvement de la première erreur ou avertissement.

`public static int Reason {get;}`

Code raison de la première erreur ou avertissement.

public static int PutSuccessCount {get;}

Nombre d'appels MQI asynchrones réussis.

public static int PutWarningCount {get;}

Nombre d'appels d'insertion MQI asynchrones ayant abouti avec un avertissement.

public static int PutFailureCount {get;}

Nombre d'appels MQI asynchrones ayant échoué.

public static int ObjectType {get;}

Type d'objet de la première erreur. Valeurs possibles :

- MQC.MQOT_ALIAS_Q
- MQC.MQOT_LOCAL_Q
- MQC.MQOT_MODEL_Q
- MQC.MQOT_Q
- MQC.MQOT_REMOTE_Q
- MQC.MQOT_TOPIC
- 0, ce qui signifie qu'aucun objet n'est renvoyé

public static string ObjectName {get;}

Nom de l'objet.

public static string ObjectQMgrName {get;}

Nom du gestionnaire de files d'attente d'objets.

public static string ResolvedObjectName {get;}

Nom de l'objet résolu.

public static string ResolvedObjectQMgrName {get;}

Nom du gestionnaire de files d'attente d'objet résolu.

Constructeurs

public MQAsyncStatus() throws MQException;

Méthode de construction: construit un objet avec des zones initialisées à zéro ou à blanc, selon le cas.

Classe MQAuthenticationInformationRecord .NET

Utilisez MQAuthenticationInformationRecord pour spécifier des informations sur un authentificateur à utiliser dans une connexion client SSL WebSphere MQ . MQAuthenticationInformationRecord encapsule un enregistrement d'informations d'authentification, MQAIR.

Classe

```
System.Object
    │
    └─ IBM.WMQ.MQAuthenticationInformationRecord
```

public class IBM.WMQ.MQAuthenticationInformationRecord extends System.Object;

- «Propriétés», à la page 1264
- «Constructeurs», à la page 1264

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

public long Version {get; set;}

Numéro de version de la structure.

public long AuthInfoType {get; set;}

Type des informations d'authentification. Cet attribut doit être défini sur l'une des valeurs suivantes:

- OCSP -La vérification du statut de révocation de certificat est effectuée à l'aide d'OCSP.
- CRLLDAP -La vérification du statut de révocation de certificat est effectuée à l'aide des listes de révocation de certificat sur les serveurs LDAP.

public string AuthInfoConnName {get; set;}

Nom DNS ou adresse IP de l'hôte sur lequel le serveur LDAP s'exécute, avec un numéro de port facultatif. Ce mot clé est obligatoire.

public string LDAPPassword {get; set;}

Mot de passe associé au nom distinctif de l'utilisateur qui accède au serveur LDAP. Cette propriété s'applique uniquement lorsque **AuthInfoType** est défini sur CRLLDAP.

public string LDAPUserName {get; set;}

Nom distinctif de l'utilisateur qui accède au serveur LDAP. Lorsque vous définissez cette propriété, LDAPUserNameLength et LDAPUserNamePtr sont automatiquement définis correctement. Cette propriété s'applique uniquement lorsque AuthInfoType est défini sur CRLLDAP.

public string OCSPResponderURL {get; set;}

Adresse URL à laquelle le canal répondeur OCSP peut être contacté. Cette propriété s'applique uniquement lorsque AuthInfoType est défini sur OCSP

Cette zone est sensible à la casse. Il doit commencer par la chaîne http:// en minuscules. Le reste de l'URL peut être sensible à la casse, en fonction de l'implémentation du serveur OCSP.

Constructeurs

MQAuthenticationInformationRecord();

Classe MQDestination .NET

Utilisez MQDestination pour accéder aux méthodes communes à MQQueue et MQTopic. MQDestination est une classe de base abstraite et ne peut pas être instanciée.

Classe

```
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;

- «Propriétés», à la page 1265

- «Des méthodes», à la page 1265
- «Constructeurs», à la page 1267

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

public DateTime CreationDateTime {get;}

Date et heure de création de la file d'attente ou de la rubrique. Initialement contenue dans MQQueue, cette propriété a été déplacée dans la classe MQDestination de base.

Il n'existe pas de valeur par défaut.

public int DestinationType {get;}

Valeur entière décrivant le type de destination utilisée. Initialisée à partir du constructeur de sous-classes, MQQueue ou MQTopic, cette valeur peut prendre l'une des valeurs suivantes:

- MQOT_Q
- MQOT_TOPIC

Il n'existe pas de valeur par défaut.

Des méthodes

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);

Emet MQException.

Extrait un message d'une file d'attente si la destination est un objet MQQueue ou d'une rubrique si la destination est un objet MQTopic , en utilisant une instance par défaut de MQGetMessageOptions pour effectuer l'extraction.

Si l'extraction échoue, l'objet MQMessage n'est pas modifié. Si elle aboutit, le descripteur de message et les parties de données de message du MQMessage sont remplacés par le descripteur de message et les données de message du message entrant.

Tous les appels à WebSphere MQ à partir d'un MQQueueManager particulier sont synchrones. Par conséquent, si vous effectuez une opération d'extraction avec attente, toutes les autres unités d'exécution utilisant le même MQQueueManager sont bloquées pour effectuer d'autres appels WebSphere MQ jusqu'à ce que l'appel Get soit effectué. Si vous avez besoin de plusieurs unités d'exécution pour accéder simultanément à WebSphere MQ , chaque unité d'exécution doit créer son propre objet MQQueueManager .

message
Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée MessageId et CorrelationId sont définis comme requis.

Un client reconnectable renvoie le code anomalie MQRC_BACKED_OUT après une reconnexion réussie, pour les messages reçus sous MQGM_SYNCPOINT.

getMessageOptions
Options contrôlant l'action de la commande get.

L'utilisation de l'option MQC.MQGMO_CONVERT peut générer une exception avec le code anomalie MQC.MQRC_CONVERTED_STRING_TOO_BIG lors de la conversion de codes de caractères codés sur un octet en codes codés sur deux octets. Dans ce cas, le message est copié dans la mémoire tampon sans conversion.

Si *getMessageOptions* n'est pas spécifié, l'option de message utilisée est MQGMO_NOWAIT.

Si vous utilisez l'option MQGMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

MaxMsgSize
Message le plus volumineux que cet objet message doit recevoir. Si le message de la file d'attente est supérieur à cette taille, l'un des deux événements suivants se produit:

- Si l'indicateur MQGMO_ACCEPT_TRUNCATED_MSG est défini dans l'objet MQGetMessageOptions , le message est rempli avec autant de données de message que possible. Une exception est émise avec le code achèvement MQCC_WARNING et le code anomalie MQRC_TRUNCATED_MSG_ACCEPTED .
- Si l'indicateur MQGMO_ACCEPT_TRUNCATED_MSG n'est pas défini, le message reste dans la file d'attente. Une exception est émise avec le code achèvement MQCC_WARNING et le code anomalie MQRC_TRUNCATED_MSG_FAILED .

Si *MaxMsgSize* n'est pas spécifié, l'intégralité du message est extraite.

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

Emet MQException.

Insère un message dans une file d'attente si la destination est un objet MQQueue ou publie un message dans une rubrique si la destination est un objet MQTopic .

Les modifications apportées à l'objet MQMessage une fois l'appel Put effectué n'affectent pas le message réel dans la file d'attente ou la rubrique de publication WebSphere MQ .

Put met à jour les propriétés MessageId et CorrelationId de l'objet MQMessage et n'efface pas les données de message. Les appels Put ou Get supplémentaires font référence aux informations mises à jour dans l'objet MQMessage . Par exemple, dans le fragment de code suivant, le premier message contient a et le second ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

message
Objet MQMessage contenant les données du descripteur de message et le message à envoyer. Le descripteur de message peut être modifié à la suite de cette méthode. Les valeurs du descripteur de message immédiatement après la fin de cette méthode sont les valeurs qui ont été placées dans la file d'attente ou publiées dans la rubrique.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- MQRC_CALL_INTERRUPTED si la connexion est interrompue lors de l'exécution d'un appel Put sur un message persistant et que la reconnexion a abouti.
- MQRC_NONE si la connexion aboutit lors de l'exécution d'un appel Put sur un message non persistant (voir Application Recovery).

putMessageOptions
Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de MQPutMessageOptons est utilisée.

Si vous utilisez l'option MQPMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

Remarque : Pour plus de simplicité et de performances, si vous souhaitez placer un seul message dans une file d'attente, utilisez l'objet MQQueueManager.Put . Vous devez disposer d'un objet MQQueue pour cela.

Constructeurs

`MQDestination` est une classe de base abstraite et ne peut pas être instanciée. Accédez à des destinations à l'aide de constructeurs `MQQueue` et `MQTopic` ou à l'aide de `MQQueueManager.AccessQueue` et `MQQueueManager.AccessTopic methods`.

Classe MQEnvironment .NET

Utilisez `MQEnvironment` pour contrôler la façon dont le constructeur `MQQueueManager` est appelé et pour sélectionner une connexion client MQI WebSphere MQ . La classe `MQEnvironment` contient des propriétés qui contrôlent le comportement de WebSphere MQ.

Classe

```
System.Object
    └─ IBM.WMQ.MQEnvironment
```

```
public class IBM.WMQ.MQEnvironment extends System.Object;
```

- «Propriétés-client uniquement», à la page 1267
- «Propriétés», à la page 1268
- «Constructeurs», à la page 1269

Propriétés-client uniquement

Test de l'émission de `MQException` lors de l'obtention des propriétés.

`public static int CertificateValPolicy {get; set;}`

Définissez les règles de validation de certificat SSL/TLS qui sont utilisées pour valider les certificats numériques reçus des systèmes partenaires distants. Les valeurs admises sont :

- MQC.CERTIFICATE_VALIDATION_POLICY_ANY
- MQC.CERTIFICATE_VALIDATION_POLICY_RFC5280

`public static ArrayList EncryptionPolicySuiteB {get; set;}`

Définissez le niveau de la cryptographie compatible Suite B. Les valeurs admises sont :

- MQC.MQ_SUITE_B_NONE -Il s'agit de la valeur par défaut.
- MQC.MQ_SUITE_B_128_BIT
- MQC.MQ_SUITE_B_192_BIT

`public static string Channel {get; set;}`

Nom du canal pour la connexion au gestionnaire de files d'attente cible. Vous *devez* définir la propriété de canal avant d'instancier une instance `MQQueueManager` en mode client.

`public static int FipsRequired {get; set;}`

Indiquez `MQC.MQSSL_FIPS_YES` pour utiliser uniquement des algorithmes certifiés FIPS si la cryptographie est effectuée dans WebSphere MQ. La valeur par défaut est `MQC.MQSSL_FIPS_NO`.

Si du matériel de cryptographie est configuré, les modules de cryptographie utilisés sont ceux fournis par le produit matériel. Selon le matériel utilisé, il se peut qu'ils ne soient pas certifiés FIPS à un niveau particulier.

`public static string Hostname {get; set;}`

Nom d'hôte TCP/IP de l'ordinateur sur lequel réside le serveur WebSphere MQ . Si le nom d'hôte n'est pas défini et qu'aucune propriété de remplacement n'est définie, le mode de liaison du serveur est utilisé pour la connexion au gestionnaire de files d'attente local.

public static int Port {get; set;}
Port auquel se connecter. Il s'agit du port sur lequel le serveur WebSphere MQ écoute les demandes de connexion entrantes. La valeur par défaut est 1414.

public static string SSLCipherSpec {get; set;}
Définissez SSLCipherSpec sur la valeur de CipherSpec définie sur le canal SVRCONN afin d'activer SSL pour la connexion. La valeur par défaut est Null et SSL n'est pas activé pour la connexion.

public static string sslPeerName {get; set;}

Modèle de nom distinctif. Si sslCipherSpec est défini, cette variable peut être utilisée pour s'assurer que le gestionnaire de files d'attente approprié est utilisé. Si la valeur est null (valeur par défaut), le nom distinctif du gestionnaire de files d'attente n'est pas exécuté. sslPeerName est ignoré si sslCipherSpec est null.

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

public static ArrayList HdrCompList {get; set;}
Liste de compression de données d'en-tête

public static int KeyResetCount {get; set;}
Indique le nombre d'octets non chiffrés envoyés et reçus dans une conversation SSL avant la renégociation de la clé secrète.

public static ArrayList MQAIRArray {get; set;}
Un tableau d'objets MQAuthenticationInformationRecord .

public static ArrayList MsgCompList {get; set;}
Liste de compression de données de message

public static string Password {get; set;}
Mot de passe à authentifier. Le mot de passe référencé à partir de la structure MQCSP est renseigné en définissant cette propriété de mot de passe.

public static string ReceiveExit {get; set;}
Un exit de réception permet d'examiner et de modifier les données reçues d'un gestionnaire de files d'attente. Il est généralement utilisé avec un exit d'émission correspondant au niveau du gestionnaire de files d'attente. Si ReceiveExit est défini sur null, aucun exit de réception n'est appelé.

public static string ReceiveUserData {get; set;}
Données utilisateur associées à un exit de réception. Limité à 32 caractères.

public static string SecurityExit {get; set;}
Un exit de sécurité permet de personnaliser les flux de sécurité qui se produisent lorsqu'une tentative de connexion à un gestionnaire de files d'attente est effectuée. Si SecurityExit est défini sur null, aucun exit de sécurité n'est appelé.

public static string SecurityUserData {get; set;}
Données utilisateur associées à un exit de sécurité. Limité à 32 caractères.

public static string SendExit {get; set;}
Un exit d'émission vous permet d'examiner ou de modifier les données envoyées à un gestionnaire de files d'attente. Il est généralement utilisé avec un exit de réception correspondant au niveau du gestionnaire de files d'attente. Si SendExit est défini sur null, aucun exit d'émission n'est appelé.

public static string SendUserData {get; set;}
Données utilisateur associées à un exit d'émission. Limité à 32 caractères.

public static string SharingConversations {get; set;}

La zone SharingConversations est utilisée sur les connexions à partir d'applications .NET, lorsque ces applications n'utilisent pas de table de définition de canal du client (CCDT).

SharingConversations détermine le nombre maximal de conversations pouvant être partagées sur un socket associé à cette connexion.

La valeur 0 signifie que le canal fonctionne comme avant WebSphere MQ Version 7.0, en ce qui concerne le partage de conversation, la lecture anticipée et le signal de présence.

La zone est transmise dans la table de hachage des propriétés en tant que SHARING_CONVERSATIONS_PROPERTY, lors de l'instanciation d'un gestionnaire de files d'attente WebSphere MQ .

Si vous ne spécifiez pas SharingConversations, la valeur par défaut 10 est utilisée.

public static string SSLCryptoHardware {get; set;}

Définit le nom de la chaîne de paramètres requise pour configurer le matériel cryptographique présent sur le système. SSLCryptoHardware est ignoré si sslCipherSpec a la valeur null.

public static string SSLKeyRepository {get; set;}

Définissez le nom de fichier qualifié complet du référentiel de clés.

Si SSLKeyRepository est défini sur null (valeur par défaut), la variable d'environnement MQSSLKEYR du certificat est utilisée pour localiser le référentiel de clés. SSLCryptoHardware est ignoré si sslCipherSpec a la valeur null.

Remarque : L'extension .kdb est une partie obligatoire du nom de fichier, mais elle n'est pas incluse dans la valeur du paramètre. Le répertoire que vous spécifiez doit exister. WebSphere MQ crée le fichier la première fois qu'il accède au nouveau référentiel de clés, sauf si le fichier existe déjà.

public static string UserId {get; set;}

ID utilisateur à authentifier. L'ID utilisateur référencé à partir de la structure MQCSP est renseigné en définissant UserId. Authentifiez UserId à l'aide d'une API ou d'un exit de sécurité.

Constructeurs

public MQEnvironment()

Classe MQException .NET

Utilisez MQException pour rechercher le code achèvement et le code anomalie d'une fonction WebSphere MQ qui a échoué. Une exception MQException est émise chaque fois qu'une erreur WebSphere MQ se produit.

Classe

```
System.Object
    └ System.Exception
        └ System.ApplicationException
            └ IBM.WMQ.MQException
```

public class IBM.WMQ.MQException extends System.ApplicationException;

- «Propriétés», à la page 1269
- «Constructeurs», à la page 1270

Propriétés

public int CompletionCode {get; set;}

Code achèvement WebSphere MQ associé à l'erreur. Les valeurs possibles sont les suivantes:

- MQException.MQCC_OK
- MQException.MQCC_WARNING
- MQException.MQCC_FAILED

public int ReasonCode {get; set;}
WebSphere MQ code anomalie décrivant l'erreur.

Constructeurs

public MQException(int *completionCode*, int *reasonCode*)

completionCode
Code achèvement WebSphere MQ .

reasonCode
Code achèvement WebSphere MQ .

Classe MQGetMessageOptions .NET

Utilisez MQGetMessageOptions pour spécifier le mode d'extraction des messages. Il modifie le comportement de MQDestination.Get.

Classe

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQGetMessageOptions
```

```
public class IBM.WMQ.MQGetMessageOptions extends IBM.WMQ.MQBaseObject;
```

- «Propriétés», à la page 1270
- «Constructeurs», à la page 1273

Propriétés

Remarque : Le comportement de certaines des options disponibles dans cette classe dépend de l'environnement dans lequel elles sont utilisées. Ces éléments sont marqués d'un astérisque *.

Test de l'émission de MQException lors de l'obtention des propriétés.

public int GroupStatus {get;}*
GroupStatus indique si le message extrait se trouve dans un groupe et s'il s'agit du dernier du groupe. Les valeurs possibles sont les suivantes :

MQC.MQGS_LAST_MSG_IN_GROUP
Le message est le dernier ou le seul message du groupe.

MQC.MQGS_MSG_IN_GROUP
Le message se trouve dans un groupe, mais n'est pas le dernier du groupe.

MQC.MQGS_NOT_IN_GROUP
Le message n'est pas dans un groupe.

public int MatchOptions {get; set;}*
MatchOptions détermine comment un message est sélectionné. Les options de correspondance suivantes peuvent être définies:

MQC.MQMO_MATCH_CORREL_ID
ID de corrélation à mettre en correspondance.

MQC.MQMO_MATCH_GROUP_ID
ID de groupe à mettre en correspondance.

MQC.MQMO_MATCH_MSG_ID
ID message à mettre en correspondance.

MQC.MQMO_MATCH_MSG_SEQ_NUMBER
Correspond au numéro de séquence de message.

MQC.MQMO_NONE
Aucune correspondance n'est requise.

public int Options {get; set;}
Les `options` contrôlent l'action de `MQQueue.get`. Vous pouvez indiquer l'une des valeurs suivantes. Si plusieurs options sont requises, les valeurs peuvent être ajoutées ou combinées à l'aide de l'opérateur OR bit à bit.

MQC.MQGMO_ACCEPT_TRUNCATED_MSG
Autoriser la troncature des données de message.

MQC.MQGMO_ALL_MSGS_AVAILABLE*
Extraire des messages d'un groupe uniquement lorsque tous les messages du groupe sont disponibles.

MQC.MQGMO_ALL_SEGMENTS_AVAILABLE*
Extraire les segments d'un message logique uniquement lorsque tous les segments du groupe sont disponibles.

MQC.MQGMO_BROWSE_FIRST
Parcourez depuis le début de la file d'attente.

MQC.MQGMO_BROWSE_MSG_UNDER_CURSOR*
Parcourir le message sous le curseur de navigation.

MQC.MQGMO_BROWSE_NEXT
Parcourez la position en cours dans la file d'attente.

MQC.MQGMO_COMPLETE_MSG*
Extraire uniquement les messages logiques complets.

MQC.MQGMO_CONVERT
Demandez la conversion des données d'application pour qu'elles soient conformes aux attributs `CharacterSet` et `Codage` de `MQMessage`, avant que les données ne soient copiées dans la mémoire tampon de messages. Etant donné que la conversion de données est également appliquée lorsque les données sont extraites de la mémoire tampon de messages, les applications ne définissent pas cette option.

L'utilisation de cette option peut entraîner des problèmes lors de la conversion de jeux de caractères codés sur un octet en jeux de caractères codés sur deux octets. A la place, effectuez la conversion à l'aide des méthodes `readString`, `readLine`et `writeString` une fois que le message a été distribué.

MQC.MQGMO_FAIL_IF_QUIESCING
Echec si le gestionnaire de files d'attente est au repos.

MQC.MQGMO_LOCK*
Verrouillez le message parcouru.

MQC.MQGMO_LOGICAL_ORDER*
Renvoie des messages dans des groupes et des segments de messages logiques, dans l'ordre logique.

Si vous utilisez l'option `MQGMO_LOGICAL_ORDER` dans un client reconnectable, le code anomalie `MQRC_RECONNECT_INCOMPATIBLE` est renvoyé à l'application.

MQC.MQGMO_MARK_SKIP_BACKOUT*
Permet l'annulation d'une unité de travail sans réinstanciation du message dans la file d'attente.

MQC.MQGMO_MSG_UNDER_CURSOR
Obtenir le message sous le curseur de navigation.

MQC.MQGMO_NONE
Aucune autre option n'a été spécifiée ; toutes les options prennent leurs valeurs par défaut.

MQC.MQGMO_NO_PROPERTIES
Aucune propriété du message, à l'exception des propriétés contenues dans le descripteur de message (ou l'extension), n'est extraite.

MQC.MQGMO_NO_SYNCPOINT
Obtenir un message sans contrôle de point de synchronisation.

MQC.MQGMO_NO_WAIT
Revenez immédiatement s'il n'y a pas de message approprié.

MQC.MQGMO_PROPERTIES_AS_Q_DEF
Extrayez les propriétés de message définies par l'attribut `PropertyControl` de `MQQueue`. L'accès aux propriétés de message dans le descripteur de message ou l'extension n'est pas affecté par l'attribut `PropertyControl` .

MQC.MQGMO_PROPERTIES_COMPATIBILITY
Extrayez les propriétés de message avec le préfixe `mcd`, `jms`, `usr`ou `mqext`, dans les en-têtes `MQRFH2` . Les autres propriétés du message, à l'exception des propriétés contenues dans le descripteur de message ou l'extension, sont supprimées.

MQC.MQGMO_PROPERTIES_FORCE_MQRFH2
Extrayez les propriétés de message, à l'exception des propriétés contenues dans le descripteur de message ou l'extension, dans les en-têtes `MQRFH2` . Utilisez `MQC.MQGMO_PROPERTIES_FORCE_MQRFH2` dans les applications qui s'attendent à extraire des propriétés mais qui ne peuvent pas être modifiées pour utiliser des descripteurs de message.

MQC.MQGMO_PROPERTIES_IN_HANDLE
Extrayez les propriétés de message à l'aide d'un `MsgHandle`.

MQC.MQGMO_SYNCPOINT
Obtenez le message sous contrôle de point de synchronisation. Le message est marqué comme étant indisponible pour d'autres applications, mais il est supprimé de la file d'attente uniquement lorsque l'unité de travail est validée. Le message est de nouveau disponible si l'unité d'oeuvre est annulée.

MQC.MQGMO_SYNCPOINT_IF_PERSISTENT*
Obtenir le message avec le contrôle de point de synchronisation si le message est persistant.

MQC.MQGMO_UNLOCK*
Déverrouiller un message précédemment verrouillé.

MQC.MQGMO_WAIT
Attendez qu'un message arrive.

public string ResolvedQueueName {get;}
Le gestionnaire de files d'attente définit `ResolvedQueueNom` sur le nom local de la file d'attente à partir de laquelle le message a été extrait. `ResolvedQueueNom` est différent du nom utilisé pour ouvrir la file d'attente si une file d'attente alias ou modèle a été ouverte.

public char Segmentation {get;}*
`Segmentation` indique si vous pouvez autoriser la segmentation pour le message extrait. Les valeurs possibles sont les suivantes :

MQC.MQSEG_INHIBITED
Ne pas autoriser la segmentation.

MQC.MQSEG_ALLOWED
Autoriser la segmentation

public byte SegmentStatus {get;}*
`SegmentStatus` est une zone de sortie qui indique si le message extrait est un segment d'un message logique. Si le message est un segment, l'indicateur indique s'il s'agit du dernier segment. Les valeurs possibles sont les suivantes :

MQC.MQSS_LAST_SEGMENT
Le message est le dernier ou le seul segment du message logique.

MQC.MQSS_NOT_A_SEGMENT
Le message n'est pas un segment.

MQC.MQSS_SEGMENT
Le message est un segment, mais n'est pas le dernier segment du message logique.

public int WaitInterval {get; set;}
`WaitInterval` est la durée maximale en millisecondes pendant laquelle un appel `MQQueue.get` attend l'arrivée d'un message approprié. Utilisez `WaitInterval` avec `MQC.MQGMO_WAIT`. Définissez la valeur `MQC.MQWI_UNLIMITED` pour attendre un temps illimité pour un message.

Constructeurs

public MQGetMessageOptions()
Construisez un nouvel objet `MQGetMessageOptions` avec `Options` défini sur `MQC.MQGMO_NO_WAIT`, `WaitInterval` défini sur zéro et `ResolvedQueueName` défini sur vide.

Classe MQManagedObject .NET

Utilisez `MQManagedObject` pour interroger et définir les attributs de `MQDestination`, `MQProcess`, `MQQueueManager`et `MQSubscription`. `MQManagedObject` est une superclasse de ces classes.

Classes

```
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;`

- «Propriétés», à la page 1273
- «Des méthodes», à la page 1274
- «Constructeurs», à la page 1275

Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

public string AlternateUserId {get; set;}
ID utilisateur de remplacement, le cas échéant, défini lors de l'ouverture de la ressource. `AlternateUserID.set` est ignoré lorsqu'il est émis pour un objet ouvert. L'`IDAlternateUser` n'est pas valide pour les abonnements.

public int CloseOptions {get; set;}

Définissez cet attribut pour contrôler la manière dont la ressource est fermée. La valeur par défaut est `MQC.MQCO_NONE`. `MQC.MQCO_NONE` est la seule valeur autorisée pour toutes les ressources autres que les files d'attente dynamiques permanentes, les files d'attente dynamiques temporaires, les abonnements et les rubriques accessibles par les objets qui les ont créées.

Pour les files d'attente et les rubriques, les valeurs supplémentaires suivantes sont autorisées:

MQC.MQCO_DELETE
Supprimez la file d'attente s'il n'y a pas de message.

MQC.MQCO_DELETE_PURGE
Supprimez la file d'attente, en purgeant les messages qu'elle contient.

MQC.MQCO_QUIESCE
Demandez la fermeture de la file d'attente, en recevant un avertissement si des messages subsistent (ce qui permet de les extraire avant la fermeture finale).

Pour les abonnements, les valeurs supplémentaires suivantes sont autorisées:

MQC.MQCO_KEEP_SUB
L'abonnement n'est pas supprimé. Cette option est valide uniquement si l'abonnement d'origine est durable. `MQC.MQCO_KEEP_SUB` est la valeur par défaut d'une rubrique durable.

MQC.MQCO_REMOVE_SUB
L'abonnement est supprimé. `MQC.MQCO_REMOVE_SUB` est la valeur par défaut pour une rubrique non durable et non gérée.

MQC.MQCO_PURGE_SUB
L'abonnement est supprimé. `MQC.MQCO_PURGE_SUB` est la valeur par défaut pour une rubrique gérée non durable.

public MQQueueManager ConnectionReference {get;}
Gestionnaire de files d'attente auquel appartient cette ressource.

public string MQDescription {get;}
Description de la ressource telle qu'elle est conservée par le gestionnaire de files d'attente. `MQDescription` renvoie une chaîne vide pour les abonnements et les rubriques.

public boolean IsOpen {get;}
Indique si la ressource est actuellement ouverte.

public string Name {get;}
Nom de la ressource. Le nom est soit celui indiqué dans la méthode d'accès, soit celui attribué par le gestionnaire de files d'attente à une file d'attente dynamique.

public int OpenOptions {get; set;}
Les OpenOptions sont définies lorsqu'un objet WebSphere MQ est ouvert. La méthode `OpenOptions.set` est ignorée et ne génère pas d'erreur. Les abonnements n'ont pas d' `OpenOptions`.

Des méthodes

public virtual void Close();
Emet `MQException`.

Ferme l'objet. Aucune autre opération sur cette ressource n'est autorisée après l'appel de `Close`. Pour modifier le comportement de la méthode `Close` , définissez l'attribut `closeOptions` .

public string GetAttributeString(int *selector*, int *length*);
Emet `MQException`.

Obtient une chaîne d'attribut.

selector
Entier indiquant l'attribut demandé.

length
Entier indiquant la longueur de la chaîne requise.

public void Inquire(int[] *selectors*, int[] *intAttrs*, byte[] *charAttrs*);
Emet `MQException`.

Renvoie un tableau d'entiers et un ensemble de chaînes de caractères contenant les attributs d'une file d'attente, d'un processus ou d'un gestionnaire de files d'attente. Les attributs à interroger sont spécifiés dans le tableau des sélecteurs.

Remarque : La plupart des attributs les plus courants peuvent être interrogés à l'aide des méthodes `Get` définies dans `MQManagedObject`, `MQQueue` et `MQQueueManager`.

selectors
Tableau d'entiers identifiant les attributs avec les valeurs à identifier.

intAttrs
Tableau dans lequel les valeurs d'attribut entières sont renvoyées. Les valeurs d'attribut de type entier sont renvoyées dans le même ordre que les sélecteurs d'attribut de type entier dans le tableau des sélecteurs.

charAttrs
Mémoire tampon dans laquelle les attributs de caractères sont renvoyés, concaténée. Les attributs de caractère sont renvoyés dans le même ordre que les sélecteurs d'attribut de caractère dans le tableau des sélecteurs. La longueur de chaque chaîne d'attribut est fixe pour chaque attribut.

public void Set(int[] *selectors*, int[] *intAttrs*, byte[] *charAttrs*);
Emet `MQException`.

Définit les attributs définis dans le vecteur des sélecteurs. Les attributs à définir sont spécifiés dans le tableau des sélecteurs.

selectors
Tableau d'entiers identifiant les attributs avec les valeurs à définir.

intAttrs
Tableau de valeurs d'attribut entières à définir. Ces valeurs doivent être dans le même ordre que les sélecteurs d'attribut d'entier dans le tableau des sélecteurs.

charAttrs
Mémoire tampon dans laquelle les attributs de caractères à définir sont concaténés. Ces valeurs doivent être dans le même ordre que les sélecteurs d'attribut de caractères dans le tableau des sélecteurs. La longueur de chaque attribut de caractère est fixe.

public void SetAttributeString(int *selector*, string *value*, int *length*);
Emet `MQException`.

Définit une chaîne d'attribut.

selector
Entier indiquant l'attribut en cours de définition.

value
Chaîne à définir comme valeur d'attribut.

length
Entier indiquant la longueur de la chaîne requise.

Constructeurs

protected MQManagedObject()
Méthode du constructeur. Cet objet est une classe de base abstraite qui ne peut pas être instanciée par elle-même.

Classe MQMessage .NET

Utilisez `MQMessage` pour accéder au descripteur de message et aux données d'un message WebSphere MQ . `MQMessage` encapsule un message WebSphere MQ .

Classe

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQMessage
```

```
public class IBM.WMQ.MQMessage extends IBM.WMQ.MQBaseObject;
```

Créez un objet MQMessage , puis utilisez les méthodes Read et Write pour transférer des données entre le message et d'autres objets de votre application. Envoyez et recevez des objets MQMessage à l'aide des méthodes Put et Get des classes MQDestination, MQQueue et MQTopic .

Obtenez et définissez les propriétés du descripteur de message à l'aide des propriétés de MQMessage. Définissez et obtenez les propriétés de message étendues à l'aide des méthodes SetProperty et GetProperty .

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

public string AccountingToken {get; set;}
Partie du contexte d'identité du message ; elle aide une application à facturer le travail effectué suite au message. La valeur par défaut est MQC.MQACT_NONE.

public string ApplicationIdData {get; set;}
Partie du contexte d'identité du message. Les donnéesApplicationId sont des informations définies par la suite d'applications et peuvent être utilisées pour fournir des informations supplémentaires sur le message ou son émetteur. La valeur par défaut est "".

public string ApplicationOriginData {get; set;}
Informations définies par l'application qui peuvent être utilisées pour fournir des informations supplémentaires sur l'origine du message. La valeur par défaut est "".

public int BackoutCount {get;}
Nombre de fois où le message a été précédemment renvoyé et annulé par un appel MQQueue.Get dans le cadre d'une unité de travail. La valeur par défaut est zéro.

public int CharacterSet {get; set;}

Identificateur de jeu de caractères codés des données de type caractères dans le message.

Définissez CharacterSet pour identifier le jeu de caractères des données de type caractère dans le message. Obtenez CharacterSet pour savoir dans quel jeu de caractères a été utilisé pour coder les données de type caractère dans le message.

Les applications .NET s'exécutent toujours en Unicode, alors que dans d'autres environnements, les applications s'exécutent dans le même jeu de caractères que le gestionnaire de files d'attente.

Les méthodes ReadString et ReadLine convertissent les données de type caractères du message en Unicode pour vous.

La méthode WriteString convertit le format Unicode au jeu de caractères codé dans CharacterSet. Si CharacterSet est défini sur sa valeur par défaut, MQC.MQCCSI_Q_MGR, qui est 0, aucune conversion n'a lieu et CharacterSet est défini sur 1200. Si vous affectez à CharacterSet une autre valeur, WriteString convertit Unicode en valeur de remplacement.

Remarque : Les autres méthodes de lecture et d'écriture n'utilisent pas CharacterSet.

- ReadChar et WriteChar lisent et écrivent un caractère Unicode dans et depuis la mémoire tampon de messages sans conversion.
- ReadUTF et WriteUTF convertissent entre une chaîne Unicode dans l'application et une chaîne UTF-8 , préfixée par une zone de longueur de 2 octets, dans la mémoire tampon de messages.
- Les méthodes Byte transfèrent des octets entre l'application et la mémoire tampon de message sans modification.

public byte[] CorrelationId {get; set;}

- Pour un appel `MQQueue.Get` , identificateur de corrélation du message à extraire. Le gestionnaire de files d'attente renvoie le premier message avec un identificateur de message et un identificateur de corrélation qui correspondent aux zones de descripteur de message. La valeur par défaut, `MQC.MQCI_NONE`, permet à tout identificateur de corrélation de correspondre.
- Pour un appel `MQQueue.Put` , identificateur de corrélation à définir.

public int DataLength {get;}
Nombre d'octets de données de message restant à lire.

public int DataOffset {get; set;}
Position actuelle du curseur dans les données du message. Les lectures et les écritures prennent effet à la position en cours.

public int Encoding {get; set;}
Représentation utilisée pour les valeurs numériques dans les données de message d'application. `Codage` s'applique aux données binaires, décimales condensées et à virgule flottante. Le comportement des méthodes de lecture et d'écriture pour ces formats numériques est modifié en conséquence. Construisez une valeur pour la zone de codage en ajoutant une valeur à partir de chacune de ces trois sections. Vous pouvez également construire la valeur en combinant les valeurs de chacune des trois sections à l'aide de l'opérateur OR bit à bit.

1. Binary Integer

   MQC.MQENC_INTEGER_NORMAL
   Entiers big endian.

   MQC.MQENC_INTEGER_REVERSED
   Entiers little endian, tels qu'utilisés dans l'architecture Intel.

2. Décimal condensé

   MQC.MQENC_DECIMAL_NORMAL
   Format décimal condensé Big-endian, tel qu'il est utilisé par z/OS.

   MQC.MQENC_DECIMAL_REVERSED
   Décimale condensée little endian.

3. virgule flottante

   MQC.MQENC_FLOAT_IEEE_NORMAL
   L'IEEE Big-endian flotte.

   MQC.MQENC_FLOAT_IEEE_REVERSED
   Little-endian IEEE flotte, comme l'architecture Intel utilisée.

   MQC.MQENC_FLOAT_S390
   z/OS : formatage des points flottants.

La valeur par défaut est :

```
MQC.MQENC_INTEGER_REVERSED |
MQC.MQENC_DECIMAL_REVERSED |
MQC.MQENC_FLOAT_IEEE_REVERSED
```

Le paramètre par défaut permet à `WriteInt` d'écrire un entier little-endian et à `ReadInt` de lire un entier little-endian. Si vous définissez l'indicateur `MQC.MQENC_INTEGER_NORMAL` à la place, `WriteInt` écrit un entier big-endian et `ReadInt` lit un entier big-endian.

Remarque : Une perte de précision peut se produire lors de la conversion de points flottants au format IEEE en points flottants au format zSeries .

public int Expiry {get; set;}
Heure d'expiration exprimée en dixièmes de seconde, définie par l'application qui insère le message. Une fois que le délai d'expiration d'un message est écoulé, il peut être supprimé par le gestionnaire de files d'attente. Si le message a spécifié l'un des indicateurs `MQC.MQRO_EXPIRATION` , un rapport

est généré lorsque le message est supprimé. La valeur par défaut est MQC.MQEI_UNLIMITED, ce qui signifie que le message n'expire jamais.

public int Feedback {get; set;}

Utilisez Commentaires avec un message de type MQC.MQMT_REPORT pour indiquer la nature du rapport. Les codes retour suivants sont définis par le système:

- 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

Les valeurs de retour d'informations définies par l'application dans la plage MQC.MQFB_APPL_FIRST à MQC.MQFB_APPL_LAST peuvent également être utilisées. La valeur par défaut de cette zone est MQC.MQFB_NONE, ce qui indique qu'aucun commentaire n'est fourni.

public string Format {get; set;}

Nom de format utilisé par l'expéditeur du message pour indiquer la nature des données du message au destinataire. Vous pouvez utiliser vos propres noms de format, mais les noms commençant par les lettres MQ ont une signification définie par le gestionnaire de files d'attente. Les formats intégrés du gestionnaire de files d'attente sont les suivants:

MQC.MQFMT_ADMIN
Message de demande / réponse du serveur de commandes.

MQC.MQFMT_COMMAND_1
Message de réponse à la commande de type 1.

MQC.MQFMT_COMMAND_2
Message de réponse à la commande de type 2.

MQC.MQFMT_DEAD_LETTER_HEADER
En-tête de rebut.

MQC.MQFMT_EVENT
Message d'événement.

MQC.MQFMT_NONE
Aucun nom de format.

MQC.MQFMT_PCF
Message défini par l'utilisateur au format de commande programmable.

MQC.MQFMT_STRING
Message composé entièrement de caractères.

MQC.MQFMT_TRIGGER
Message de déclenchement

MQC.MQFMT_XMIT_Q_HEADER
En-tête de file d'attente de transmission.

La valeur par défaut est MQC.MQFMT_NONE.

public byte[] GroupId {get; set;}
Chaîne d'octets qui identifie le groupe de messages auquel appartient le message physique. La valeur par défaut est MQC.MQGI_NONE.

public int MessageFlags {get; set;}
Indicateurs contrôlant la segmentation et l'état d'un message.

public byte[] MessageId {get; set;}

Pour un appel MQQueue.Get , cette zone indique l'identificateur du message à extraire. Normalement, le gestionnaire de files d'attente renvoie le premier message avec un identificateur de message et un identificateur de corrélation qui correspondent aux zones de descripteur de message. Autorisez la mise en correspondance de tout identificateur de message à l'aide de la valeur spéciale MQC.MQMI_NONE.

Pour un appel MQQueue.Put , cette zone indique l'identificateur de message à utiliser. Si MQC.MQMI_NONE est spécifié, le gestionnaire de files d'attente génère un identificateur de message unique lors de l'insertion du message. La valeur de cette variable de membre est mise à jour après l'insertion, pour indiquer l'identificateur de message utilisé. La valeur par défaut est MQC.MQMI_NONE.

public int MessageLength {get;}
Nombre d'octets de données de message dans l'objet MQMessage .

public int MessageSequenceNumber {get; set;}
Numéro de séquence d'un message logique au sein d'un groupe.

public int MessageType {get; set;}
Indique le type du message. Les valeurs suivantes sont actuellement définies par le système:

- MQC.MQMT_DATAGRAM
- MQC.MQMT_REPLY
- MQC.MQMT_REPORT
- MQC.MQMT_REQUEST

Les valeurs définies par l'application peuvent également être utilisées, dans la plage MQC.MQMT_APPL_FIRST à MQC.MQMT_APPL_LAST. La valeur par défaut de cette zone est MQC.MQMT_DATAGRAM.

public int Offset {get; set;}
Dans un message segmenté, décalage des données dans un message physique à partir du début d'un message logique.

public int OriginalLength {get; set;}
Longueur d'origine d'un message segmenté.

public int Persistence {get; set;}

Persistance des messages. Les valeurs suivantes sont définies :

- MQC.MQPER_NOT_PERSISTENT

  Si vous définissez cette option dans un client reconnectable, le code anomalieMQRC_NONE est renvoyé à l'application lorsque la connexion aboutit.

- MQC.MQPER_PERSISTENT

  Si vous définissez cette option dans un client reconnectable, le code anomalie MQRC_CALL_INTERRUPTED est renvoyé à l'application une fois la connexion établie.

- MQC.MQPER_PERSISTENCE_AS_Q_DEF

La valeur par défaut est MQC.MQPER_PERSISTENCE_AS_Q_DEF, qui extrait la persistance du message de l'attribut de persistance par défaut de la file d'attente de destination.

public int Priority {get; set;}
Priorité du message. La valeur spéciale MQC.MQPRI_PRIORITY_AS_Q_DEF peut également être définie dans le message sortant. La priorité du message est ensuite extraite de

l'attribut de priorité par défaut de la file d'attente de destination. La valeur par défaut est MQC.MQPRI_PRIORITY_AS_Q_DEF.

public int PropertyValidation {get; set;}
Indique si la validation des propriétés a lieu lorsqu'une propriété du message est définie. Les valeurs possibles sont les suivantes :

- MQCMHO_DEFAULT_VALIDATION
- MQCMHO_VALIDATE
- MQCMHO_NO_VALIDATION

La valeur par défaut est MQCMHO_DEFAULT_VALIDATION.

public string PutApplicationName {get; set;}
Nom de l'application qui a inséré le message. La valeur par défaut est "".

public int PutApplicationType {get; set;}
Type de l'application ayant placé le message en file d'attente. PutApplicationType peut être une valeur définie par le système ou par l'utilisateur. Les valeurs suivantes sont définies par le système:

- 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

La valeur par défaut est MQC.MQAT_NO_CONTEXT, ce qui indique qu'aucune information contextuelle n'est présente dans le message.

public DateTime PutDateTime {get; set;}
Date et heure à laquelle le message a été inséré.

public string ReplyToQueueManagerName {get; set;}
Nom du gestionnaire de files d'attente pour l'envoi des messages de réponse ou de rapport. La valeur par défaut est ""et le gestionnaire de files d'attente fournit le nom ReplyToQueueManager.

public string ReplyToQueueName {get; set;}
Nom de la file d'attente de messages à laquelle l'application qui a émis la demande d'obtention du message envoie des messages MQC.MQMT_REPLY et MQC.MQMT_REPORT . Le ReplyToQueueName par défaut est "".

public int Report {get; set;}
Utilisez Rapport pour spécifier les options relatives aux messages de rapport et de réponse:

- Indique si des rapports sont requis.
- Indique si les données de message d'application doivent être incluses dans les rapports.
- Comment définir les identificateurs de message et de corrélation dans le rapport ou la réponse.

Toute combinaison des quatre types de rapport peut être demandée:

- Indiquez une combinaison des quatre types de rapport. Sélection de l'une des trois options pour chaque type de rapport, selon que les données de message d'application doivent ou non être incluses dans le message de rapport.

  1. Confirmer à réception

- MQC.MQRO_COA
- MQC.MQRO_COA_WITH_DATA
- MQC.MQRO_COA_WITH_FULL_DATA**

2. Confirmer à expédition
   - MQC.MQRO_COD
   - MQC.MQRO_COD_WITH_DATA
   - MQC.MQRO_COD_WITH_FULL_DATA**
3. Exception
   - MQC.MQRO_EXCEPTION
   - MQC.MQRO_EXCEPTION_WITH_DATA
   - MQC.MQRO_EXCEPTION_WITH_FULL_DATA**
4. Expiration dans
   - MQC.MQRO_EXPIRATION
   - MQC.MQRO_EXPIRATION_WITH_DATA
   - MQC.MQRO_EXPIRATION_WITH_FULL_DATA**

Remarque : Les valeurs signalées par ** dans la liste ne sont pas prises en charge par les gestionnaires de files d'attente z/OS . Ne les utilisez pas si votre application est susceptible d'accéder à un gestionnaire de files d'attente z/OS , quelle que soit la plateforme sur laquelle l'application s'exécute.

- Indiquez l'une des options suivantes pour contrôler la façon dont l'ID message est généré pour le message de rapport ou de réponse:
  - MQC.MQRO_NEW_MSG_ID
  - MQC.MQRO_PASS_MSG_ID
- Indiquez l'une des options suivantes pour contrôler la façon dont l'ID de corrélation du message de rapport ou de réponse doit être défini:
  - MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID
  - MQC.MQRO_PASS_CORREL_ID
- Indiquez l'une des options suivantes pour contrôler la disposition du message d'origine lorsqu'il ne peut pas être distribué à la file d'attente de destination:
  - MQC.MQRO_DEAD_LETTER_Q
  - MQC.MQRO_DISCARD_MSG**
- Si aucune option de rapport n'est spécifiée, la valeur par défaut est:

```
MQC.MQRO_NEW_MSG_ID |
MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID |
MQC.MQRO_DEAD_LETTER_Q
```

- Vous pouvez spécifier l'un des éléments suivants ou les deux pour demander à l'application de réception d'envoyer un message de rapport d'action positive ou négative.
  - MQC.MQRO_PAN
  - MQC.MQRO_NAN

public int TotalMessageLength {get;}

Nombre total d'octets du message tels qu'ils sont stockés dans la file d'attente de messages à partir de laquelle ce message a été reçu.

public string UserId {get; set;}
UserId fait partie du contexte d'identité du message. Le gestionnaire de files d'attente fournit généralement la valeur. Vous pouvez remplacer la valeur si vous avez le droit de définir le contexte d'identité.

public int Version {get; set;}
Version de la structure MQMD utilisée.

Méthodes de message Read et Write

Les méthodes Read et Write exécutent les mêmes fonctions que les membres des classes BinaryReader et BinaryWriter dans l'espace de nom .NET System.IO . Voir MSDN pour obtenir des exemples complets de syntaxe et d'utilisation du langage. Les méthodes lisent ou écrivent à partir de la position en cours dans la mémoire tampon du message. Ils déplacent la position actuelle vers l'avant en fonction du nombre d'octets lus ou écrits.

Remarque : Si les données de message contiennent un en-tête MQRFH ou MQRFH2 , vous devez utiliser la méthode ReadBytes pour lire les données.

- Toutes les méthodes émettent IOException.
- Les méthodes ReadFully redimensionnent automatiquement le tableau byte ou sbyte cible pour qu'il corresponde exactement au message. Un tableau null est également redimensionné.
- Les méthodes Read émettent EndOfStreamException.
- Les méthodes WriteDecimal émettent MQException.
- Les méthodes ReadString, ReadLine et WriteString sont converties entre Unicode et le jeu de caractères du message ; voir CharacterSet.
- Les méthodes Decimal lisent et écrivent des nombres décimaux condensés codés au format big-endian, MQC.MQENC_DECIMAL_NORMALou little-endian MQC.MQENC_DECIMAL_REVERSE , en fonction de la valeur de Encoding. Les plages décimales et les types .NET correspondants sont les suivants:

  Decimal2/short
  -999 à 999

  Decimal4/int
  -9999999 à 9999999

  Decimal8/long
  -999999999999999 à 999999999999999
- Les méthodes Double et Float lisent et écrivent des valeurs flottantes codées aux formats IEE big-endian et little-endian, MQC.MQENC_FLOAT_IEEE_NORMAL et MQC.MQENC_FLOAT_IEEE_REVERSED, ou au format S/390 , MQC.MQENC_FLOAT_S390, en fonction de la valeur de Encoding.
- Les méthodes Int lisent et écrivent des valeurs entières codées au format big-endian, MQC.MQENC_INTEGER_NORMALou little-endian, MQC.MQENC_INTEGER_REVERSED, en fonction de la valeur de Encoding. Les entiers sont tous signés, à l'exception de l'ajout d'un type entier de 2 octets non signé. Les tailles d'entier et les types .NET et WebSphere MQ sont les suivants:

  2 octets
  short, Int2, ushort, UInt2

  4 octets
  int, Int4

  8 octets
  long, Int8
- WriteObject transfère la classe d'un objet, les valeurs de ses zones non transitoires et non statiques et les zones de ses supertypes, dans la mémoire tampon de messages.
- ReadObject crée un objet à partir de la classe de l'objet, de la signature de la classe et des valeurs de ses zones non transitoires et non statiques, ainsi que des zones de ses supertypes.

Tableau 608. Méthodes de lecture et d'écriture de messages

Type de cible	Signatures de méthode

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)`

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)`

Tableau 608. Méthodes de lecture et d'écriture de messages (suite)

Type de cible	Signatures de méthode

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)`

Méthodes de mémoire tampon

public void ClearMessage();

Emet IOException.

Supprime toutes les données de la mémoire tampon de messages et redéfinit le décalage des données sur zéro.

public void ResizeBuffer(int *size*)

Emet IOException.

Suggestion à l'objet MQMessage concernant la taille de la mémoire tampon pouvant être requise pour les opérations d'extraction suivantes. Si le message contient actuellement des données de message et que la nouvelle taille est inférieure à la taille en cours, les données de message sont tronquées.

public void Seek(int *pos*)

Emet IOException, ArgumentOutOfRangeException, ArgumentException.

Déplace le curseur à la position absolue dans la mémoire tampon de messages indiquée par *pos*. Les lectures et écritures suivantes agissent à cet emplacement dans la mémoire tampon.

public int SkipBytes(int *i*)

Emet IOException, EndOfStreamException.

Avance de n octets dans la mémoire tampon du message et renvoie n, le nombre d'octets ignorés.

La méthode SkipBytes se bloque jusqu'à ce que l'un des événements suivants se produise:

- Tous les octets sont ignorés
- La fin de la mémoire tampon de messages est détectée
- Une exception est émise

Méthodes de propriété

public void DeleteProperty(string *name*);

Emet MQException.

Supprime du message une propriété portant le nom spécifié.

name
Nom de la propriété à supprimer.

public System.Collections.IEnumerator GetPropertyNames(string *name*)

Emet MQException.

Renvoie un IEnumerator de tous les noms de propriété correspondant au nom spécifié. Le signe de pourcentage '%' peut être utilisé à la fin du nom en tant que caractère générique pour filtrer les propriétés du message, correspondant à zéro ou à plusieurs caractères, y compris le point.

name
Nom de la propriété à mettre en correspondance.

- Toutes les méthodes SetProperty et GetProperty émettent MQException

Tableau 609. Méthodes SetProperty et GetProperty

Tapez	Signatures de méthode

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);`

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);`

Tableau 609. Méthodes SetProperty et GetProperty (suite)

Tapez	Signatures de méthode

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);`

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);`

Tableau 609. Méthodes SetProperty et GetProperty (suite)	
Tapez	**Signatures de méthode**

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);`

Constructeurs

public MQMessage();

Crée un objet MQMessage avec des informations de descripteur de message par défaut et une mémoire tampon de message vide.

Classe MQProcess .NET

Utilisez MQProcess pour interroger les attributs d'un processus WebSphere MQ . Créez un objet MQProcess à l'aide d'un constructeur ou d'une méthode MQQueueManager AccessProcess .

Classe

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQManagedObject
                └ IBM.WMQ.MQProcess
```

```
public class IBM.WMQ.MQProcess extends IBM.WMQ.MQManagedObject;
```

- «Propriétés», à la page 1287
- «Constructeurs», à la page 1288

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

public string ApplicationId {get;}

Extrait la chaîne de caractères qui identifie l'application à démarrer. ApplicationId est utilisé par une application de moniteur de déclenchement. ApplicationId est envoyé à la file d'attente d'initialisation dans le cadre du message de déclenchement.

La valeur par défaut est Null (indéfinie).

public int ApplicationType {get;}

Identifie le type de processus à démarrer par une application de moniteur de déclenchement. Des types standard sont définis, mais d'autres types peuvent être utilisés:

- MQAT_AIX
- MQAT_CICS
- MQAT_IMS
- MQAT_MVS
- MQAT_NATIVE

- MQAT_OS400
- MQAT_UNIX
- MQAT_WINDOWS
- MQAT_JAVA
- MQAT_USER_FIRST
- MQAT_USER_LAST

La valeur par défaut est MQAT_NATIVE.

public string EnvironmentData {get;}

Permet d'obtenir des informations sur l'environnement de l'application à démarrer.

La valeur par défaut est Null (indéfinie).

public string UserData {get;}

Obtient les informations fournies par l'utilisateur sur l'application à démarrer.

La valeur par défaut est Null (indéfinie).

Constructeurs

public MQProcess(MQQueueManager *queueManager*, string *processName*, int *openOptions*);
public MQProcess(MQQueueManager *qMgr*, string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

Emet MQException.

Accédez à un processus WebSphere MQ sur le gestionnaire de files d'attente *qMgr* pour interroger les attributs de processus.

qMgr
Gestionnaire de files d'attente auquel accéder.

processName
Nom du processus à ouvrir.

openOptions
Options qui contrôlent l'ouverture du processus. Les options valides qui peuvent être ajoutées ou combinées à l'aide d'un OR bit à bit sont les suivantes:

- MQC.MQOO_FAIL_IF_QUIESCING
- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

queueManagerName
Nom du gestionnaire de files d'attente sur lequel le processus est défini. Vous pouvez laisser un nom de gestionnaire de files d'attente vide ou null si le gestionnaire de files d'attente est identique à celui auquel le processus accède.

alternateUserId

Si MQC.MQOO_ALTERNATE_USER_AUTHORITY est spécifié dans le paramètre *openOptions* , *alternateUserId* indique l'autre ID utilisateur utilisé pour vérifier l'autorisation de l'action. Si MQOO_ALTERNATE_USER_AUTHORITY n'est pas spécifié, *alternateUserId* peut être vide ou null.

Les droits utilisateur par défaut sont utilisés pour la connexion au gestionnaire de files d'attente si MQC.MQOO_ALTERNATE_USER_AUTHORITY n'est pas spécifié.

public MQProcess MQQueueManager.AccessProcess(string *processName*, int *openOptions*);
public MQProcess MQQueueManager.AccessProcess(string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

Emet MQException.

Accédez à un processus WebSphere MQ sur ce gestionnaire de files d'attente pour consulter les attributs de processus.

processName
Nom du processus à ouvrir.

openOptions
Options qui contrôlent l'ouverture du processus. Les options valides qui peuvent être ajoutées ou combinées à l'aide d'un OR bit à bit sont les suivantes:

- MQC.MQOO_FAIL_IF_QUIESCING
- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

queueManagerName
Nom du gestionnaire de files d'attente sur lequel le processus est défini. Vous pouvez laisser un nom de gestionnaire de files d'attente vide ou null si le gestionnaire de files d'attente est identique à celui auquel le processus accède.

alternateUserId

Si MQC.MQOO_ALTERNATE_USER_AUTHORITY est spécifié dans le paramètre *openOptions* , *alternateUserId* indique l'autre ID utilisateur utilisé pour vérifier l'autorisation de l'action. Si MQOO_ALTERNATE_USER_AUTHORITY n'est pas spécifié, *alternateUserId* peut être vide ou null.

Les droits utilisateur par défaut sont utilisés pour la connexion au gestionnaire de files d'attente si MQC.MQOO_ALTERNATE_USER_AUTHORITY n'est pas spécifié.

Classe MQPropertyDescriptor .NET

Utilisez MQPropertyDescriptor comme paramètre pour les méthodes MQMessage GetProperty et SetProperty . MQPropertyDescriptor décrit une propriété MQMessage .

Classe

```
System.Object
        |
        └─ IBM.WMQ.MQPropertyDescriptor
```

```
public class IBM.WMQ.MQPropertyDescriptor extends System.Object;
```


- «Propriétés», à la page 1289
- «Constructeurs», à la page 1291

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

public int Context {get; set;}

Contexte de message auquel appartient la propriété. Les valeurs possibles sont les suivantes :

MQC.MQPD_NO_CONTEXT
La propriété n'est pas associée à un contexte de message.

MQC.MQPD_USER_CONTEXT
La propriété est associée au contexte utilisateur.

Si l'utilisateur est autorisé, une propriété associée au contexte utilisateur est sauvegardée lorsqu'un message est extrait. Une méthode Put ultérieure référençant le contexte sauvegardé peut transmettre la propriété dans le nouveau message.

public int CopyOptions {get; set;}

CopyOptions décrit le type de message dans lequel la propriété peut être copiée.

Lorsqu'un gestionnaire de files d'attente reçoit un message contenant une propriété WebSphere MQ définie que le gestionnaire de files d'attente reconnaît comme étant incorrecte, il corrige la valeur de la zone CopyOptions .

Vous pouvez indiquer n'importe quelle combinaison des options suivantes. Combinez les options en ajoutant les valeurs ou en utilisant le ORbit à bit.

MQC.MQCOPY_ALL
La propriété est copiée dans tous les types de messages suivants.

MQC.MQCOPY_FORWARD
La propriété est copiée dans un message en cours de transfert.

MQC.MQCOPY_PUBLISH
La propriété est copiée dans le message reçu par un abonné lorsqu'un message est publié.

MQC.MQCOPY_REPLY
La propriété est copiée dans un message de réponse.

MQC.MQCOPY_REPORT
La propriété est copiée dans un message de rapport.

MQC.MQCOPY_DEFAULT
La valeur indiquée indique qu'aucune autre option de copie n'a été spécifiée. Il n'existe aucune relation entre la propriété et les messages suivants. MQC.MQCOPY_DEFAULT est toujours renvoyé pour les propriétés de descripteur de message.

MQC.MQCOPY_NONE
Identique à MQC.MQCOPY_DEFAULT

public int Options { set; }
Options prend par défaut la valeur CMQC.MQPD_NONE. Vous ne pouvez pas définir d'autre valeur.

public int Support { get; set; }

Définissez Support pour spécifier le niveau de prise en charge requis pour les propriétés de message définies par WebSphere MQ. La prise en charge de toutes les autres propriétés est facultative. Aucune des valeurs suivantes ne peut être spécifiée

MQC.MQPD_SUPPORT_OPTIONAL
La propriété est acceptée par un gestionnaire de files d'attente même si elle n'est pas prise en charge. La propriété peut être supprimée pour que le message soit transmis à un gestionnaire de files d'attente qui ne prend pas en charge les propriétés de message. Cette valeur est également affectée aux propriétés qui ne sont pas WebSphere MQ définies.

MQC.MQPD_SUPPORT_REQUIRED
La prise en charge de la propriété est requise. Si vous placez le message dans un gestionnaire de files d'attente qui ne prend pas en charge la propriété WebSphere MQdéfinie, la méthode échoue. Elle renvoie le code achèvement MQC.MQCC_FAILED et le code anomalie MQC.MQRC_UNSUPPORTED_PROPERTY.

MQC.MQPD_SUPPORT_REQUIRED_IF_LOCAL

La prise en charge de la propriété est requise si le message est destiné à une file d'attente locale. Si vous placez le message dans une file d'attente locale d'un gestionnaire de files d'attente qui ne prend pas en charge la propriété WebSphere MQdéfinie, la méthode échoue. Elle renvoie le code achèvement MQC.MQCC_FAILED et le code anomalie MQC.MQRC_UNSUPPORTED_PROPERTY.

Aucune vérification n'est effectuée si le message est inséré dans un gestionnaire de files d'attente éloignées.

Constructeurs

PropertyDescriptor();
Créez un descripteur de propriété.

Classe MQPutMessageOptions .NET

Utilisez MQPutMessageOptions pour spécifier le mode d'envoi des messages. Il modifie le comportement de MQDestination.Put.

Classe

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQPutMessageOptions
```

```
public class IBM.WMQ.MQPutMessageOptions extends IBM.WMQ.MQBaseObject;
```

- «Propriétés», à la page 1291«Constructeurs», à la page 1293

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

Remarque : Le comportement de certaines des options disponibles dans cette classe dépend de l'environnement dans lequel elles sont utilisées. Ces éléments sont marqués d'un astérisque, *.

public MQQueue ContextReference {get; set;}

Si la zone options inclut MQC.MQPMO_PASS_IDENTITY_CONTEXT ou MQC.MQPMO_PASS_ALL_CONTEXT, définissez cette zone pour qu'elle fasse référence au MQQueue à partir duquel les informations de contexte doivent être prises.

La valeur initiale de cette zone est null.

public int InvalidDestCount {get;}*

Généralement, utilisé pour les listes de distribution, InvalidDestCount indique le nombre de messages qui n'ont pas pu être envoyés aux files d'attente d'une liste de distribution. Le nombre inclut les files d'attente dont l'ouverture a échoué, ainsi que les files d'attente dont l'ouverture a abouti, mais pour lesquelles l'opération d'insertion a échoué.

.NET ne prend pas en charge les listes de distribution, mais InvalidDestCount est défini lors de l'ouverture d'une file d'attente unique.

public int KnownDestCount {get;}*

Généralement utilisé pour les listes de distribution, KnownDestCount indique le nombre de messages que l'appel en cours a envoyés avec succès à des files d'attente qui se résolvent en files d'attente locales.

.NET ne prend pas en charge les listes de distribution, mais InvalidDestCount est défini lors de l'ouverture d'une file d'attente unique.

public int Options {get; set;}

Options qui contrôlent l'action de MQDestination.put et de MQQueueManager.put. Aucune des valeurs suivantes ne peut être indiquée. Si plusieurs options sont requises, les valeurs peuvent être ajoutées ou combinées à l'aide de l'opérateur OR bit à bit.

MQC.MQPMO_ASYNC_RESPONSE
Cette option permet d'effectuer l'appel MQDestination.put de manière asynchrone avec des données de réponse.

MQC.MQPMO_DEFAULT_CONTEXT
Associez le contexte par défaut au message.

MQC.MQPMO_FAIL_IF_QUIESCING
Echec si le gestionnaire de files d'attente est au repos.

MQC.MQPMO_LOGICAL_ORDER*
Placez les messages logiques et les segments des groupes de messages dans leur ordre logique.

Si vous utilisez l'option MQPMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé à l'application.

MQC.MQPMO_NEW_CORREL_ID*
Générez un nouvel ID de corrélation pour chaque message envoyé.

MQC.MQPMO_NEW_MSG_ID*
Générez un nouvel ID message pour chaque message envoyé.

MQC.MQPMO_NONE
Aucune option spécifiée. Ne pas utiliser avec d'autres options.

MQC.MQPMO_NO_CONTEXT
Aucun contexte ne doit être associé au message.

MQC.MQPMO_NO_SYNCPOINT
Placez un message sans contrôle de point de synchronisation. Si l'option de contrôle du point de synchronisation n'est pas spécifiée, la valeur par défaut est "aucun point de synchronisation".

MQC.MQPMO_PASS_ALL_CONTEXT
Transmettez tout le contexte à partir d'un descripteur de file d'attente d'entrée.

MQC.MQPMO_PASS_IDENTITY_CONTEXT
Transmettez le contexte d'identité à partir d'un descripteur de file d'attente d'entrée.

MQC.MQPMO_RESPONSE_AS_Q_DEF

Pour un appel MQDestination.put , cette option prend le type de réponse d'insertion de l'attribut DEFPRESP de la file d'attente.

Pour un appel MQQueueManager.put , cette option permet d'effectuer l'appel de manière synchrone.

MQC.MQPMO_RESPONSE_AS_TOPIC_DEF
MQC.MQPMO_RESPONSE_AS_TOPIC_DEF est un synonyme de MQC.MQPMO_RESPONSE_AS_Q_DEF à utiliser avec les objets de rubrique.

MQC.MQPMO_RETAIN

La publication envoyée doit être conservée par le gestionnaire de files d'attente. Si cette option est utilisée et que la publication ne peut pas être conservée, le message n'est pas publié et l'appel échoue avec MQC.MQRC_PUT_NOT_RETAINED.

Demandez une copie de cette publication après l'heure à laquelle elle a été publiée, en appelant la méthode MQSubscription.RequestPublicationUpdate . La publication sauvegardée est envoyée aux applications qui créent un abonnement sans définir l'option MQC.MQSO_NEW_PUBLICATIONS_ONLY . Vérifiez la propriété de message MQIsRetained d'une publication, lorsqu'elle est reçue, pour déterminer s'il s'agit de la publication conservée.

Lorsque des publications conservées sont demandées par un abonné, l'abonnement utilisé peut contenir un caractère générique dans la chaîne de rubrique. S'il existe plusieurs publications

conservées dans l'arborescence de rubriques qui correspondent à l'abonnement, elles sont toutes envoyées.

MQC.MQPMO_SET_ALL_CONTEXT
Définissez tous les contextes à partir de l'application.

MQC.MQPMO_SET_IDENTITY_CONTEXT
Définissez le contexte d'identité à partir de l'application.

MQC.MQPMO_SYNC_RESPONSE
Cette option permet d'effectuer l'appel `MQDestination.put` ou `MQQueueManager.put` de manière synchrone, avec des données de réponse complètes.

MQC.MQPMO_SUPPRESS_REPLYTO
Les informations renseignées dans les zones `ReplyToQueueName` et `ReplyToQueueManagerName` de la publication ne sont pas transmises aux abonnés. Si cette option est utilisée en combinaison avec une option de rapport qui requiert un `ReplyToQueueName`, l'appel échoue avec `MQC.MQRC_MISSING_REPLY_TO_Q`.

MQC.MQPMO_SYNCPOINT
Placez un message avec le contrôle de point de synchronisation. Le message n'est pas visible en dehors de l'unité de travail tant que l'unité de travail n'est pas validée. Si l'unité d'oeuvre est annulée, le message est supprimé.

public int RecordFields {get; set;}*

Informations sur les listes de distribution. Les listes de distribution ne sont pas prises en charge dans .NET.

public string ResolvedQueueManagerName {get;}

Zone de sortie définie par le gestionnaire de files d'attente sur le nom du gestionnaire de files d'attente propriétaire de la file d'attente spécifiée par le nom de la file d'attente éloignée. `ResolvedQueueManagerName` peut être différent du nom du gestionnaire de files d'attente à partir duquel la file d'attente a été consultée si la file d'attente est une file d'attente éloignée.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique. Si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

public string ResolvedQueueName {get;}

Zone de sortie définie par le gestionnaire de files d'attente sur le nom de la file d'attente dans laquelle le message est placé. `ResolvedQueueNom` peut être différent du nom utilisé pour ouvrir la file d'attente si la file d'attente ouverte était un alias ou une file d'attente modèle.

Une valeur non vide est renvoyée uniquement si l'objet est une file d'attente unique. Si l'objet est une liste de distribution ou une rubrique, la valeur renvoyée n'est pas définie.

public int UnknownDestCount {get;}*

Généralement utilisé pour les listes de distribution, `UnknownDestCount` est une zone de sortie définie par le gestionnaire de files d'attente. Il indique le nombre de messages que l'appel en cours a envoyés avec succès aux files d'attente qui se résolvent en files d'attente éloignées.

.NET ne prend pas en charge les listes de distribution, mais `InvalidDestCount` est défini lors de l'ouverture d'une file d'attente unique.

Constructeurs

public MQPutMessageOptions();

Construisez un nouvel objet `MQPutMessageOptions` sans option définie et un nom `ResolvedQueue` et un nom `ResolvedQueueManagerName`vide.

Classe MQQueue .NET

Utilisez MQQueue pour envoyer et recevoir des messages et des attributs de requête d'une file d'attente WebSphere MQ . Créez un objet MQQueue à l'aide d'un constructeur ou d'une méthode MQQueueManager.AccessProcess .

Classe

```
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;
```

- «Propriétés», à la page 1294
- «Des méthodes», à la page 1296
- «Constructeurs», à la page 1299

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

public int ClusterWorkLoadPriority {get;}
Indique la priorité de la file d'attente. Ce paramètre est valide uniquement pour les files d'attente locales, distantes et alias.

public int ClusterWorkLoadRank {get;}
Indique le rang de la file d'attente. Ce paramètre est valide uniquement pour les files d'attente locales, distantes et alias.

public int ClusterWorkLoadUseQ {get;}
Indique le comportement d'une opération MQPUT lorsque la file d'attente cible comporte une instance locale et au moins une instance de cluster distant. Ce paramètre ne s'applique pas si MQPUT provient d'un canal de cluster. Ce paramètre est valide uniquement pour les files d'attente locales.

public DateTime CreationDateTime {get;}
Date et heure de création de cette file d'attente.

public int CurrentDepth {get;}
Extrait le nombre de messages actuellement dans la file d'attente. Cette valeur est incrémentée lors d'un appel d'insertion et lors de l'annulation d'un appel d'extraction. Il est décrémenté lors d'une opération get non-browse et lors de l'annulation d'un appel put.

public int DefinitionType {get;}
Mode de définition de la file d'attente. Les valeurs possibles sont les suivantes:

- MQC.MQQDT_PREDEFINED
- MQC.MQQDT_PERMANENT_DYNAMIC
- MQC.MQQDT_TEMPORARY_DYNAMIC

public int InhibitGet {get; set;}
Contrôle si vous pouvez obtenir des messages dans cette file d'attente ou pour cette rubrique. Les valeurs possibles sont les suivantes:

- MQC.MQQA_GET_INHIBITED
- MQC.MQQA_GET_ALLOWED

public int InhibitPut {get; set;}
Contrôle si vous pouvez placer des messages dans cette file d'attente ou pour cette rubrique. Les valeurs possibles sont les suivantes:

- MQQA_PUT_INHIBITED
- MQQA_PUT_ALLOWED

public int MaximumDepth {get;}
Nombre maximal de messages pouvant exister simultanément dans la file d'attente. Une tentative d'insertion d'un message dans une file d'attente qui contient déjà ce nombre de messages échoue avec le code anomalie MQC.MQRC_Q_FULL.

public int MaximumMessageLength {get;}
Longueur maximale des données d'application pouvant exister dans chaque message de cette file d'attente. Une tentative d'insertion d'un message supérieur à cette valeur échoue avec le code anomalie MQC.MQRC_MSG_TOO_BIG_FOR_Q.

public int NonPersistentMessageClass {get;}
Niveau de fiabilité des messages non persistants insérés dans cette file d'attente.

public int OpenInputCount {get;}
Nombre de descripteurs actuellement valides pour supprimer des messages de la file d'attente. OpenInputCount est le nombre total de descripteurs d'entrée valides connus du gestionnaire de files d'attente local, et pas seulement les descripteurs créés par l'application.

public int OpenOutputCount {get;}
Nombre de descripteurs actuellement valides pour l'ajout de messages à la file d'attente. OpenOutputNombre est le nombre total de descripteurs de sortie valides connus du gestionnaire de files d'attente local, et pas seulement les descripteurs créés par l'application.

public int QueueAccounting {get;}
Indique si vous pouvez activer la collecte des informations de comptabilité pour la file d'attente.

public int QueueMonitoring {get;}
Indique si vous pouvez activer la surveillance de la file d'attente.

public int QueueStatistics {get;}
Indique si vous pouvez activer la collecte de statistiques pour la file d'attente.

public int QueueType {get;}
Type de cette file d'attente avec l'une des valeurs suivantes:

- MQC.MQQT_ALIAS
- MQC.MQQT_LOCAL
- MQC.MQQT_REMOTE
- MQC.MQQT_CLUSTER

public int Shareability {get;}
Indique si la file d'attente peut être ouverte en entrée plusieurs fois. Les valeurs possibles sont les suivantes:

- MQC.MQQA_SHAREABLE
- MQC.MQQA_NOT_SHAREABLE

public string TPIPE {get;}
Nom TPIPE utilisé pour la communication avec OTMA à l'aide de la passerelle WebSphere MQ IMS .

public int TriggerControl {get; set;}
Indique si les messages de déclenchement sont écrits dans une file d'attente d'initialisation, afin de démarrer une application pour traiter la file d'attente. Les valeurs possibles sont les suivantes:

- MQC.MQTC_OFF
- MQC.MQTC_ON

public string TriggerData {get; set;}
Données de format libre que le gestionnaire de files d'attente insère dans le message de déclenchement. Il insère `TriggerData` lorsqu'un message arrivant dans cette file d'attente entraîne l'écriture d'un message de déclenchement dans la file d'attente d'initialisation. La longueur maximale autorisée de la chaîne est donnée par `MQC.MQ_TRIGGER_DATA_LENGTH`.

public int TriggerDepth {get; set;}
Nombre de messages qui doivent être dans la file d'attente avant qu'un message de déclenchement ne soit écrit lorsque le type de déclencheur est défini sur `MQC.MQTT_DEPTH`.

public int TriggerMessagePriority {get; set;}
Priorité de message sous laquelle les messages ne contribuent pas à la génération de messages de déclenchement. Autrement dit, le gestionnaire de files d'attente ignore ces messages lorsqu'il décide de générer ou non un déclencheur. La valeur zéro entraîne la contribution de tous les messages à la génération de messages de déclenchement.

public int TriggerType {get; set;}
Conditions dans lesquelles les messages de déclenchement sont écrits suite à l'arrivée de messages dans cette file d'attente. Les valeurs possibles sont les suivantes:

- `MQC.MQTT_NONE`
- `MQC.MQTT_FIRST`
- `MQC.MQTT_EVERY`
- `MQC.MQTT_DEPTH`

Des méthodes

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);

Emet `MQException`.

Extrait un message d'une file d'attente.

Si l'extraction échoue, l'objet `MQMessage` n'est pas modifié. Si elle aboutit, le descripteur de message et les parties de données de message du `MQMessage` sont remplacés par le descripteur de message et les données de message du message entrant.

Tous les appels à WebSphere MQ à partir d'un `MQQueueManager` particulier sont synchrones. Par conséquent, si vous effectuez une opération d'extraction avec attente, toutes les autres unités d'exécution utilisant le même `MQQueueManager` sont bloquées pour effectuer d'autres appels WebSphere MQ jusqu'à ce que l'appel Get soit effectué. Si vous avez besoin de plusieurs unités d'exécution pour accéder simultanément à WebSphere MQ , chaque unité d'exécution doit créer son propre objet `MQQueueManager` .

message
Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée `MessageId` et `CorrelationId` sont définis comme requis.

Un client reconnectable renvoie le code anomalie `MQRC_BACKED_OUT` après une reconnexion réussie, pour les messages reçus sous `MQGM_SYNCPOINT`.

getMessageOptions
Options contrôlant l'action de la commande get.

L'utilisation de l'option `MQC.MQGMO_CONVERT` peut générer une exception avec le code anomalie `MQC.MQRC_CONVERTED_STRING_TOO_BIG` lors de la conversion de codes de caractères codés sur un octet en codes codés sur deux octets. Dans ce cas, le message est copié dans la mémoire tampon sans conversion.

Si *getMessageOptions* n'est pas spécifié, l'option de message utilisée est `MQGMO_NOWAIT`.

Si vous utilisez l'option MQGMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

MaxMsgSize
Message le plus volumineux que cet objet message doit recevoir. Si le message de la file d'attente est supérieur à cette taille, l'un des deux événements suivants se produit:

- Si l'indicateur MQGMO_ACCEPT_TRUNCATED_MSG est défini dans l'objet MQGetMessageOptions , le message est rempli avec autant de données de message que possible. Une exception est émise avec le code achèvement MQCC_WARNING et le code anomalie MQRC_TRUNCATED_MSG_ACCEPTED .
- Si l'indicateur MQGMO_ACCEPT_TRUNCATED_MSG n'est pas défini, le message reste dans la file d'attente. Une exception est émise avec le code achèvement MQCC_WARNING et le code anomalie MQRC_TRUNCATED_MSG_FAILED .

Si *MaxMsgSize* n'est pas spécifié, l'intégralité du message est extraite.

```
public void Put(MQMessage message);
public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);
```

Emet MQException.

Insère un message dans une file d'attente.

Les modifications apportées à l'objet MQMessage une fois l'appel Put effectué n'affectent pas le message réel dans la file d'attente ou la rubrique de publication WebSphere MQ .

Put met à jour les propriétés MessageId et CorrelationId de l'objet MQMessage et n'efface pas les données de message. Les appels Put ou Get supplémentaires font référence aux informations mises à jour dans l'objet MQMessage . Par exemple, dans le fragment de code suivant, le premier message contient a et le second ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

message
Objet MQMessage contenant les données du descripteur de message et le message à envoyer. Le descripteur de message peut être modifié à la suite de cette méthode. Les valeurs du descripteur de message immédiatement après la fin de cette méthode sont les valeurs qui ont été placées dans la file d'attente ou publiées dans la rubrique.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- MQRC_CALL_INTERRUPTED si la connexion est interrompue lors de l'exécution d'un appel Put sur un message persistant et que la reconnexion a abouti.
- MQRC_NONE si la connexion aboutit lors de l'exécution d'un appel Put sur un message non persistant (voir Application Recovery).

putMessageOptions
Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de MQPutMessageOptons est utilisée.

Si vous utilisez l'option MQPMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

Remarque : Pour plus de simplicité et de performances, si vous souhaitez placer un seul message dans une file d'attente, utilisez l'objet MQQueueManager.Put . Vous devez disposer d'un objet MQQueue pour cela.

public void PutForwardMessage(MQMessage *message*);
public void PutForwardMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

Emissions MQException

Insérez un message en cours de réacheminement dans la file d'attente, où *message* est le message d'origine.

message
Objet MQMessage contenant les données du descripteur de message et le message à envoyer. Le descripteur de message peut être modifié à la suite de cette méthode. Les valeurs du descripteur de message immédiatement après la fin de cette méthode sont les valeurs qui ont été placées dans la file d'attente ou publiées dans la rubrique.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- MQRC_CALL_INTERRUPTED si la connexion est interrompue lors de l'exécution d'un appel Put sur un message persistant et que la reconnexion a abouti.
- MQRC_NONE si la connexion aboutit lors de l'exécution d'un appel Put sur un message non persistant (voir Application Recovery).

putMessageOptions
Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de MQPutMessageOptons est utilisée.

Si vous utilisez l'option MQPMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

public void PutReplyMessage(MQMessage *message*)
public void PutReplyMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*)

Emet MQException.

Placez un message de réponse dans la file d'attente, où *message* est le message d'origine.

message
Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée MessageId et CorrelationId sont définis comme requis.

Un client reconnectable renvoie le code anomalie MQRC_BACKED_OUT après une reconnexion réussie, pour les messages reçus sous MQGM_SYNCPOINT.

putMessageOptions
Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de MQPutMessageOptons est utilisée.

Si vous utilisez l'option MQPMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

public void PutReportMessage(MQMessage *message*)
public void PutReportMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*)

Emet MQException.

Placez un message de rapport dans la file d'attente, où *message* est le message d'origine.

message
Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée MessageId et CorrelationId sont définis comme requis.

Un client reconnectable renvoie le code anomalie MQRC_BACKED_OUT après une reconnexion réussie, pour les messages reçus sous MQGM_SYNCPOINT.

putMessageOptions
Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de MQPutMessageOptons est utilisée.

Si vous utilisez l'option MQPMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

Constructeurs

public MQQueue MQQueueManager.AccessQueue(string *queueName*, int *openOptions*);
public MQQueue MQQueueManager.AccessQueue(string *queueName*, int *openOptions*,
string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

Emet MQException.

Accède à une file d'attente sur ce gestionnaire de files d'attente.

Vous pouvez extraire ou parcourir des messages, insérer des messages, demander des informations sur les attributs de la file d'attente ou définir les attributs de la file d'attente. Si la file d'attente nommée est une file d'attente modèle, une file d'attente locale dynamique est créée. Interrogez l'attribut name de l'objet MQQueue résultant pour trouver le nom de la file d'attente dynamique.

queueName
Nom de la file d'attente à ouvrir.

openOptions
Options qui contrôlent l'ouverture de la file d'attente.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Valider avec l'ID utilisateur spécifié.

MQC.MQOO_BIND_AS_QDEF
Utilisez la liaison par défaut pour la file d'attente.

MQC.MQOO_BIND_NOT_FIXED
Ne vous liez pas à une destination spécifique.

MQC.MQOO_BIND_ON_OPEN
Descripteur de liaison à la destination lors de l'ouverture de la file d'attente.

MQC.MQOO_BROWSE
Ouvrir pour parcourir le message.

MQC.MQOO_FAIL_IF_QUIESCING
Echec si le gestionnaire de files d'attente est au repos.

MQC.MQOO_INPUT_AS_Q_DEF
Ouvrir pour obtenir les messages à l'aide de la valeur par défaut définie par la file d'attente.

MQC.MQOO_INPUT_SHARED
Ouvert pour obtenir les messages avec un accès partagé.

MQC.MQOO_INPUT_EXCLUSIVE
Ouvrir pour obtenir des messages avec un accès exclusif.

MQC.MQOO_INQUIRE
Ouvert pour interrogation-obligatoire si vous souhaitez interroger les propriétés.

MQC.MQOO_OUTPUT
Ouvrir pour insérer des messages.

MQC.MQOO_PASS_ALL_CONTEXT
Autoriser la transmission de tous les contextes.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Autoriser la transmission du contexte d'identité.

MQC.MQOO_SAVE_ALL_CONTEXT
Sauvegarder le contexte lors de l'extraction du message.

MQC.MQOO_SET
Ouvert pour définir des attributs-requis si vous souhaitez définir des propriétés.

MQC.MQOO_SET_ALL_CONTEXT
Permet de définir tous les contextes.

MQC.MQOO_SET_IDENTITY_CONTEXT
Permet de définir le contexte d'identité.

queueManagerName
Nom du gestionnaire de files d'attente sur lequel la file d'attente est définie. Un nom entièrement vide ou null indique le gestionnaire de files d'attente auquel l'objet `MQQueueManager` est connecté.

dynamicQueueName
dynamicQueueName est ignoré sauf si `queueName` spécifie le nom d'une file d'attente modèle. Si tel est le cas, *dynamicQueueName* indique le nom de la file d'attente dynamique à créer. Un nom vide ou nul n'est pas valide si `queueName` spécifie le nom d'une file d'attente modèle. Si le dernier caractère non blanc du nom est un astérisque, `*`, le gestionnaire de files d'attente remplace l'astérisque par une chaîne de caractères. Les caractères garantissent que le nom généré pour la file d'attente est unique sur ce gestionnaire de files d'attente.

alternateUserId
Si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` est spécifié dans le paramètre `openOptions` , *alternateUserId* indique l'autre identificateur utilisateur utilisé pour vérifier l'autorisation d'ouverture. Si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` n'est pas spécifié, *alternateUserId* peut être laissé vide ou null.

public MQQueue(MQQueueManager *queueManager*, string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

Emet `MQException`.

Accède à une file d'attente sur `queueManager`.

Vous pouvez extraire ou parcourir des messages, insérer des messages, demander des informations sur les attributs de la file d'attente ou définir les attributs de la file d'attente. Si la file d'attente nommée est une file d'attente modèle, une file d'attente locale dynamique est créée. Interrogez l'attribut `name` de l'objet `MQQueue` résultant pour trouver le nom de la file d'attente dynamique.

queueManager
Gestionnaire de files d'attente sur lequel accéder à la file d'attente.

queueName
Nom de la file d'attente à ouvrir.

openOptions
Options qui contrôlent l'ouverture de la file d'attente.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Valider avec l'ID utilisateur spécifié.

MQC.MQOO_BIND_AS_QDEF
Utilisez la liaison par défaut pour la file d'attente.

MQC.MQOO_BIND_NOT_FIXED
Ne vous liez pas à une destination spécifique.

MQC.MQOO_BIND_ON_OPEN
Descripteur de liaison à la destination lors de l'ouverture de la file d'attente.

MQC.MQOO_BROWSE
Ouvrir pour parcourir le message.

MQC.MQOO_FAIL_IF_QUIESCING
Echec si le gestionnaire de files d'attente est au repos.

MQC.MQOO_INPUT_AS_Q_DEF
Ouvrir pour obtenir les messages à l'aide de la valeur par défaut définie par la file d'attente.

MQC.MQOO_INPUT_SHARED
Ouvert pour obtenir les messages avec un accès partagé.

MQC.MQOO_INPUT_EXCLUSIVE
Ouvrir pour obtenir des messages avec un accès exclusif.

MQC.MQOO_INQUIRE
Ouvert pour interrogation-obligatoire si vous souhaitez interroger les propriétés.

MQC.MQOO_OUTPUT
Ouvrir pour insérer des messages.

MQC.MQOO_PASS_ALL_CONTEXT
Autoriser la transmission de tous les contextes.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Autoriser la transmission du contexte d'identité.

MQC.MQOO_SAVE_ALL_CONTEXT
Sauvegarder le contexte lors de l'extraction du message.

MQC.MQOO_SET
Ouvert pour définir des attributs-requis si vous souhaitez définir des propriétés.

MQC.MQOO_SET_ALL_CONTEXT
Permet de définir tous les contextes.

MQC.MQOO_SET_IDENTITY_CONTEXT
Permet de définir le contexte d'identité.

queueManagerName
Nom du gestionnaire de files d'attente sur lequel la file d'attente est définie. Un nom entièrement vide ou null indique le gestionnaire de files d'attente auquel l'objet `MQQueueManager` est connecté.

dynamicQueueName
dynamicQueueName est ignoré sauf si `queueName` spécifie le nom d'une file d'attente modèle. Si tel est le cas, *dynamicQueueName* indique le nom de la file d'attente dynamique à créer. Un nom vide ou nul n'est pas valide si `queueName` spécifie le nom d'une file d'attente modèle. Si le dernier caractère non blanc du nom est un astérisque, `*`, le gestionnaire de files d'attente remplace l'astérisque par une chaîne de caractères. Les caractères garantissent que le nom généré pour la file d'attente est unique sur ce gestionnaire de files d'attente.

alternateUserId
Si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` est spécifié dans le paramètre `openOptions` , *alternateUserId* indique l'autre identificateur utilisateur utilisé pour vérifier l'autorisation d'ouverture. Si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` n'est pas spécifié, *alternateUserId* peut être laissé vide ou null.

Classe `MQQueueManager` .NET

Utilisez `MQQueueManager` pour vous connecter à un gestionnaire de files d'attente et accéder aux objets de gestionnaire de files d'attente. Il contrôle également les transactions. Le constructeur `MQQueueManager` crée une connexion client ou serveur.

Classe

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.ManagedObject
                └ IBM.WMQ.MQQueueManager
```

```
public class IBM.WMQ.MQQueueManager extends IBM.WMQ.MQManagedObject;
```


- «Propriétés», à la page 1302
- «Des méthodes», à la page 1305
- «Constructeurs», à la page 1311

Propriétés

Test de l'émission de `MQException` lors de l'obtention des propriétés.

public int AccountingConnOverride {get;}
Indique si les applications peuvent remplacer les valeurs de comptabilité et de comptabilité de file d'attente `MQI` .

public int AccountingInterval {get;}
Durée avant l'écriture des enregistrements comptables intermédiaires (en secondes).

public int ActivityRecording {get;}
Ctrl la génération des rapports d'activité.

public int AdoptNewMCACheck {get;}
Indique quels éléments sont vérifiés pour déterminer si l'agent MCA est adopté lorsqu'un nouveau canal entrant est détecté. Pour être adopté, le nom de l'agent MCA doit correspondre à celui d'un agent MCA actif.

public int AdoptNewMCAInterval {get;}
Durée, en secondes, pendant laquelle le nouveau canal attend l'arrêt du canal orphelin.

public int AdoptNewMCAType {get;}
Indique si une instance MCA orpheline doit être adoptée (redémarrée) lorsqu'une nouvelle demande de canal entrant correspondant à la valeur MCACheck AdoptNewest détectée.

public int BridgeEvent {get;}
Indique si les événements de pont IMS sont générés.

public int ChannelEvent {get;}
Indique si des événements de canal sont générés.

public int ChannelInitiatorControl {get;}
Indique si l'initiateur de canal démarre automatiquement au démarrage du gestionnaire de files d'attente.

public int ChannelInitiatorAdapters {get;}
Nombre de sous-tâches de l'adaptateur devant traiter les appels WebSphere MQ .

public int ChannelInitiatorDispatchers {get;}
Nombre de répartiteurs à utiliser pour l'initiateur de canal.

public int ChannelInitiatorTraceAutoStart {get;}
Indique si la trace de l'initiateur de canal démarre automatiquement.

public int ChannelInitiatorTraceTableSize {get;}
Taille, en mégaoctets, de l'espace de données de trace d'un initiateur de canal .

public int ChannelMonitoring {get;}
Indique si la surveillance des canaux est utilisée.

public int ChannelStatistics {get;}
Ctrl la collecte des données de statistiques pour les canaux.

public int CharacterSet {get;}
Renvoie l'ID de jeu de caractères codés (CCSID) du gestionnaire de files d'attente. `CharacterSet` est utilisé par le gestionnaire de files d'attente pour toutes les zones de chaîne de caractères de l'interface de programme d'application.

public int ClusterSenderMonitoring {get;}
Contrôle la collecte des données de surveillance en ligne pour les canaux émetteurs de cluster définis automatiquement.

public int ClusterSenderStatistics {get;}
Contrôle la collecte des données statistiques pour les canaux émetteurs de cluster définis automatiquement.

public int ClusterWorkLoadMRU {get;}
Nombre maximal de canaux de cluster sortants.

public int ClusterWorkLoadUseQ {get;}
La valeur par défaut de la propriété `MQQueue` , `ClusterWorkLoadUseQ`, si elle spécifie la valeur `QMGR`.

public int CommandEvent {get;}
Définit si des événements Commande sont générés.

public string CommandInputQueueName {get;}
Renvoie le nom de la file d'attente d'entrée de commandes définie sur le gestionnaire de files d'attente. Les applications peuvent envoyer des commandes à cette file d'attente, si elles y sont autorisées.

public int CommandLevel {get;}
Indique le niveau de fonction du gestionnaire de files d'attente. L'ensemble des fonctions qui correspondent à un niveau de fonction particulier dépend de la plateforme. Sur une plateforme particulière, vous pouvez vous appuyer sur chaque gestionnaire de files d'attente prenant en charge les fonctions au niveau fonctionnel le plus bas commun à tous les gestionnaires de files d'attente.

public int CommandLevel {get;}
Indique si le serveur de commandes démarre automatiquement au démarrage du gestionnaire de files d'attente.

public string DNSGroup {get;}
Nom du groupe que le programme d'écoute TCP gérant les transmissions entrantes pour le groupe de partage de files d'attente doit joindre. Il rejoint ce groupe lors de l'utilisation de la prise en charge de Workload Manager for Dynamic Domain Name Services (DDNS).

public int DNSWLM {get;}
Indique si le programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente doit s'enregistrer auprès de Workload Manager for DDNS.

public int IPAddressVersion {get;}
Protocole IP (IPv4 ou IPv6) à utiliser pour une connexion de canal.

public boolean IsConnected {get;}
Renvoie la valeur de `isConnected`.

Si la valeur est true, une connexion au gestionnaire de files d'attente a été établie et n'est pas rompue. Tous les appels à IsConnected ne tentent pas activement d'atteindre le gestionnaire de files d'attente. Il est donc possible que la connectivité physique soit rompée, mais IsConnected peut toujours renvoyer la valeur true. L'état IsConnected est uniquement mis à jour lorsque l'activité, par exemple, l'insertion d'un message, l'obtention d'un message, est effectuée sur le gestionnaire de files d'attente.

Si la valeur est false, une connexion au gestionnaire de files d'attente n'a pas été établie, a été interrompue ou a été déconnectée.

public int KeepAlive {get;}
Indique si la fonction TCP KEEPALIVE doit être utilisée pour vérifier que l'autre extrémité de la connexion est toujours disponible. S'il n'est pas disponible, le canal est fermé.

public int ListenerTimer {get;}
Intervalle, en secondes, entre les tentatives de WebSphere MQ pour redémarrer le programme d'écoute après un échec APPC ou TCP/IP.

public int LoggerEvent {get;}
Indique si les événements du consignateur sont générés.

public string LU62ARMSuffix {get;}
Suffixe du membre APPCPM de SYS1.PARMLIB. Ce suffixe désigne la LUADD de cet initiateur de canal. Lorsque le gestionnaire de redémarrage automatique (ARM) redémarre l'initiateur de canal, la commande z/OS SET APPC=xx est émise.

public string LUGroupName {get; z/os}
Nom d'unité logique générique à utiliser par le programme d'écoute d'unité logique 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente.

public string LUName {get;}
Nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes.

public int MaximumActiveChannels {get;}
Le nombre maximal de canaux pouvant être actifs en même temps.

public int MaximumCurrentChannels {get;}
Nombre maximal de canaux pouvant être à jour à tout moment (y compris les canaux de connexion serveur avec des clients connectés).

public int MaximumLU62Channels {get;}
Nombre maximal de canaux pouvant être en cours, ou de clients pouvant être connectés, qui utilisent le protocole de transmission LU 6.2 .

public int MaximumMessageLength {get;}
Renvoie la longueur maximale d'un message (en octets) pouvant être traité par le gestionnaire de files d'attente. Aucune file d'attente ne peut être définie avec une longueur de message maximale supérieure à `MaximumMessageLongueur`.

public int MaximumPriority {get;}
Renvoie la priorité maximale des messages prise en charge par le gestionnaire de files d'attente. Les priorités sont comprises entre zéro (valeur la plus faible) et cette valeur. Emet `MQException` si vous appelez cette méthode après la déconnexion du gestionnaire de files d'attente.

public int MaximumTCPChannels {get;}
Nombre maximal de canaux pouvant être en cours, ou de clients pouvant être connectés, qui utilisent le protocole de transmission TCP/IP.

public int MQIAccounting {get;}
Ctrl la collecte des informations comptables pour les données MQI.

public int MQIStatistics {get;}
Ctrl la collecte d'informations de contrôle de statistiques pour le gestionnaire de files d'attente.

public int OutboundPortMax {get;}
Valeur maximale dans la plage de numéros de port à utiliser lors de la liaison de canaux sortants.

public int OutboundPortMin {get;}
Valeur minimale dans la plage de numéros de port à utiliser lors de la liaison de canaux sortants.

public int QueueAccounting {get;}
Indique si les données de comptabilité de classe 3 (comptabilité de niveau unité d'exécution et de niveau file d'attente) doivent être utilisées pour toutes les files d'attente.

public int QueueMonitoring {get;}
Ctrl la collecte des données de surveillance en ligne pour les files d'attente.

public int QueueStatistics {get;}
Ctrl la collecte des données de statistiques pour les files d'attente.

public int ReceiveTimeout {get;}
Durée pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire avant de revenir à l'état inactif.

public int ReceiveTimeoutMin {get;}
Durée minimale pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire avant de revenir à l'état inactif.

public int ReceiveTimeoutType {get;}
Qualificateur à appliquer à la valeur dans `ReceiveTimeout`.

public int SharedQueueQueueManagerName {get;}
Indique comment distribuer des messages dans une file d'attente partagée. Si l'insertion spécifie un gestionnaire de files d'attente différent du même groupe de partage de files d'attente que le gestionnaire de files d'attente cible, le message est distribué de deux manières:

MQC.MQSQQM_USE
Les messages sont distribués au gestionnaire de files d'attente d'objets avant d'être placés dans la file d'attente partagée.

MQCMQSQQM_IGNORE
Les messages sont placés directement dans la file d'attente partagée.

public int SSLEvent {get;}
Indique si des événements SSL sont générés.

public int SSLFips {get;}
Indique si seuls les algorithmes certifiés FIPS doivent être utilisés si la cryptographie est effectuée dans WebSphere MQ, plutôt que dans du matériel de cryptographie.

public int SSLKeyResetCount {get;}
Indique le nombre d'octets non chiffrés envoyés et reçus dans une conversation SSL avant la renégociation de la clé secrète.

public int ClusterSenderStatistics {get;}
Indique l'intervalle, en minutes, entre les collectes de statistiques consécutives.

public int SyncpointAvailability {get;}
Indique si le gestionnaire de files d'attente prend en charge les unités de travail et les points de synchronisation avec les méthodes `MQQueue.get` et `MQQueue.put` .

public string TCPName {get;}
Nom du système TCP/IP unique ou par défaut à utiliser, en fonction de la valeur de TCPStackType.

public int TCPStackType {get;}
Indique si l'initiateur de canal utilise uniquement l'espace adresse TCP/IP spécifié dans `TCPName`. L'initiateur de canal peut également se connecter à n'importe quelle adresse TCP/IP.

public int TraceRouteRecording {get;}
Contrôle l'enregistrement des informations de trace de route.

Des méthodes

public MQProcess AccessProcess(string *processName*, int *openOptions*);
public MQProcess AccessProcess(string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

Emet `MQException`.

Accédez à un processus WebSphere MQ sur ce gestionnaire de files d'attente pour consulter les attributs de processus.

processName
Nom du processus à ouvrir.

openOptions
Options qui contrôlent l'ouverture du processus. Les options valides qui peuvent être ajoutées ou combinées à l'aide d'un OR bit à bit sont les suivantes:

- `MQC.MQOO_FAIL_IF_QUIESCING`
- `MQC.MQOO_INQUIRE`

- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

queueManagerName
Nom du gestionnaire de files d'attente sur lequel le processus est défini. Vous pouvez laisser un nom de gestionnaire de files d'attente vide ou null si le gestionnaire de files d'attente est identique à celui auquel le processus accède.

alternateUserId

Si MQC.MQOO_ALTERNATE_USER_AUTHORITY est spécifié dans le paramètre *openOptions* , *alternateUserId* indique l'autre ID utilisateur utilisé pour vérifier l'autorisation de l'action. Si MQOO_ALTERNATE_USER_AUTHORITY n'est pas spécifié, *alternateUserId* peut être vide ou null.

Les droits utilisateur par défaut sont utilisés pour la connexion au gestionnaire de files d'attente si MQC.MQOO_ALTERNATE_USER_AUTHORITY n'est pas spécifié.

public MQQueue AccessQueue(string *queueName*, int *openOptions*);
public MQQueue AccessQueue(string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

Emet MQException.

Accède à une file d'attente sur ce gestionnaire de files d'attente.

Vous pouvez extraire ou parcourir des messages, insérer des messages, demander des informations sur les attributs de la file d'attente ou définir les attributs de la file d'attente. Si la file d'attente nommée est une file d'attente modèle, une file d'attente locale dynamique est créée. Interrogez l'attribut name de l'objet MQQueue résultant pour trouver le nom de la file d'attente dynamique.

queueName
Nom de la file d'attente à ouvrir.

openOptions
Options qui contrôlent l'ouverture de la file d'attente.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Valider avec l'ID utilisateur spécifié.

MQC.MQOO_BIND_AS_QDEF
Utilisez la liaison par défaut pour la file d'attente.

MQC.MQOO_BIND_NOT_FIXED
Ne vous liez pas à une destination spécifique.

MQC.MQOO_BIND_ON_OPEN
Descripteur de liaison à la destination lors de l'ouverture de la file d'attente.

MQC.MQOO_BROWSE
Ouvrir pour parcourir le message.

MQC.MQOO_FAIL_IF_QUIESCING
Echec si le gestionnaire de files d'attente est au repos.

MQC.MQOO_INPUT_AS_Q_DEF
Ouvrir pour obtenir les messages à l'aide de la valeur par défaut définie par la file d'attente.

MQC.MQOO_INPUT_SHARED
Ouvert pour obtenir les messages avec un accès partagé.

MQC.MQOO_INPUT_EXCLUSIVE
Ouvrir pour obtenir des messages avec un accès exclusif.

MQC.MQOO_INQUIRE
Ouvert pour interrogation-obligatoire si vous souhaitez interroger les propriétés.

MQC.MQOO_OUTPUT
Ouvrir pour insérer des messages.

MQC.MQOO_PASS_ALL_CONTEXT
Autoriser la transmission de tous les contextes.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Autoriser la transmission du contexte d'identité.

MQC.MQOO_SAVE_ALL_CONTEXT
Sauvegarder le contexte lors de l'extraction du message.

MQC.MQOO_SET
Ouvert pour définir des attributs-requis si vous souhaitez définir des propriétés.

MQC.MQOO_SET_ALL_CONTEXT
Permet de définir tous les contextes.

MQC.MQOO_SET_IDENTITY_CONTEXT
Permet de définir le contexte d'identité.

queueManagerName
Nom du gestionnaire de files d'attente sur lequel la file d'attente est définie. Un nom entièrement vide ou null indique le gestionnaire de files d'attente auquel l'objet `MQQueueManager` est connecté.

dynamicQueueName
dynamicQueueName est ignoré sauf si `queueName` spécifie le nom d'une file d'attente modèle. Si tel est le cas, *dynamicQueueName* indique le nom de la file d'attente dynamique à créer. Un nom vide ou nul n'est pas valide si `queueName` spécifie le nom d'une file d'attente modèle. Si le dernier caractère non blanc du nom est un astérisque, `*`, le gestionnaire de files d'attente remplace l'astérisque par une chaîne de caractères. Les caractères garantissent que le nom généré pour la file d'attente est unique sur ce gestionnaire de files d'attente.

alternateUserId
Si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` est spécifié dans le paramètre `openOptions` , *alternateUserId* indique l'autre identificateur utilisateur utilisé pour vérifier l'autorisation d'ouverture. Si `MQC.MQOO_ALTERNATE_USER_AUTHORITY` n'est pas spécifié, *alternateUserId* peut être laissé vide ou null.

```
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);
```

Accédez à une rubrique de ce gestionnaire de files d'attente.

Les objets `MQTopic` sont étroitement liés aux objets de rubrique d'administration, qui sont parfois appelés objets de rubrique. En entrée, `topicObject` pointe vers un objet de rubrique d'administration. Le constructeur `MQTopic` obtient une chaîne de rubrique de l'objet de rubrique et la combine avec `topicName` pour créer un nom de rubrique. `topicObject` ou `topicName` peut être null. Le nom de rubrique est mis en correspondance avec l'arborescence de rubriques et le nom de l'objet de rubrique d'administration correspondant le plus proche est renvoyé dans `topicObject`.

Les rubriques associées à l'objet MQTopic sont le résultat de la combinaison de deux chaînes de rubrique. La première chaîne de rubrique est définie par l'objet de rubrique d'administration identifié par *topicObject*. La deuxième chaîne de rubrique est *topicString*. La chaîne de rubrique résultante associée à l'objet MQTopic peut identifier plusieurs rubriques en incluant des caractères génériques.

Selon que la rubrique est ouverte pour la publication ou l'abonnement, vous pouvez utiliser les méthodes MQTopic.Put pour la publication sur des rubriques ou les méthodes MQTopic.Get pour la réception de publications sur des rubriques. Si vous souhaitez publier et vous abonner à la même rubrique, vous devez accéder à la rubrique deux fois, une fois pour la publication et une fois pour l'abonnement.

Si vous créez un objet MQTopic pour l'abonnement, sans fournir d'objet MQDestination , un abonnement géré est supposé. Si vous transmettez une file d'attente en tant qu'objet MQDestination , un abonnement non géré est supposé. Vous devez vous assurer que les options d'abonnement que vous définissez sont cohérentes avec l'abonnement géré ou non géré.

destination
destination est une instance MQQueue . En fournissant *destination*, MQTopic est ouvert en tant qu'abonnement non géré. Les publications de la rubrique sont distribuées dans la file d'attente accessible en tant que *destination*.

topicName
Chaîne de rubrique correspondant à la deuxième partie du nom de la rubrique. *topicName* est concaténé avec la chaîne de rubrique définie dans l'objet de rubrique d'administration *topicObject* . Vous pouvez définir *topicName* sur null, auquel cas le nom de rubrique est défini par la chaîne de rubrique dans *topicObject*.

topicObject
En entrée, *topicObject* est le nom de l'objet de rubrique qui contient la chaîne de rubrique qui forme la première partie du nom de rubrique. La chaîne de rubrique dans *topicObject* est concaténée avec *topicName*. Les règles de construction des noms de rubrique sont définies dans Combinaison de chaînes de rubrique.

Dans la sortie, *topicObject* contient le nom de l'objet de rubrique d'administration qui est la correspondance la plus proche dans l'arborescence de rubriques avec la rubrique identifiée par le nom de la rubrique.

openAs
Accédez à la rubrique pour la publication ou l'abonnement. Le paramètre ne peut contenir qu'une seule des options suivantes:

- MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION
- MQC.MQTOPIC_OPEN_AS_PUBLICATION

options

Combinez les options qui contrôlent l'ouverture de la rubrique pour la publication ou l'abonnement. Utilisez les constantes MQC.MQSO_* pour accéder à une rubrique pour l'abonnement et les constantes MQC.MQOO_* pour accéder à une rubrique pour la publication.

Si plusieurs options sont requises, ajoutez les valeurs ensemble ou combinez les valeurs d'option à l'aide de l'opérateur OR bit à bit.

alternateUserId
Indiquez l'ID utilisateur de remplacement utilisé pour vérifier l'autorisation requise pour terminer l'opération. Vous devez spécifier *alternateUserId*, si MQC.MQOO_ALTERNATE_USER_AUTHORITY ou MQC.MQSO_ALTERNATE_USER_AUTHORITY est défini dans le paramètre d'options.

subscriptionName
subscriptionName est obligatoire si les options MQC.MQSO_DURABLE ou MQC.MQSO_ALTER sont fournies. Dans les deux cas, MQTopic est implicitement ouvert pour l'abonnement. Une exception est émise si MQC.MQSO_DURABLE est défini et que l'abonnement existe, ou si MQC.MQSO_ALTER est défini, et que l'abonnement n'existe pas.

properties
Définissez les propriétés d'abonnement spéciales répertoriées à l'aide d'une table de hachage. Les entrées spécifiées dans la table de hachage sont mises à jour avec les valeurs de sortie. Les entrées ne sont pas ajoutées à la table de hachage pour signaler les valeurs de sortie.

- 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();

Emissions MQException

Renvoie un objet MQAsyncStatus qui représente l'activité asynchrone pour la connexion du gestionnaire de files d'attente.

public void Backout();

Emet MQException.

Annuler les messages qui ont été lus ou écrits dans le point de synchronisation depuis le dernier point de synchronisation.

Les messages qui ont été écrits avec l'indicateur MQC.MQPMO_SYNCPOINT défini sont supprimés des files d'attente. Les messages lus avec l'indicateur MQC.MQGMO_SYNCPOINT sont réintégrés dans les files d'attente dont ils proviennent. Si les messages sont persistants, les modifications sont consignées.

Pour les clients reconnectables, le code anomalie MQRC_NONE est renvoyé à un client une fois que la reconnexion a abouti.

public void Begin();

Emet MQException.

Begin est pris en charge uniquement en mode liaisons de serveur. Il démarre une unité de travail globale.

public void Commit();

Emet MQException.

Validez tous les messages qui ont été lus ou écrits dans le point de synchronisation depuis le dernier point de synchronisation.

Les messages écrits avec l'indicateur MQC.MQPMO_SYNCPOINT défini sont mis à la disposition d'autres applications. Les messages extraits avec l'indicateur MQC.MQGMO_SYNCPOINT sont supprimés. Si les messages sont persistants, les modifications sont consignées.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- MQRC_CALL_INTERRUPTED si la connexion est perdue lors de l'exécution de l'appel de validation.
- MQRC_BACKED_OUT si l'appel de validation est émis après la reconnexion.

Disconnect();

Emet MQException.

Fermez la connexion au gestionnaire de files d'attente. Tous les objets accessibles sur ce gestionnaire de files d'attente ne sont plus accessibles à cette application. Pour accéder à nouveau aux objets, créez un objet MQQueueManager .

En règle générale, tout travail effectué dans le cadre d'une unité de travail est validé. Toutefois, si l'unité de travail est gérée par .NET, l'unité de travail peut être annulée.

```
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);
```

Emet `MQException`.

Place un message unique dans une file d'attente ou une rubrique sans créer d'objet `MQQueue` ou `MQTopic` au préalable.

queueName
Nom de la file d'attente dans laquelle placer le message.

destinationName
Nom d'un objet de destination. Il s'agit d'une file d'attente ou d'une rubrique en fonction de la valeur de *type*.

type
Type de l'objet de destination. Vous ne devez pas combiner les options.

MQC.MQOT_Q
File d'attente

MQC.MQOT_TOPIC
Topic

queueManagerName

Nom du gestionnaire de files d'attente ou de l'alias du gestionnaire de files d'attente sur lequel la file d'attente est définie. Si le type `MQC.MQOT_TOPIC` est spécifié, ce paramètre est ignoré.

Si la file d'attente est une file d'attente modèle et que le nom du gestionnaire de files d'attente résolu n'est pas ce gestionnaire de files d'attente, une exception `MQException` est émise.

topicString

topicString est combiné avec le nom de la rubrique dans l'objet de rubrique *destinationName* .

topicString est ignoré si *destinationName* est une file d'attente.

message
Message à envoyer. Le message est un objet d'entrée/sortie.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- `MQRC_CALL_INTERRUPTED` si la connexion est interrompue lors de l'exécution d'un appel Put sur un message persistant.
- `MQRC_NONE` si la connexion aboutit lors de l'exécution d'un appel Put sur un message non persistant (voir Application Recovery).

putMessageOptions

Options contrôlant les actions de l'insertion.

Si vous omettez *putMessageOptions*, une instance par défaut de *putMessageOptions* est créée. *putMessageOptions* est un objet d'entrée-sortie.

Si vous utilisez l'option MQPMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

alternateUserId

Indique un autre ID utilisateur utilisé pour vérifier l'autorisation lors du placement du message dans une file d'attente.

Vous pouvez omettre *alternateUserId* si vous ne définissez pas MQC.MQOO_ALTERNATE_USER_AUTHORITY dans *putMessageOptions*. Si vous définissez MQC.MQOO_ALTERNATE_USER_AUTHORITY, vous devez également définir *alternateUserId*. *alternateUserId* n'a pas d'effet sauf si vous avez également défini MQC.MQOO_ALTERNATE_USER_AUTHORITY.

Constructeurs

```
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);
```

Emet MQException.

Crée une connexion à un gestionnaire de files d'attente. Sélectionnez entre la création d'une connexion client ou d'une connexion serveur.

Vous devez disposer du droit d'interrogation (inq) sur le gestionnaire de files d'attente lorsque vous tentez de vous connecter au gestionnaire de files d'attente. Sans droit d'interrogation, la tentative de connexion échoue.

Une connexion client est créée si l'une des conditions suivantes est vérifiée:

1. *channel* ou *connName* sont spécifiés dans le constructeur.
2. *HostName*, *Port*ou *Channel* sont spécifiés dans *properties*.
3. *MQEnvironment.HostName*, *MQEnvironment.Port*ou *MQEnvironment.Channel* sont spécifiés.

Les valeurs des propriétés de connexion sont définies par défaut dans l'ordre indiqué. Les valeurs *channel* et *connName* du constructeur sont prioritaires sur les valeurs de propriété du constructeur. Les valeurs de propriété du constructeur sont prioritaires sur les propriétés MQEnvironment .

Le nom d'hôte, le nom de canal et le port sont définis dans la classe MQEnvironment .

queueManagerName
Nom du gestionnaire de files d'attente ou du groupe de gestionnaires de files d'attente auquel se connecter.

Omettez le paramètre, ou laissez-le null ou vide pour effectuer une sélection de gestionnaire de files d'attente par défaut. La connexion de gestionnaire de files d'attente par défaut sur un serveur correspond au gestionnaire de files d'attente par défaut sur le serveur. La connexion de gestionnaire de files d'attente par défaut sur une connexion client est établie avec le gestionnaire de files d'attente auquel le programme d'écoute est connecté.

options
Spécifiez les options de connexion MQCNO . Les valeurs doivent être applicables au type de connexion en cours d'établissement. Par exemple, si vous spécifiez les propriétés de connexion serveur suivantes pour une connexion client, une exception MQException est émise.

- MQC.MQCNO_FASTPATH_BINDING
- MQC.MQCNO_STANDARD_BINDING

properties

Le paramètre properties prend une série de paires clé / valeur qui remplacent les propriétés définies par MQEnvironment; voir l'exemple «Remplacer les propriétés MQEnvironment», à la page 1314. Les propriétés suivantes peuvent être remplacées:

- 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
Nom d'un canal de connexion serveur

connName
Nom de connexion au format *HostName* (*Port*).

Vous pouvez fournir une liste de *noms d'hôte* et de *ports* en tant qu'argument au constructeur MQQueueManager(String queueManagerName, Hashtable properties) à l'aide de CONNECTION_NAME_PROPERTY.

Exemple :

```
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);
```

Lorsqu'une tentative de connexion est effectuée, la liste des noms de connexion est traitée dans l'ordre. Si la tentative de connexion au premier nom d'hôte et au premier port échoue, la connexion à la deuxième paire d'attributs est tentée. Le client répète ce processus jusqu'à ce qu'une connexion soit établie ou que la liste soit épuisée. Si la liste est épuisée, un code anomalie et un code achèvement appropriés sont renvoyés à l'application client.

Lorsqu'aucun numéro de port n'est fourni pour le nom de connexion, le port par défaut (configuré dans `mqclient.ini`) est utilisé.

Définition de la liste de connexions

Vous pouvez définir la liste de connexion à l'aide des méthodes suivantes lorsque les options de reconnexion automatique du client sont définies:

Définition de la liste de connexions via MQSERVER

Vous pouvez définir la liste de connexion à l'aide de l'invite de commande.

Dans l'invite de commande, définissez

```
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)
```

Si vous définissez la connexion dans MQSERVER, ne la définissez pas dans l'application.

Si vous définissez la liste de connexions dans l'application, l'application écrase tout ce qui est défini dans la variable d'environnement MQSERVER.

Définir la liste de connexion via l'application

Vous pouvez définir la liste de connexion dans l'application en spécifiant le nom d'hôte et les propriétés de port.

```
String connName = "fred.mq.com(2344), nick.mq.com(3746), chris.mq.com(4288)";
MQQueueManager qm = new MQQueueManager("QM1", "TestChannel", connName);
```

Définition de la liste de connexion via `app.config`

`App.config` est un fichier XML dans lequel vous spécifiez les paires clé-valeur.

Dans la liste de connexion, indiquez

```
<app.Settings>
<add key="Connection1" value="Hostname1(Port1)"/>
<add key="Connection2" value="Hostname2(Port2)"/>
<app.Settings>
```

Exemple :

```
<app.Settings>
<add key>="Connection1" value="fred.mq.com(2966)"/>
<add key>="Connection2" value="alex.mq.com(6533)"/>
<app.Settings>
```

Vous pouvez modifier directement la liste de connexion dans le fichier `app.config` .

Définition de la liste de connexion via `MQEnvironment`

Pour définir la liste de connexions via `MQEnvironment`, utilisez la propriété *ConnectionName* .

```
MQEnvironment.ConnectionName = "fred.mq.com(4288),"alex.mq.com(5211);
```

La propriété *ConnectionName* remplace les propriétés de nom d'hôte et de port définies dans `MQEnvironment`.

Créer une connexion client

L'exemple suivant montre comment créer une connexion client à un gestionnaire de files d'attente. Vous pouvez créer une connexion client en définissant les variables `MQEnvironment` avant de créer un objet `MQQueueManager` .

```
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");
```

Figure 43. Connexion client

Remplacer les propriétés `MQEnvironment`

L'exemple suivant montre comment créer un gestionnaire de files d'attente avec son ID utilisateur et son mot de passe définis dans une table de hachage.

```
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);
}
```

Figure 44. Remplacement des propriétés `MQEnvironment`

Créer une connexion reconnectable

L'exemple suivant montre comment reconnecter automatiquement un client à un gestionnaire de files d'attente.

```
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);
```

Figure 45. Reconnexion automatique d'un client à un gestionnaire de files d'attente

Classe `MQSubscription` .NET

Utilisez `MQSubscription` pour demander que les publications conservées soient envoyées à l'abonné. `MQSubscription` est une propriété d'un objet `MQTopic` ouvert pour abonnement.

Classe

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQManagedObject
                └ IBM.WMQ.MQSubscription
```

```
public class IBM.WMQ.MQSubscription extends IBM.WMQ.MQManagedObject;
```


- «Propriétés», à la page 1315
- «Des méthodes», à la page 1315
- «Constructeurs», à la page 1315

Propriétés

Accédez aux propriétés d'abonnement à l'aide de la classe MQManagedObject ; voir «Propriétés», à la page 1273.

Des méthodes

Accédez aux méthodes d'abonnement Inquire, Set et Get à l'aide de la classe MQManagedObject ; voir «Des méthodes», à la page 1274.

public int RequestPublicationUpdate(int *options*);

Emet MQException.

Demander une publication mise à jour pour la rubrique en cours. Si le gestionnaire de files d'attente possède des publications conservées pour la rubrique, elles sont envoyées à l'abonné.

Avant d'appeler RequestPublicationUpdate, ouvrez une rubrique pour l'abonnement afin d'obtenir un objet MQSubscription .

En règle générale, ouvrez l'abonnement avec l'option MQC.MQSO_PUBLICATIONS_ON_REQUEST . Si aucun caractère générique n'est présent dans la chaîne de rubrique, une seule publication est envoyée suite à cet appel. Si la chaîne de rubrique contient des caractères génériques, de nombreuses publications peuvent être envoyées. La méthode renvoie le nombre de publications conservées qui sont envoyées à la file d'attente d'abonnement. Il n'est pas garanti que ces nombreuses publications soient reçues, surtout s'il s'agit de messages non persistants.

options

MQC.MQSRO_FAIL_IF_QUIESCING
La méthode échoue si le gestionnaire de files d'attente est à l'état de repos. Sous z/OS, pour une application CICS ou IMS , MQC.MQSRO_FAIL_IF_QUIESCING force également la méthode à échouer si la connexion est à l'état de repos.

MQC.MQSRO_NONE
Aucune option n'est spécifiée.

Constructeurs

Aucun constructeur public .

Un objet MQSubscription est renvoyé dans la propriété SubscriptionReference d'un objet MQTopic ouvert pour l'abonnement,

Appelez la méthode RequestPublicationUpdate . MQSubscription est une sous-classe de MQManagedObject. Utilisez la référence pour accéder aux propriétés et aux méthodes de MQManagedObject.

Classe MQTopic .NET

Utilisez MQTopic pour publier ou abonner des messages sur une rubrique, ou pour interroger ou définir des attributs d'une rubrique. Créez un objet MQTopic pour la publication ou l'abonnement à l'aide d'un constructeur ou de la méthode MQQueueManager.AccessTopic .

Classe

```
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;
```


- «Propriétés», à la page 1316
- «Des méthodes», à la page 1316
- «Constructeurs», à la page 1318

Propriétés

Test de l'émission de MQException lors de l'obtention des propriétés.

public Boolean IsDurable {get;}
Propriété en lecture seule qui renvoie True si l'abonnement est durable ou False dans le cas contraire. Si la rubrique a été ouverte pour publication, la propriété est ignorée et renvoie toujours False.

public Boolean IsManaged {get;};
Propriété en lecture seule qui renvoie True si l'abonnement est géré par le gestionnaire de files d'attente ou False dans le cas contraire. Si la rubrique a été ouverte pour publication, la propriété est ignorée et renvoie toujours False.

public Boolean IsSubscribed {get;};
Propriété en lecture seule qui renvoie True si la rubrique a été ouverte pour abonnement et False si la rubrique a été ouverte pour publication.

public MQSubscription SubscriptionReference {get;};
Propriété en lecture seule qui renvoie l'objet MQSubscription associé à un objet de rubrique ouvert pour abonnement. La référence est disponible si vous souhaitez modifier les options de fermeture ou démarrer l'une des méthodes d'objet.

public MQDestination UnmanagedDestinationReference {get;};
Propriété en lecture seule qui renvoie le MQQueue associé à un abonnement non géré. Il s'agit de la destination spécifiée lors de la création de l'objet de rubrique. La propriété renvoie la valeur null pour tous les objets de rubrique ouverts pour publication ou avec un abonnement géré.

Des méthodes

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);
Emet une exception MQException.

Publie un message dans le sujet.

Les modifications apportées à l'objet MQMessage une fois l'appel Put effectué n'affectent pas le message réel dans la file d'attente ou la rubrique de publication WebSphere MQ .

Put met à jour les propriétés MessageId et CorrelationId de l'objet MQMessage et n'efface pas les données de message. Les appels Put ou Get supplémentaires font référence aux informations mises à jour dans l'objet MQMessage . Par exemple, dans le fragment de code suivant, le premier message contient a et le second ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

message
Objet MQMessage contenant les données du descripteur de message et le message à envoyer. Le descripteur de message peut être modifié à la suite de cette méthode. Les valeurs du descripteur de message immédiatement après la fin de cette méthode sont les valeurs qui ont été placées dans la file d'attente ou publiées dans la rubrique.

Les codes anomalie suivants sont renvoyés à un client reconnectable:

- MQRC_CALL_INTERRUPTED si la connexion est interrompue lors de l'exécution d'un appel Put sur un message persistant et que la reconnexion a abouti.
- MQRC_NONE si la connexion aboutit lors de l'exécution d'un appel Put sur un message non persistant (voir Application Recovery).

putMessageOptions
Options contrôlant l'action de l'insertion.

Si *putMessageOptions* n'est pas spécifié, l'instance par défaut de MQPutMessageOptons est utilisée.

Si vous utilisez l'option MQPMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

Remarque : Pour plus de simplicité et de performances, si vous souhaitez placer un seul message dans une file d'attente, utilisez l'objet MQQueueManager.Put . Vous devez disposer d'un objet MQQueue pour cela.

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);
Emet une exception MQException.

Extrait un message de la rubrique.

Cette méthode utilise une instance par défaut de MQGetMessageOptions pour effectuer l'extraction. L'option de message utilisée est MQGMO_NOWAIT.

Si l'extraction échoue, l'objet MQMessage n'est pas modifié. Si elle aboutit, le descripteur de message et les parties de données de message du MQMessage sont remplacés par le descripteur de message et les données de message du message entrant.

Tous les appels à WebSphere MQ à partir d'un MQQueueManager particulier sont synchrones. Par conséquent, si vous effectuez une opération d'extraction avec attente, toutes les autres unités d'exécution utilisant le même MQQueueManager sont bloquées pour effectuer d'autres appels WebSphere MQ jusqu'à ce que l'appel Get soit effectué. Si vous avez besoin de plusieurs unités d'exécution pour accéder simultanément à WebSphere MQ , chaque unité d'exécution doit créer son propre objet MQQueueManager .

message
Contient le descripteur de message et les données de message renvoyées. Certaines des zones du descripteur de message sont des paramètres d'entrée. Il est important de s'assurer que les paramètres d'entrée MessageId et CorrelationId sont définis comme requis.

Un client reconnectable renvoie le code anomalie MQRC_BACKED_OUT après une reconnexion réussie, pour les messages reçus sous MQGM_SYNCPOINT.

getMessageOptions
Options contrôlant l'action de la commande get.

L'utilisation de l'option MQC.MQGMO_CONVERT peut générer une exception avec le code anomalie MQC.MQRC_CONVERTED_STRING_TOO_BIG lors de la conversion de codes de caractères codés sur un octet en codes codés sur deux octets. Dans ce cas, le message est copié dans la mémoire tampon sans conversion.

Si *getMessageOptions* n'est pas spécifié, l'option de message utilisée est MQGMO_NOWAIT.

Si vous utilisez l'option MQGMO_LOGICAL_ORDER dans un client reconnectable, le code anomalie MQRC_RECONNECT_INCOMPATIBLE est renvoyé.

MaxMsgSize
Message le plus volumineux que cet objet message doit recevoir. Si le message de la file d'attente est supérieur à cette taille, l'un des deux événements suivants se produit:

- Si l'indicateur MQGMO_ACCEPT_TRUNCATED_MSG est défini dans l'objet MQGetMessageOptions , le message est rempli avec autant de données de message que possible. Une exception est émise avec le code achèvement MQCC_WARNING et le code anomalie MQRC_TRUNCATED_MSG_ACCEPTED .
- Si l'indicateur MQGMO_ACCEPT_TRUNCATED_MSG n'est pas défini, le message reste dans la file d'attente. Une exception est émise avec le code achèvement MQCC_WARNING et le code anomalie MQRC_TRUNCATED_MSG_FAILED .

Si *MaxMsgSize* n'est pas spécifié, l'intégralité du message est extraite.

Constructeurs

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*);

Accédez à une rubrique sur *queueManager*.

Les objets MQTopic sont étroitement liés aux objets de rubrique d'administration, qui sont parfois appelés objets de rubrique. En entrée, topicObject pointe vers un objet de rubrique d'administration. Le constructeur MQTopic obtient une chaîne de rubrique de l'objet de rubrique et la combine avec topicName pour créer un nom de rubrique. topicObject ou topicName peut être null. Le nom de rubrique est mis en correspondance avec l'arborescence de rubriques et le nom de l'objet de rubrique d'administration correspondant le plus proche est renvoyé dans topicObject.

Les rubriques associées à l'objet MQTopic sont le résultat de la combinaison de deux chaînes de rubrique. La première chaîne de rubrique est définie par l'objet de rubrique d'administration identifié par *topicObject*. La deuxième chaîne de rubrique est *topicString*. La chaîne de rubrique

résultante associée à l'objet MQTopic peut identifier plusieurs rubriques en incluant des caractères génériques.

Selon que la rubrique est ouverte pour la publication ou l'abonnement, vous pouvez utiliser les méthodes MQTopic.Put pour la publication sur des rubriques ou les méthodes MQTopic.Get pour la réception de publications sur des rubriques. Si vous souhaitez publier et vous abonner à la même rubrique, vous devez accéder à la rubrique deux fois, une fois pour la publication et une fois pour l'abonnement.

Si vous créez un objet MQTopic pour l'abonnement, sans fournir d'objet MQDestination , un abonnement géré est supposé. Si vous transmettez une file d'attente en tant qu'objet MQDestination , un abonnement non géré est supposé. Vous devez vous assurer que les options d'abonnement que vous définissez sont cohérentes avec l'abonnement géré ou non géré.

queueManager
Gestionnaire de files d'attente sur lequel accéder à une rubrique.

destination
destination est une instance MQQueue . En fournissant *destination*, MQTopic est ouvert en tant qu'abonnement non géré. Les publications de la rubrique sont distribuées dans la file d'attente accessible en tant que *destination*.

topicName
Chaîne de rubrique correspondant à la deuxième partie du nom de la rubrique. *topicName* est concaténé avec la chaîne de rubrique définie dans l'objet de rubrique d'administration *topicObject* . Vous pouvez définir *topicName* sur null, auquel cas le nom de rubrique est défini par la chaîne de rubrique dans *topicObject*.

topicObject
En entrée, *topicObject* est le nom de l'objet de rubrique qui contient la chaîne de rubrique qui forme la première partie du nom de rubrique. La chaîne de rubrique dans *topicObject* est concaténée avec *topicName*. Les règles de construction des noms de rubrique sont définies dans Combinaison de chaînes de rubrique.

Dans la sortie, *topicObject* contient le nom de l'objet de rubrique d'administration qui est la correspondance la plus proche dans l'arborescence de rubriques avec la rubrique identifiée par le nom de la rubrique.

openAs
Accédez à la rubrique pour la publication ou l'abonnement. Le paramètre ne peut contenir qu'une seule des options suivantes:

- MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION
- MQC.MQTOPIC_OPEN_AS_PUBLICATION

options

Combinez les options qui contrôlent l'ouverture de la rubrique pour la publication ou l'abonnement. Utilisez les constantes MQC.MQSO_* pour accéder à une rubrique pour l'abonnement et les constantes MQC.MQOO_* pour accéder à une rubrique pour la publication.

Si plusieurs options sont requises, ajoutez les valeurs ensemble ou combinez les valeurs d'option à l'aide de l'opérateur OR bit à bit.

alternateUserId
Indiquez l'ID utilisateur de remplacement utilisé pour vérifier l'autorisation requise pour terminer l'opération. Vous devez spécifier *alternateUserId*, si MQC.MQOO_ALTERNATE_USER_AUTHORITY ou MQC.MQSO_ALTERNATE_USER_AUTHORITY est défini dans le paramètre d'options.

subscriptionName
subscriptionName est obligatoire si les options MQC.MQSO_DURABLE ou MQC.MQSO_ALTER sont fournies. Dans les deux cas, MQTopic est implicitement ouvert pour l'abonnement. Une exception est émise si MQC.MQSO_DURABLE est défini et que l'abonnement existe, ou si MQC.MQSO_ALTER est défini, et que l'abonnement n'existe pas.

properties
Définissez les propriétés d'abonnement spéciales répertoriées à l'aide d'une table de hachage. Les entrées spécifiées dans la table de hachage sont mises à jour avec les valeurs de sortie. Les entrées ne sont pas ajoutées à la table de hachage pour signaler les valeurs de sortie.

- 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*);

Accédez à une rubrique de ce gestionnaire de files d'attente.

Les objets MQTopic sont étroitement liés aux objets de rubrique d'administration, qui sont parfois appelés objets de rubrique. En entrée, topicObject pointe vers un objet de rubrique d'administration. Le constructeur MQTopic obtient une chaîne de rubrique de l'objet de rubrique et la combine avec topicName pour créer un nom de rubrique. topicObject ou topicName peut être null. Le nom de rubrique est mis en correspondance avec l'arborescence de rubriques et le nom de l'objet de rubrique d'administration correspondant le plus proche est renvoyé dans topicObject.

Les rubriques associées à l'objet MQTopic sont le résultat de la combinaison de deux chaînes de rubrique. La première chaîne de rubrique est définie par l'objet de rubrique d'administration identifié par *topicObject*. La deuxième chaîne de rubrique est *topicString*. La chaîne de rubrique résultante associée à l'objet MQTopic peut identifier plusieurs rubriques en incluant des caractères génériques.

Selon que la rubrique est ouverte pour la publication ou l'abonnement, vous pouvez utiliser les méthodes MQTopic.Put pour la publication sur des rubriques ou les méthodes MQTopic.Get pour la réception de publications sur des rubriques. Si vous souhaitez publier et vous abonner à la même rubrique, vous devez accéder à la rubrique deux fois, une fois pour la publication et une fois pour l'abonnement.

Si vous créez un objet MQTopic pour l'abonnement, sans fournir d'objet MQDestination , un abonnement géré est supposé. Si vous transmettez une file d'attente en tant qu'objet MQDestination , un abonnement non géré est supposé. Vous devez vous assurer que les options d'abonnement que vous définissez sont cohérentes avec l'abonnement géré ou non géré.

destination
destination est une instance MQQueue . En fournissant *destination*, MQTopic est ouvert en tant qu'abonnement non géré. Les publications de la rubrique sont distribuées dans la file d'attente accessible en tant que *destination*.

topicName
Chaîne de rubrique correspondant à la deuxième partie du nom de la rubrique. *topicName* est concaténé avec la chaîne de rubrique définie dans l'objet de rubrique d'administration *topicObject* . Vous pouvez définir *topicName* sur null, auquel cas le nom de rubrique est défini par la chaîne de rubrique dans *topicObject*.

topicObject
En entrée, *topicObject* est le nom de l'objet de rubrique qui contient la chaîne de rubrique qui forme la première partie du nom de rubrique. La chaîne de rubrique dans *topicObject* est concaténée avec *topicName*. Les règles de construction des noms de rubrique sont définies dans Combinaison de chaînes de rubrique.

Dans la sortie, *topicObject* contient le nom de l'objet de rubrique d'administration qui est la correspondance la plus proche dans l'arborescence de rubriques avec la rubrique identifiée par le nom de la rubrique.

openAs
Accédez à la rubrique pour la publication ou l'abonnement. Le paramètre ne peut contenir qu'une seule des options suivantes:

- MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION
- MQC.MQTOPIC_OPEN_AS_PUBLICATION

options

Combinez les options qui contrôlent l'ouverture de la rubrique pour la publication ou l'abonnement. Utilisez les constantes MQC.MQSO_* pour accéder à une rubrique pour l'abonnement et les constantes MQC.MQOO_* pour accéder à une rubrique pour la publication.

Si plusieurs options sont requises, ajoutez les valeurs ensemble ou combinez les valeurs d'option à l'aide de l'opérateur OR bit à bit.

alternateUserId
Indiquez l'ID utilisateur de remplacement utilisé pour vérifier l'autorisation requise pour terminer l'opération. Vous devez spécifier *alternateUserId*, si MQC.MQOO_ALTERNATE_USER_AUTHORITY ou MQC.MQSO_ALTERNATE_USER_AUTHORITY est défini dans le paramètre d'options.

subscriptionName
subscriptionName est obligatoire si les options MQC.MQSO_DURABLE ou MQC.MQSO_ALTER sont fournies. Dans les deux cas, MQTopic est implicitement ouvert pour l'abonnement. Une exception est émise si MQC.MQSO_DURABLE est défini et que l'abonnement existe, ou si MQC.MQSO_ALTER est défini, et que l'abonnement n'existe pas.

properties
Définissez les propriétés d'abonnement spéciales répertoriées à l'aide d'une table de hachage. Les entrées spécifiées dans la table de hachage sont mises à jour avec les valeurs de sortie. Les entrées ne sont pas ajoutées à la table de hachage pour signaler les valeurs de sortie.

- 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

Interface IMQObjectTrigger .NET

Implémentez IMQObjectTrigger pour traiter les messages transmis par le moniteur **runmqdnm** .NET.

utilisateur

```
public interface IBM.WMQMonitor.IMQObjectTrigger();
```

Selon que le contrôle de point de synchronisation est spécifié dans la commande **runmqdnm** , le message est supprimé de la file d'attente avant ou après le retour de la méthode Execute .

Des méthodes

void Execute (MQQueueManager *queueManager*, MQQueue *queue*, MQMessage *message*, string *param*);

queueManager
Gestionnaire de files d'attente hébergeant la file d'attente surveillée.

queue
File d'attente surveillée.

message
Message lu à partir de la file d'attente.

param
Données transmises à partir de UserParameter.

Interface MQC .NET

Faites référence à une constante MQI en ajoutant le préfixe MQC. au nom de la constante. MQC définit toutes les constantes utilisées par MQI.

utilisateur

```
System.Object
    │
    └─ IBM.WMQ.MQC
```

```
public interface IBM.WMQ.MQC extends System.Object;
```

Exemple

```
MQQueue queue;
queue.closeOptions = MQC.MQCO_DELETE;
```

Identificateurs de jeu de caractères pour les applications .NET

Description des jeux de caractères que vous pouvez sélectionner pour coder les messages .NET IBM WebSphere MQ

Jeu de caractères	Description
37	ibm037
437	ibm437 / PC d'origine
500	ibm500
819	iso-8859-1 / latin1 / ibm819
1 200	unicode

Jeu de caractères	Description
1208	UTF-8
273	ibm273
277	ibm277
278	ibm278
280	ibm280
284	ibm284
285	ibm285
297	ibm297
420	ibm420
424	ibm424
737	ibm737 / PC Grec
775	ibm775 / PC Baltique
813	iso-8859-7 / Greek / ibm813
838	ibm838
850	ibm850 / PC Latin 1
852	ibm852 / PC Latin 2
855	ibm855 / PC Cyrillique
856	ibm856
857	ibm857 / PC Turc
860	ibm860 / PC Portugais
861	ibm861 / PC islandais
862	ibm862 / PC Hébreu
863	ibm863 / PC Français canadien
864	ibm864 / PC Arabe
865	ibm865 / PC Nordic
866	ibm866 / PC Russe
868	ibm868
869	ibm869 / PC Grec moderne
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

Jeu de caractères	Description
915	iso-8859-5 / cyrillique / ibm915
916	iso-8859-8 / hebrew / ibm916
918	ibm918
920	iso-8859-9 / latin5 / ibm920
921	ibm921
922	ibm922
930	ibm930
932	PC Japonais
933	ibm933
935	ibm935
937	ibm937
939	ibm939
942	ibm942
943	ibm943
948	ibm948
949	ibm949
950	ibm950 / Big 5 Chinois traditionnel
954	EUCJIS
964	ibm964 / CNS 11643 Chinois traditionnel
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 Latin 2
1251	Windows Cyrillique
1252	Windows Latin 1
1253	Windows Grec
1254	Windows Turc

Jeu de caractères	Description
1255	Windows Hébreu
1256	Windows Arabe
1257	Windows Balte
1258	Windows Vietnamien
1381	ibm1381
1383	ibm1383
2022	JIS
5601	ksc-5601 Coréen
33722	ibm33722

IBM WebSphere MQ classes C++

Les classes C++ IBM WebSphere MQ encapsulent l'interface MQI (IBM WebSphere MQ Message Queue Interface). Il existe un fichier d'en-tête C++ unique, **imqi.hpp**, qui couvre toutes ces classes.

Pour chaque classe, les informations suivantes sont affichées:

Diagramme de hiérarchie de classes
Diagramme de classes montrant la classe dans sa relation d'héritage avec ses classes parent immédiates, le cas échéant.

Autres classes pertinentes
Le document contient des liens vers d'autres classes pertinentes, telles que les classes parent, et les classes d'objets utilisées dans les signatures de méthode.

Attributs d'objet
Attributs de la classe. Ces attributs s'ajoutent à ceux définis pour les classes parent. De nombreux attributs reflètent les membres de la structure de données WebSphere MQ (voir «Références croisées C++ et MQI», à la page 1326). Pour des descriptions détaillées, voir «Attributs des objets», à la page 787.

Constructeurs
Signatures des méthodes spéciales utilisées pour créer un objet de la classe.

Méthodes d'objet (public)
Signatures des méthodes qui nécessitent une instance de la classe pour leur fonctionnement et qui n'ont pas de restrictions d'utilisation.

Lorsqu'elle s'applique, les informations suivantes sont également affichées:

Méthodes de classe (public)
Signatures des méthodes qui ne nécessitent pas d'instance de la classe pour leur fonctionnement et qui n'ont pas de restrictions d'utilisation.

Méthodes surchargées (classe parent)
Signatures des méthodes virtuelles définies dans les classes parent, mais présentant un comportement polymorphe différent pour cette classe.

Méthodes d'objet (protégées)
Signatures des méthodes qui nécessitent une instance de la classe pour leur opération et qui sont réservées pour être utilisées par les implémentations de classes dérivées. Cette section n'intéresse que les auteurs de classe, par opposition aux utilisateurs de classe.

Données d'objet (protégées)
Détails d'implémentation des données d'instance d'objet disponibles pour les implémentations de classes dérivées. Cette section n'intéresse que les auteurs de classe, par opposition aux utilisateurs de classe.

Codes raison
Les valeurs MQRC_ * (voir Codes anomalie d'API) qui peuvent être attendues des méthodes qui échouent. Pour obtenir une liste exhaustive des codes anomalie pouvant se produire pour un objet d'une classe, consultez la documentation de la classe parent. La liste documentée des codes raison d'une classe n'inclut pas les codes raison des classes parent.

Remarque :

1. Les objets de ces classes ne sont pas sécurisés pour les unités d'exécution. Cela garantit des performances optimales, mais il est important de ne pas accéder à un objet à partir de plusieurs unités d'exécution.
2. Pour un programme à unités d'exécution multiples, il est recommandé d'utiliser un objet ImqQueueManager distinct pour chaque unité d'exécution. Chaque objet gestionnaire doit disposer de sa propre collection indépendante d'autres objets, ce qui garantit que les objets des différentes unités d'exécution sont isolés les uns des autres.

Ces classes sont les suivantes :

Références croisées C++ et MQI

Cette collection de rubriques contient des informations relatives à C++ pour l'interface MQI.

Lisez ces informations en même temps que «Types de données utilisés dans l'interface MQI», à la page 217.

Cette table associe les structures de données MQI aux classes C++ et aux fichiers d'inclusion. Les rubriques suivantes présentent des informations de référence croisée pour chaque classe C + +. Ces références croisées concernent l'utilisation des interfaces de procédure WebSphere MQ sous-jacentes.

Les classes ImqBinary, ImqDistributionList et ImqString ne comportent aucun attribut appartenant à cette catégorie et sont exclues.

Tableau 610. Référence croisée à la structure de données, à la classe et au fichier d'inclusion

Structure de données	Classe	fichier include
MQAIR	Enregistrement ImqAuthentication	imqair.hpp
	ImqBinary	imqbin.hpp
	ImqCache	imqcac.hpp
MQCD	ImqChannel	imqchl.hpp
MQCIH	En-tête ImqCICSBridge	imqcih.hpp
MQDLH	ImqDeadLetterHeader	imqdlh.hpp
MQOR	Liste ImqDistribution	imqdst.hpp
	ImqError	imqerr.hpp
MQGMO	ImqGetMessageOptions	imqgmo.hpp
	ImqHeader	imqhdr.hpp
MQIIH	ImqIMSBridge	imqiih.hpp
	ImqItem	imqitm.hpp
MQMD	ImqMessage	imqmsg.hpp
	Dispositif de suivi ImqMessage	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	Gestionnaire ImqQueue	imqmgr.hpp
MQRMH	En-tête ImqReference	imqrfh.hpp
	ImqString	imqstr.hpp
MQTM	ImqTrigger	imqtrg.hpp
MQTMC		
MQTMC2	ImqTrigger	imqtrg.hpp
MQXQH		
MQWIH	En-tête ImqWork	imqwih.hpp

ImqAuthentication-Référence croisée d'enregistrement

Référence croisée des attributs, des structures de données, des zones et des appels pour la classe C++ d'enregistrement ImqAuthentication.

Attribut	Structure de données	Zone	Appel
Nom de connexion	MQAIR	AuthInfoConnName	MQCONNX

Attribut	Structure de données	Zone	Appel
mot de passe	MQAIR	LDAPPassword	MQCONNX
type	MQAIR	AuthInfoType	MQCONNX
Nom d'utilisateur	MQAIR	LDAPUserNamePtr	MQCONNX
	MQAIR	LDAPUserNameDécalage	MQCONNX
	MQAIR	LDAPUserNameLongueur	MQCONNX

Référence croisée ImqCache

Référence croisée des attributs et des appels pour la classe C++ ImqCache .

Attribut	Appel
mémoire tampon automatique	MQGET
Capacité de la mémoire tampon	MQGET
pointeur de mémoire tampon	MQGET, MQPUT
Longueur de données	MQGET
Position des données	MQGET
pointeur de données	MQGET
LONGUEUR DE MESSAGE	MQGET, MQPUT

Référence croisée ImqChannel

Référence croisée des attributs, des structures de données, des zones et des appels pour la classe C++ ImqChannel .

Attribut	Structure de données	Zone	Appel
signal de présence par lots	MQCD	BatchHeartbeat	MQCONNX
nom de canal	MQCD	ChannelName	MQCONNX
Nom de connexion	MQCD	ConnectionName	MQCONNX
	MQCD	ShortConnectionNom	MQCONNX
Compression d'en-tête	MQCD	Liste HdrComp	MQCONNX
intervalle des signaux de présence	MQCD	HeartbeatInterval	MQCONNX
intervalle d'attente	MQCD	KeepAliveInterval	MQCONNX
adresse locale	MQCD	LocalAddress	MQCONNX
longueur maximale des messages	MQCD	Longueur maximale des messages	MQCONNX
Compression de message	MQCD	Liste MsgComp	MQCONNX
Nom de mode	MQCD	ModeName	MQCONNX
mot de passe	MQCD	Mot de passe	MQCONNX
nombre de sorties de réception	MQCD		MQCONNX

Attribut	Structure de données	Zone	Appel
noms d'exit de réception	MQCD	ReceiveExit	MQCONNX
	MQCD	ReceiveExitsdéfini	MQCONNX
	MQCD	ReceiveExitPtr	MQCONNX
recevoir des données utilisateur	MQCD	ReceiveUserData	MQCONNX
	MQCD	ReceiveUserDataPtr	MQCONNX
Nom d'exit de sécurité	MQCD	SecurityExit	MQCONNX
données utilisateur de sécurité	MQCD	Données SecurityUser	MQCONNX
Nombre d'exits d'envoi	MQCD		MQCONNX
noms d'exit d'émission	MQCD	SendExit	MQCONNX
	MQCD	SendExitsdéfini	MQCONNX
	MQCD	SendExitPtr	MQCONNX
envoyer des données utilisateur	MQCD	SendUserData	MQCONNX
	MQCD	SendUserDataPtr	MQCONNX
SSL CipherSpec	MQCD	Spécification sslCipher	MQCONNX
Type d'authentification du client SSL	MQCD	Authentification sslClient	MQCONNX
Nom de l'homologue SSL	MQCD	SSLPEERNAME	MQCONNX
Nom de programme transactionnel	MQCD	TpName	MQCONNX
type de transfert	MQCD	TransportType	MQCONNX
ID utilisateur	MQCD	UserIdentifier	MQCONNX

Références croisées de l'en-tête ImqCICSBridge

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'en-tête ImqCICSBridge.

Attribut	Structure de données	Zone
code de fin anormale de pont	MQCIH	AbendCode
Descripteur ADS	MQCIH	AdsDescriptor
identificateur d'attention	MQCIH	AttentionId
authentificateur	MQCIH	Authentificateur
code achèvement de pont	MQCIH	Code BridgeCompletion
décalage d'erreur de pont	MQCIH	ErrorOffset
code anomalie de pont	MQCIH	BridgeReason
code d'annulation de pont	MQCIH	CancelCode
tâche conversationnelle	MQCIH	ConversationalTask
Position de curseur	MQCIH	CursorPosition
jeton de fonction	MQCIH	Installation

Attribut	Structure de données	Zone
durée de conservation de l'installation	MQCIH	Heure FacilityKeep
fonction telle que	MQCIH	FacilityLike
fonction	MQCIH	Function
intervalle d'attente d'obtention	MQCIH	Intervalle GetWait
Type de lien	MQCIH	LinkType
identificateur de transaction suivant	MQCIH	ID NextTransaction
longueur des données de sortie	MQCIH	OutputDataLongueur
format de réponse	MQCIH	ReplyToFormat
code retour de pont	MQCIH	ReturnCode
code de début	MQCIH	StartCode
statut de fin de tâche	MQCIH	Statut TaskEnd
Identifiant de transaction	MQCIH	TransactionId
contrôle d'unité de travail	MQCIH	UowControl
version	MQCIH	Version

Référence croisée ImqDeadLetterHeader

Référence croisée des attributs, des structures de données et des zones pour la classe C++ ImqDeadLetterHeader .

Attribut	Structure de données	Zone
code raison de rebut	MQDLH	Motif
Nom du gestionnaire de files d'attente de destination	MQDLH	DestQMgrName
nom file d'attente de destination	MQDLH	DestQName
Nom d'application d'insertion	MQDLH	PutApplName
Type d'application d'insertion	MQDLH	PutApplType
Date d'insertion	MQDLH	PutDate
Heure d'insertion	MQDLH	PutTime

Référence croisée ImqError

Référence croisée des attributs et des appels pour la classe C++ ImqError .

Attribut	Appel
code achèvement	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET
code anomalie	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET

Référence croisée ImqGetMessageOptions

Référence croisée des attributs, des structures de données et des zones pour la classe C++ ImqGetMessageOptions .

Attribut	Structure de données	Zone
statut de groupe	MQGMO	GroupStatus
options de mise en correspondance	MQGMO	MatchOptions
jeton de message	MQGMO	MessageToken
options	MQGMO	Options
Nom de file d'attente résolu	MQGMO	ResolvedQName
longueur renvoyée	MQGMO	ReturnedLength
segmentation	MQGMO	Segmentation
statut de segment	MQGMO	SegmentStatus
	MQGMO	Signal1
	MQGMO	Signal2
participation au point de synchronisation	MQGMO	Options
Intervalle d'attente	MQGMO	WaitInterval

Référence croisée ImqHeader

Référence croisée des attributs, des structures de données et des zones pour la classe C++ ImqHeader .

Attribut	Structure de données	Zone
jeu de caractères	MQDLH, MQIIH	CodedCharSetId
codage	MQDLH, MQIIH	Codage
format	MQDLH, MQIIH	Format
indicateurs d'en-tête	MQIIH, MQRMH	Indicateurs

Références croisées de l'en-tête ImqIMSBridge

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Attribut	Structure de données	Zone
authentificateur	MQIIH	Authentificateur
Mode de validation	MQIIH	CommitMode
Substitution de terminal logique	MQIIH	LTermOverride
Nom de mappe des services de structuration de messages	MQIIH	MFSMapName
format de réponse	MQIIH	ReplyToFormat
Etendue de la sécurité	MQIIH	SecurityScope

Attribut	Structure de données	Zone
ID instance de transaction	MQIIH	ID TranInstance
ETAT DE TRANSACTION	MQIIH	TranState

Référence croisée ImqItem

Référence croisée des attributs et des appels pour la classe C++ ImqItem .

Attribut	Appel
ID structure	MQGET

Référence croisée ImqMessage

Référence croisée des attributs, des structures de données, des zones et des appels pour la classe C++ ImqMessage .

Attribut	Structure de données	Zone	Appel
Données d'ID application	MQMD	ApplIdentityData	
Données d'origine de l'application	MQMD	ApplOriginData	
Nombre d'annulations	MQMD	BackoutCount	
jeu de caractères	MQMD	CodedCharSetId	
codage	MQMD	Codage	
expiration	MQMD	Expiration	
format	MQMD	Format	
Indicateurs de message	MQMD	MsgFlags	
type de message	MQMD	MsgType	
position	MQMD	Décalage	
Longueur d'origine	MQMD	OriginalLength	
persistance	MQMD	Persistance	
priorité	MQMD	Priorité	
Nom d'application d'insertion	MQMD	PutApplName	
Type d'application d'insertion	MQMD	PutApplType	
Date d'insertion	MQMD	PutDate	
Heure d'insertion	MQMD	PutTime	
nom du gestionnaire de files d'attente de réponse	MQMD	ReplyToQMgr	
Nom de file d'attente de réponses	MQMD	ReplyToQ	
rapport	MQMD	Rapport	
numéro de séquence	MQMD	MsgSeqNumber	
longueur totale des messages		DataLength	MQGET

Attribut	Structure de données	Zone	Appel
ID utilisateur	MQMD	UserIdentifier	

Référence croisée de la fonction de suivi ImqMessage

Référence croisée des attributs, des structures de données et des zones pour la classe C++ du dispositif de suivi ImqMessage.

Attribut	Structure de données	Zone
Jeton de comptabilité	MQMD	AccountingToken
ID de corrélation	MQMD	CorrelId
commentaires en retour	MQMD	Commentaires
ID de groupe	MQMD	GroupId
ID de message	MQMD	MsgId

Référence croisée ImqNamelist

Référence croisée des attributs, des demandes et des appels pour la classe C++ ImqNamelist .

Attribut	Interrogation	Appel
Nombre de noms	NOM_NOM_NOM_MQT	MQINQ
Nom de la liste de noms	NOM_NOM_MQCA	MQINQ

Référence croisée ImqObject

Référence croisée des attributs, des structures de données, des zones, des demandes et des appels pour la classe C++ ImqObject .

Attribut	Structure de données	Zone	Interrogation	Appel
Date de modification			MQCA_ALTERATION_DATE	MQINQ
Heure de modification			MQCA_ALTERATION_TIME	MQINQ
Autre ID utilisateur	MQOD	AlternateUserid		
Autre ID de sécurité				
options de fermeture				MQCLOSE
description			MQCA_Q_DESC, MQCA_Q_MGR_DESC, MQCA_PROCESS_DESC	MQINQ
nom	MQOD	ObjectName	MQCA_Q_MGR_NAME, MQCQ_Q_NAME, MQCA_PROCESS_NAME	MQINQ
Options d'ouverture				MQOPEN
état ouvert				MQOPEN, MQCLOSE

Attribut	Structure de données	Zone	Interrogation	Appel
Identificateur du gestionnaire de files	Identificateur du gestionnaire de files		MQCA_Q_MGR_IDENTIFIER	MQINQ

Référence croisée ImqProcess

Référence croisée des attributs, des demandes et des appels pour la classe C++ d'enregistrement ImqAuthentication.

Attribut	Interrogation	Appel
ID application	ID_APPL_MQCA	MQINQ
Type d'application	TYPE_APPL_MQAI	MQINQ
Données d'environnement	DONNEES_ENV_MQCA	MQINQ
données utilisateur	DONNEES_UTILISATEUR_MQCA	MQINQ

Référence croisée ImqPutMessageOptions

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Tableau 611. Référence croisée ImqPutMessageOptions

Attribut	Structure de données	Zone
référence de contexte	MQPMO	Contexte
	MQPMO	InvalidDestCount
	MQPMO	KnownDestCount
options	MQPMO	Options
zones d'enregistrement	MQPMO	PutMsgRecFields
Nom de gestionnaire de files d'attente résolu	MQPMO	ResolvedQMgrNom
Nom de file d'attente résolu	MQPMO	ResolvedQName
	MQPMO	Délai d'attente
	MQPMO	UnknownDestCount
participation au point de synchronisation	MQPMO	Options

Référence croisée ImqQueue

Référence croisée des attributs, des structures de données, des zones, des demandes et des appels pour la classe C++ ImqQueue .

Tableau 612. Référence croisée ImqQueue

Attribut	Structure de données	Zone	Interrogation	Appel
Nom de file d'attente d'annulation			MQCA_BACKOUT_REQ_Q_NAME	MQINQ
Seuil d'annulation			MQIA_BACKOUT_SEUIL	MQINQ
nom file d'attente de base			MQCA_BASE_Q_NAME	MQINQ
Nom du cluster			NOM_CLUSTER_MQCA_	MQINQ
Liste de noms du cluster			MQCA_CLUSTER_NAMELIST	MQINQ
Rang charge cluster			MQIA_CLWL_Q_RANK	MQINQ
Priorité charge cluster			MQIA_CLWL_Q_PRIORITY	MQINQ
File d'attente d'utilisation de charge de travail de cluster			MQIA_CLWL_USEQ	MQINQ
date de création			DATE DE CREATION_MQCA	MQINQ
Date/heure de création			HEURE DE CREATION_MQCA	MQINQ
Longueur en cours			MQIA_CURRENT_Q_DEPTH	MQINQ
liaison par défaut			MQIA_BIND par défaut	MQINQ
Option d'ouverture en entrée par défaut			MQIA_DEF_INPUT_OPEN_OPTION	MQINQ
Persistance par défaut			MQIA_DEF_PERSISTENCE	MQINQ
Priorité par défaut			MQIA_DEF_PRIORITE	MQINQ
Type de définition			MQIA_DEFINITION_TYPE	MQINQ
événement de profondeur élevée			MQIA_Q_DEPTH_HIGH_EVENT	MQINQ
limite supérieure de la profondeur			MQIA_Q_DEPTH_HIGH_LIMIT	MQINQ
événement de profondeur faible			MQIA_Q_DEPTH_LOW_EVENT	MQINQ
limite inférieure de la profondeur			MQIA_Q_DEPTH_LOW_LIMIT	MQINQ
Evénement de profondeur maximale			MQIA_Q_DEPTH_MAX_LIMIT	MQINQ
listes de distribution			MQIA_DIST_LISTS	MQINQ, MQSET
Nom de la file d'attente dynamique	MQOD	DynamicQName		

Tableau 612. Référence croisée ImqQueue (suite)

Attribut	Structure de données	Zone	Interrogation	Appel
Sauvegarde nb d'annulations			MQIA_HARDEN_GET_BACKOUT	MQINQ
Type d'index			TYPE_INDEX_MQ	MQINQ
empêcher l'obtention			MQIA_INHIBIT_GET	MQINQ, MQSET
interdire l'insertion			MQIA_INHIBIT_PUT	MQINQ, MQSET
Nom de la file d'attente d'initialisation			NOM_Q_INITIATION_MQCA_	MQINQ
Profondeur maximale			MQIA_MAX_Q_DEPTH	MQINQ
longueur maximale des messages			LONGUEUR_MSG_MAX_MQIA_MAX	MQINQ
Séquence livraison messages			MQIA_MSG_DELIVERY_SEQUENCE	MQINQ
file d'attente répartie suivante				
Classe msgs non persistants			CLASSE_NPM_MQIA_	MQINQ
Nb d'ouvertures en entrée			MQIA_NOMBRE_ENTRÉES_OUVERTE	MQINQ
Nb d'ouvertures en sortie			MQIA_NOMBRE_SORTIES_OUVERTES	MQINQ
file d'attente distribuée précédente				
Nom de processus			MQCA_NOM_PROCESSUS	MQINQ
Comptabilité file			MQIA_ACCOUNTING_Q	MQINQ
nom de gestionnaire de files d'attente	MQOD	ObjectQMgrName		
Ctrl file d'attente			MQIA_MONITORING_Q	MQINQ
statistiques de file d'attente			MQIA_STATISTICS_Q	MQINQ
Type de file d'attente			MQIA_Q_TYPE	MQINQ
Nom du gestionnaire de files d'attente éloignées			MQCA_REMOTE_Q_MGR_NAME	MQINQ
Nom de la file d'attente distante			MQCA_NOM_QTE_DISTANT	MQINQ

Tableau 612. Référence croisée ImqQueue (suite)

Attribut	Structure de données	Zone	Interrogation	Appel
Nom de gestionnaire de files d'attente résolu	MQOD	ResolvedQMgrNom		
Nom de file d'attente résolu	MQOD	ResolvedQName		
Intervalle de conservation			INTERVALLE_CONSERVATION-MQIA_RETENTION_INTERVALLE	MQINQ
portée			PORT_MQAI	MQINQ
intervalle de maintenance			MQIA_Q_SERVICE_INTERVAL	MQINQ
événement d'intervalle de maintenance			MQIA_Q_ÉVÉNEMENT_INTERVALLE_SERVICE	MQINQ
Partage de file d'attente			MQIA_SHAREABILITY	MQINQ
classe d'archivage			MQCA_STORAGE_CLASS	MQINQ
Nom de la file d'attente de transmission			MQCA_XMIT_Q_NAME	MQINQ
Contrôle du déclenchement			MQIA_TRIGGER_CONTROL	MQINQ, MQSET
Données de déclenchement			DONNEES_TRIGGER_MQCA	MQINQ, MQSET
Longueur de déclenchement			PROFONDEUR-MQIA_TRIGGER_PROFONDEUR	MQINQ, MQSET
Priorité msg pour déclench.			MQIA_TRIGGER_MSG_PRIORITY	MQINQ, MQSET
type du déclencheur			TYPE_TRIGGER_MQIA_	MQINQ, MQSET
utilisation			MQIA_UTILISATION	MQINQ

Références croisées du gestionnaire ImqQueue

Référence croisée des attributs, des structures de données, des zones, des demandes et des appels pour la classe C++ du gestionnaire ImqQueue.

Attribut	Structure de données	Zone	Interrogation	Appel
remplacement des connexions de comptabilité			MQIA_ACCOUNTING_CONN_OVERRIDE	MQINQ

Attribut	Structure de données	Zone	Interrogation	Appel
Intervalle de comptabilité			MQIA_ACCOUNTING_INTERVAL	MQINQ
Enregistrement de l'activité			ENREGISTREMENT_ACTIVITÉ_MQIA_ENREGISTREMENT	MQINQ
Adoption de la vérification du nouveau MCA			MQIA_ADOPTNEWMCA_CHECK	MQINQ
Adoption du nouveau type de MCA			MQIA_ADOPTNEWMCA_TYPE	MQINQ
Type d'authentification	MQCSP	AuthenticationType		MQCONNX
événement d'autorité			MQIA_AUTHORITY_EVENT	MQINQ
Options de début	Objet MQBO	Options		MQBEGIN
événement de pont			MQIA_ÉVÉNEMENT_PONT	MQINQ
Définition automatique de canal			MQIA_CHANNEL_AUTO_DEF	MQINQ
événement de définition automatique de canal			MQIA_CHANNEL_AUTO_EVENT	Interface MQIA
Exit de définition automatique de canal			MQIA_CHANNEL_AUTO_EXIT	Interface MQIA
événement Canal			MQIA_CHANNEL_EVENT	MQINQ
Adaptateurs d'initiateur de canal			MQIA_CHINIT_ADAPTERS	MQINQ
Ctrl init. canal			MQIA_CHINIT_CONTROL	MQINQ
Répartiteurs d'initiateur de canal			MQIA_CHINIT_RÉPARTITEURS	MQINQ
Démarrage automatique de la trace de l'initiateur de canal			MQIA_CHINIT_TRACE_AUTO_START	MQINQ
Taille de la table de trace de l'initiateur de canal			MQIA_CHINIT_TRACE_TABLE_SIZE	MQINQ

Attribut	Structure de données	Zone	Interrogation	Appel
Contrôle des canaux			MQIA_CANAL_SURVEILLANCE	MQINQ
référence de canal	MQCD	ChannelType		MQCONNX
Statistiques de canal			MQIA_STATISTICS_CANAL	MQINQ
jeu de caractères			MQIA_CODED_CHAR_SET_ID	MQINQ
Ctrl émetteur cluster			MQIA_MONITORING_AUTO_CLUSSDR	MQINQ
Stats émetteur cluster			MQIA_STATISTICS_AUTO_CLUSSDR	MQINQ
Données de charge de travail de cluster			MQCA_CLUSTER_DONNÉES_CHARGE_TRAVAIL_CLUSTER	MQINQ
Exit de charge de travail du cluster			MQCA_CLUSTER_WORKLOAD_EXIT	MQINQ
Longueur de charge de travail de cluster			MQIA_CLUSTER_LONGUEUR_TRAVAIL_CLUSTER	MQINQ
mru de charge de travail de cluster			MQIA_CLWL_MRU_CHANNELS	MQINQ
File d'attente d'utilisation de charge de travail de cluster			MQIA_CLWL_USEQ	MQINQ
événement de commande			ÉVÉNEMENT_COMMANDE_MQIA_COMMANDE	MQINQ
Nom de la file d'attente d'entrée des commandes			MQCA_COMMAND_INPUT_Q_NAME	MQINQ
Niveau de commande			MQIA_NIVEAU DE COMMANDE	MQINQ
Contrôle du serveur de commandes			MQIA_CMD_SERVER_CONTROL	MQINQ
Options de connexion	MQCNO	Options		MQCONN, MQCONNX
ID de connexion	MQCNO	ConnectionId		MQCONNX
statut des connexions				MQCONN, MQCONNX, MQDISC
Balise de connexion	MQCD	ConnTag		MQCONNX

Attribut	Structure de données	Zone	Interrogation	Appel
Matériel de cryptographie	MQSCO	CryptoHardware		MQCONNX
Nom de la file d'attente de rebut			MQCA_NOM_LETTRE_INTERBLOCAGE	MQINQ
nom de file de transmission par défaut			MQCA_DEF_XMIT_Q_NAME	MQINQ
listes de distribution			MQIA_DIST_LISTS	MQINQ
groupe dns			GROUPE_DNS_MQCA	MQINQ
dns wlm			MQIA_DNS_WLM	MQINQ
premier enregistrement d'authentification	MQSCO	AuthInfoRecOffset		MQCONNX
	MQSCO	AuthInfoRecPtr		MQCONNX
événement d'interdiction			MQIA_INHIBIT_EVENT	MQINQ
Version adresse IP			MQIA_IP_ADRESSE_VERSION	MQINQ
référentiel de clés	MQSCO	KeyRepository		MQCONNX
nombre de réinitialisations de clé	MQSCO	Nombre de KeyReset		MQCONNX
Délai du programme d'écoute			MQIA_LISTENER_TIMER	MQINQ
événement local			MQIA_LOCAL_EVENT	MQINQ
Evénement consignateur			MQIA_LOGGER_EVENT	MQINQ
Nom de groupe de LU			NOM_GROUPE_LU_MQCA	MQINQ
Nom de LU			NOM_LU_MQCA	MQINQ
Suffixe de bras lu62			MQCA_LU62_ARM_SUFFIX	MQINQ
Canaux lu62			MQIA_LU62_CHANNELS	MQINQ
nombre maximal de canaux actifs			MQIA_ACTIVE_CHANNELS	MQINQ
Nombre maximal de canaux			CANAL_MAX_MQIA_CANAUX	MQINQ
nombre maximal de descripteurs			MQIA_MAX_DESCRIPTEURS	MQINQ

Attribut	Structure de données	Zone	Interrogation	Appel
longueur maximale des messages			LONGUEUR_MSG_MAX_MQIA_MAX	MQINQ
Priorité maximum			MQIA_MAX_PRIORITE	MQINQ
Nb max. messages non validés			MQIA_MAX_UNCOMMITTED_MSGS	MQINQ
Comptabilité MQI			MQIA_ACCOUNTING_MQI	MQINQ
Statistiques MQI			MQIA_STATISTICS_MQI	MQINQ
port de communications sortantes maximum			MQIA_OUTBOUND_PORT_MAX	MQINQ
port de communications sortantes minimum			MQIA_OUTBOUND_PORT_MIN	MQINQ
mot de passe	MQCSP	CSPPasswordPtr		MQCONNX
	MQCSP	CSPPasswordOffset		MQCONNX
	MQCSP	CSPPasswordLength		MQCONNX
événement Performances			ÉVÉNEMENT_PERFORMANCE_MQI	MQINQ
plateforme			MQIA_PLATEFORME	MQINQ
Comptabilité file			MQIA_ACCOUNTING_Q	MQINQ
Ctrl file d'attente			MQIA_MONITORING_Q	MQINQ
statistiques de file d'attente			MQIA_STATISTICS_Q	MQINQ
Délai de réception			DÉLAI_RÉCEPTION_MQIA_EXPIRATION	MQINQ
délai d'attente minimal de réception			MQIA_RECEIVE_TIMEOUT_MIN	MQINQ
Type de délai de réception			MQIA_RECEIVE_TIMEOUT_TYPE	MQINQ
événement distant			EVÉNEMENT MQIA_REMOTE_EVENT	MQINQ
Nom du référentiel			MQCA_NOM_RÉFÉRENTIEL	MQINQ
Liste de noms du référentiel			MQCA_REPOSITORY_NAMELIST	MQINQ
nom du gestionnaire de files d'attente partagées			MQIA_SHARED_Q_Q_MGR_NAME	MQINQ

Attribut	Structure de données	Zone	Interrogation	Appel
événement ssl			MQIA_SSL_EVENT	MQINQ
Fips ssl			MQIA_SSL_FIPS_REQUIRED	MQINQ
Nbre avant réinit clé SSL			MQIA_SSL_RESET_COUNT	MQINQ
événement de démarrage / arrêt			MQIA_START_STOP_EVENT	MQINQ
Intervalle de statistiques			INTERVALLE MQIA_STATISTICS_INTERVAL	MQINQ
Prise en charge points sync.			MQIA_SYNCPOINT	MQINQ
canaux tcp			MQIA_TCP_CHANNELS	MQINQ
Signal de présence TCP			MQIA_TCP_KEEP_ALIVE	MQINQ
Nom TCP			NOM_TCP_MQCA	MQINQ
Type de pile TCP			MQIA_TCP_STACK_TYPE	MQINQ
Enregistrement trace route			MQIA_TRACE_ROUTE_ENREGISTREMENT	MQINQ
Intervalle de déclenchement			INTERVALLE_DÉCLENCHEUR_MQIA_	MQINQ
ID utilisateur	MQCSP	CSPUserIdPtr		MQCONNX
	MQCSP	CSPUserIdDécalage		MQCONNX
	MQCSP	CSPUserIdLongueur		MQCONNX

Référence croisée d'en-tête ImqReference

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Attribut	Structure de données	Zone
Environnement cible	MQRMH	DestEnvLongueur, DestEnvDécalage
nom de destination	MQRMH	DestNameLongueur, DestNameDécalage
ID d'instance	MQRMH	ID ObjectInstance
longueur logique	MQRMH	Longueur DataLogical
décalage logique	MQRMH	DataLogical
décalage logique 2	MQRMH	DataLogicalOffset2
Type de référence	MQRMH	ObjectType
Environnement source	MQRMH	SrcEnvLongueur, SrcEnvDécalage
Nom de source	MQRMH	SrcNameLongueur, SrcNameDécalage

Référence croisée ImqTrigger

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Tableau 613. Référence croisée ImqTrigger

Attribut	Structure de données	Zone
ID application	MQTM	ApplId
Type d'application	MQTM	ApplType
Données d'environnement	MQTM	EnvData
Nom de processus	MQTM	ProcessName
Nom de la file d'attente	MQTM	nomQ
Données de déclenchement	MQTM	TriggerData
données utilisateur	MQTM	UserData

Référence croisée d'en-tête ImqWork

Référence croisée des attributs, des structures de données et des zones pour la classe C++ d'enregistrement ImqAuthentication.

Attribut	Structure de données	Zone
jeton de message	MQWIH	MessageToken
nom de service	MQWIH	ServiceName
étape de service	MQWIH	ServiceStep

Classe C++ d'enregistrement ImqAuthentication

Cette classe encapsule un enregistrement d'informations d'authentification (MQAIR) à utiliser lors de l'exécution de la méthode ImqQueueManager: :connect, pour les connexions client SSL personnalisées.

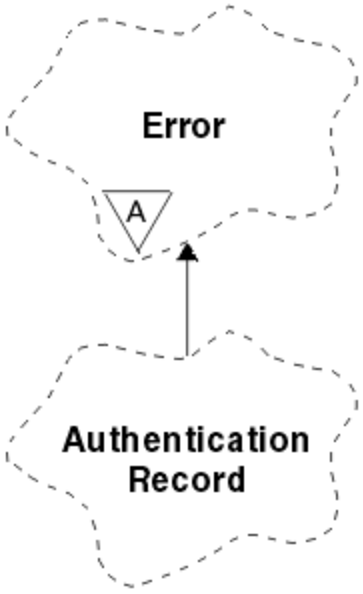


Figure 46. Classe d'enregistrement ImqAuthentication

Pour plus de détails, voir la description de la méthode ImqQueueManager: :connect. Cette classe n'est pas disponible sur la plateforme z/OS .

Attributs d'objet

Nom de connexion
Nom de la connexion au serveur CRL LDAP. Il s'agit de l'adresse IP ou du nom DNS, suivi éventuellement du numéro de port, entre parenthèses.

référence de connexion
Référence à un objet gestionnaire ImqQueuequi fournit la connexion requise à un gestionnaire de files d'attente (local). La valeur initiale est zéro. Ne le confondez pas avec le nom du gestionnaire de files d'attente qui identifie un gestionnaire de files d'attente (éventuellement distant) pour une file d'attente nommée.

enregistrement d'authentification suivant
Objet suivant de cette classe, sans ordre particulier, ayant la même **référence de connexion** que cet objet. La valeur initiale est zéro.

mot de passe
Mot de passe fourni pour l'authentification de connexion au serveur CRL LDAP.

enregistrement d'authentification précédent
Objet précédent de cette classe, sans ordre particulier, ayant la même **référence de connexion** que cet objet. La valeur initiale est zéro.

type
Type des informations d'authentification contenues dans l'enregistrement.

Nom d'utilisateur
Identificateur utilisateur fourni pour l'autorisation sur le serveur CRL LDAP.

Constructeurs

ImqAuthenticationRecord () ;
Constructeur par défaut.

Méthodes d'objet (public)

void operator = (const ImqAuthenticationRecord & *air*) ;
Copie les données d'instance à partir de *air*, en remplaçant les données d'instance existantes.

const ImqString & connectionName () const ;
Renvoie le **nom de connexion**.

void setConnectionNom (const ImqString & *nom*) ;
Définit le **nom de connexion**.

void setConnectionName (const char * *name* = 0) ;
Définit le **nom de connexion**.

ImqQueueManager * connectionReference () const ;
Renvoie la **référence de connexion**.

void setConnectionReference (ImqQueueManager & *manager*) ;
Définit la **référence de connexion**.

void setConnectionReference (ImqQueueManager * *manager* = 0) ;
Définit la **référence de connexion**.

void copyOut (MQAIR * *pAir*) ;
Copie les données d'instance dans *pAir*, en remplaçant les données d'instance existantes. Cela peut impliquer l'allocation de mémoire dépendante.

void clear (MQAIR * *pAir*) ;
Efface la structure et libère le stockage dépendant référencé par *pAir*.

ImqAuthenticationEnregistrement * nextAuthenticationEnregistrement () const ;
Renvoie l' **enregistrement d'authentification suivant**.

const ImqString & password () const ;
Renvoie le **mot de passe**.

void setPassword (const ImqString & *mot_de_passe*) ;
Définit le **mot de passe**.

void setPassword (const char * *mot_de_passe* = 0) ;
Définit le **mot de passe**.

ImqAuthenticationRecord * previousAuthenticationRecord () const ;
Renvoie l' **enregistrement d'authentification précédent**.

Type MQLONG () const ;
Renvoie le **type**.

void setType (const MQLONG *type*) ;
Définit le **type**.

const ImqString & userName () const ;
Renvoie le **nom d'utilisateur**.

void setUserName (const ImqString & *nom*) ;
Définit le **nom d'utilisateur**.

void setUserName (const char * *name* = 0) ;
Définit le **nom d'utilisateur**.

Méthodes d'objet (protégées)

void setNextAuthenticationRecord (ImqAuthenticationRecord * *pAir* = 0) ;
Définit l' **enregistrement d'authentification suivant**.

Attention: utilisez cette fonction uniquement si vous êtes certain qu'elle ne rompt pas la liste des enregistrements d'authentification.

void setPreviousAuthenticationRecord (ImqAuthenticationRecord * *pAir* = 0) ;
Définit l' **enregistrement d'authentification précédent**.

Attention: utilisez cette fonction uniquement si vous êtes certain qu'elle ne rompt pas la liste des enregistrements d'authentification.

Classe C++ ImqBinary

Cette classe encapsule un tableau d'octets binaire qui peut être utilisé pour les valeurs ImqMessage **jeton de comptabilité**, **ID de corrélation**et **ID de message** . Il facilite l'affectation, la copie et la comparaison.

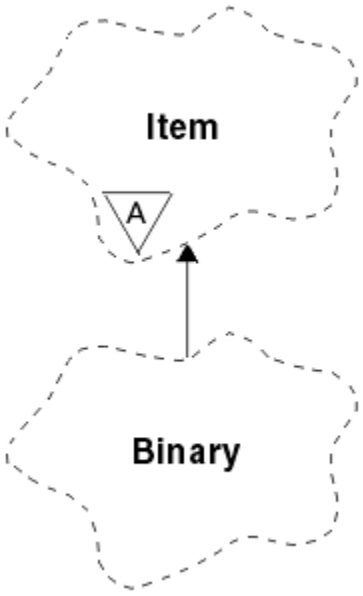


Figure 47. Classe ImqBinary

Attributs d'objet

data
Tableau d'octets de données binaires. La valeur initiale est null.

Longueur de données
Nombre d'octets. La valeur initiale est zéro.

Pointeur de données
Adresse du premier octet des **données**. La valeur initiale est zéro.

Constructeurs

ImqBinary( );
Constructeur par défaut.

ImqBinary(const ImqBinary & *binaire*) ;
Constructeur de copie.

ImqBinary(const void * *data*, const size_t *longueur*) ;
Copie *longueur* octets à partir de *données*.

Méthodes ImqItem surchargées

virtual ImqBoolean copyOut(ImqMessage & *msg*) ;
Copie les **données** dans la mémoire tampon de messages, en remplaçant tout contenu existant. Définit le *msg* **format** sur MQFMT_NONE.

Voir la description de la méthode de classe ImqItem pour plus de détails.

virtuel ImqBoolean pasteIn(ImqMessage & ***msg*** **) ;**
Définit les **données** en transférant les données restantes de la mémoire tampon de messages, en remplaçant les **données**existantes.

Pour que l'opération aboutisse, le format ImqMessage doit être MQFMT_NONE.

Voir la description de la méthode de classe ImqItem pour plus de détails.

Méthodes d'objet (public)

void operator = (const ImqBinary & ***binaire*** **) ;**
Copie des octets à partir de *binaire*.

ImqBoolean operator == (const ImqBinary & ***binaire*** **) ;**
Compare cet objet à *binaire*. Elle renvoie FALSE si elle n'est pas égale et TRUE dans le cas contraire. Les objets sont égaux s'ils ont la même **longueur de données** et que les octets correspondent.

ImqBoolean copyOut(void * ***buffer*****, const size_t** ***length*****, const char** ***pad*** **= 0) ;**
Copie jusqu'à *longueur* octets à partir du **pointeur de données** vers *buffer*. Si la **longueur des données** est insuffisante, l'espace restant dans la mémoire tampon est rempli avec *pad* octets. *buffer* peut être égal à zéro si *length* est également égal à zéro. *longueur* ne doit pas être négative. Elle renvoie TRUE en cas de succès.

size_t dataLength() const ;
Renvoie la **longueur des données**.

ImqBoolean setDataLongueur(const size_t ***longueur*** **) ;**
Définit la **longueur des données**. Si la **longueur des données** est modifiée suite à cette méthode, les données de l'objet ne sont pas initialisées. Elle renvoie TRUE en cas de succès.

void * dataPointer() const ;
Renvoie le **pointeur de données**.

ImqBoolean isNull() const ;
Renvoie TRUE si la **longueur des données** est égale à zéro ou si tous les octets de **données** sont égaux à zéro. Sinon, elle retourne FALSE.

ImqBoolean set(const void * ***buffer*****, const size_t** ***longueur*** **) ;**
Copie *longueur* octets de *mémoire tampon*. Elle renvoie TRUE en cas de succès.

Méthodes d'objet (protégées)

void clear() ;
Réduit la **longueur des données** à zéro.

Codes raison

- MQRC_NO_BUFFER
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_INCONSISTENT_FORMAT

Classe C++ ImqCache

Utilisez cette classe pour stocker ou convertir des données en mémoire.

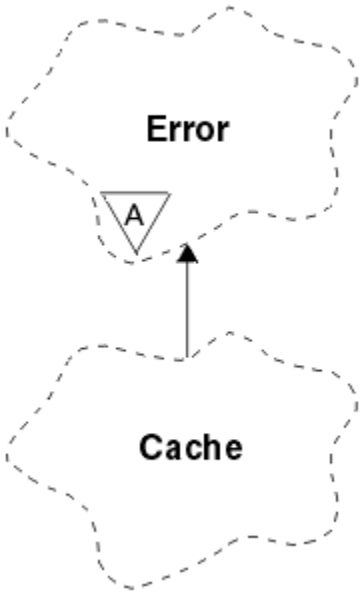


Figure 48. Classe ImqCache

Utilisez cette classe pour stocker ou convertir des données en mémoire. Vous pouvez désigner une mémoire tampon de taille fixe ou le système peut fournir une quantité de mémoire flexible automatiquement. Cette classe est liée aux appels MQI répertoriés dans «Référence croisée ImqCache», à la page 1328.

Attributs d'objet

mémoire tampon automatique
Indique si la mémoire tampon est gérée automatiquement par le système (TRUE) ou si elle est fournie par l'utilisateur (FALSE). Il est initialement défini sur TRUE.

Cet attribut n'est pas défini directement. Il est défini indirectement à l'aide de la méthode **useEmptyBuffer** ou de la méthode **useFullBuffer** .

Si le stockage utilisateur est fourni, cet attribut est FALSE, la mémoire tampon ne peut pas augmenter et des erreurs de dépassement de mémoire tampon peuvent se produire. L'adresse et la longueur de la mémoire tampon restent constantes.

Si le stockage utilisateur n'est pas fourni, cet attribut est TRUE et la mémoire tampon peut croître de manière incrémentielle pour contenir une quantité arbitraire de données de message. Cependant, lorsque la taille de la mémoire tampon augmente, l'adresse de la mémoire tampon peut changer. Par conséquent, soyez prudent lorsque vous utilisez le **pointeur de mémoire tampon** et le **pointeur de données**.

Capacité de la mémoire tampon
Nombre d'octets de mémoire dans la mémoire tampon. La valeur initiale est zéro.

Pointeur de mémoire tampon
Adresse de la mémoire tampon. La valeur initiale est null.

Longueur de données
Nombre d'octets suivant le **pointeur de données**. Cette valeur doit être inférieure ou égale à la **longueur du message**. La valeur initiale est zéro.

Position des données
Nombre d'octets précédant le **pointeur de données**. Cette valeur doit être inférieure ou égale à la **longueur du message**. La valeur initiale est zéro.

Pointeur de données
Adresse de la partie de la mémoire tampon qui doit être écrite ou lue à partir de la suivante. La valeur initiale est null.

LONGUEUR DE MESSAGE
Nombre d'octets de données significatives dans la mémoire tampon. La valeur initiale est zéro.

Constructeurs

ImqCache( );
Constructeur par défaut.

ImqCache(const ImqCache & ***cache*** **) ;**
Constructeur de copie.

Méthodes d'objet (public)

void operator = (const ImqCache & ***cache*** **) ;**
Copie jusqu'à **longueur de message** octets de données de l'objet *cache* vers l'objet. Si la **mémoire tampon automatique** est FALSE, la **longueur de la mémoire tampon** doit déjà être suffisante pour accueillir les données copiées.

ImqBoolean automaticBuffer() const ;
Renvoie la valeur de la **mémoire tampon automatique** .

size_t bufferLength() const ;
Renvoie la **longueur de la mémoire tampon**.

char * bufferPointer() const ;
Renvoie le **pointeur de mémoire tampon**.

void clearMessage() ;
Définit la **longueur de message** et le **décalage de données** sur zéro.

size_t dataLength() const ;
Renvoie la **longueur des données**.

size_t dataOffset() const ;
Renvoie le **décalage de données**.

ImqBoolean setDataDécalage(const size_t ***décalage*** **) ;**
Définit le **décalage de données**. La **longueur de message** est augmentée si nécessaire pour s'assurer qu'elle n'est pas inférieure au **décalage de données**. Cette méthode renvoie TRUE en cas de succès.

char * dataPointer() const ;
Renvoie une copie du **pointeur de données**.

size_t messageLength() const ;
Renvoie la **longueur de message**.

ImqBoolean setMessageLongueur(const size_t ***longueur*** **) ;**
Définit la **longueur de message**. Augmente la **longueur de la mémoire tampon** si nécessaire pour que la **longueur de message** ne soit pas supérieure à la **longueur de la mémoire tampon**. Réduit le **décalage de données** si nécessaire pour s'assurer qu'il n'est pas supérieur à la **longueur de message**. Elle renvoie TRUE en cas de succès.

ImqBoolean moreBytes(const size_t ***bytes-required*** **) ;**
S'assure que *bytes-required* plus d'octets sont disponibles (pour l'écriture) entre le **pointeur de données** et la fin de la mémoire tampon. Elle renvoie TRUE en cas de succès.

Si **automatic buffer** a la valeur TRUE, davantage de mémoire est acquise selon les besoins ; sinon, la **longueur de la mémoire tampon** doit déjà être adéquate.

ImqBoolean read(const size_t *longueur*, char * & *mémoire tampon externe*) ;
Copie *longueur* octets, à partir de la mémoire tampon commençant à la position **pointeur de données** , dans la *mémoire tampon externe*. Une fois les données copiées, le **décalage de données** est augmenté de *longueur*. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean resizeBuffer(const size_t *longueur*) ;
Fait varier la **longueur de la mémoire tampon**, à condition que la **mémoire tampon automatique** soit TRUE. Ceci est réalisé en réallouant la mémoire tampon. Jusqu'à **longueur de message** octets de données de la mémoire tampon existante sont copiés dans la nouvelle. Le nombre maximal d'octets copiés est de *longueur* . Le **pointeur de mémoire tampon** est modifié. La **longueur des messages** et le **décalage des données** sont préservés aussi près que possible dans les limites de la nouvelle mémoire tampon. Elle renvoie TRUE en cas de succès, et FALSE si la **mémoire tampon automatique** est FALSE.

Remarque : Cette méthode peut échouer avec MQRC_STORAGE_NOT_AVAILABLE en cas de problème avec les ressources système.

ImqBoolean useEmptyBuffer(const char * *external-buffer*, const size_t *longueur*) ;
Identifie une mémoire tampon utilisateur vide, en définissant le **pointeur de mémoire tampon** pour qu'il pointe vers *external-buffer*, la **longueur de mémoire tampon** sur *longueur*et la **longueur de message** sur zéro. Effectue un **clearMessage**. Si la mémoire tampon est entièrement amorcée avec des données, utilisez la méthode **useFullBuffer** à la place. Si la mémoire tampon est partiellement primée avec des données, utilisez la méthode **setMessageLength** pour indiquer la quantité correcte. Cette méthode renvoie TRUE en cas de succès.

Cette méthode peut être utilisée pour identifier une quantité fixe de mémoire, comme décrit précédemment (*external-buffer* n'est pas null et *length* est différent de zéro), auquel cas **automatic buffer** est défini sur FALSE, ou elle peut être utilisée pour rétablir la mémoire flexible gérée par le système (*external-buffer* est null et *length* est zéro), auquel cas **automatic buffer** est défini sur TRUE.

ImqBoolean useFullBuffer(const char * *externalBuffer*, const size_t *longueur*) ;
Comme pour **useEmptyBuffer**, sauf que la **longueur de message** est définie sur *length*. Elle renvoie TRUE en cas de succès.

ImqBoolean write(const size_t *longueur*, const char * *mémoire tampon externe*) ;
Copie *longueur* octets, à partir de la *mémoire tampon externe*, dans la mémoire tampon à partir de la position **pointeur de données** . Une fois les données copiées, le **décalage de données** est augmenté de *longueur*et la **longueur de message** est augmentée si nécessaire pour s'assurer qu'elle n'est pas inférieure à la nouvelle valeur de **décalage de données** . Cette méthode renvoie TRUE en cas de succès.

Si **automatic buffer** a pour valeur TRUE, une quantité de mémoire adéquate est garantie ; dans le cas contraire, le **décalage de données** final ne doit pas dépasser la **longueur de la mémoire tampon**.

Codes raison

- MQRC_BUFFER_NOT_AUTOMATIC
- MQRC_DATA_TRONQUÉ
- MQRC_INSUFFISENT_BUFFER
- MQRC_DONNÉES_INSUFFISANTES
- POINT_NULL_MQRC_NULL
- MQRC_STORAGE_NOT_AVAILABLE
- LONGUEUR_ZÉRO_MQRC_

Classe C++ ImqChannel

Cette classe encapsule une définition de canal (MQCD) à utiliser lors de l'exécution de la méthode Manager: :connect pour les connexions client personnalisées.

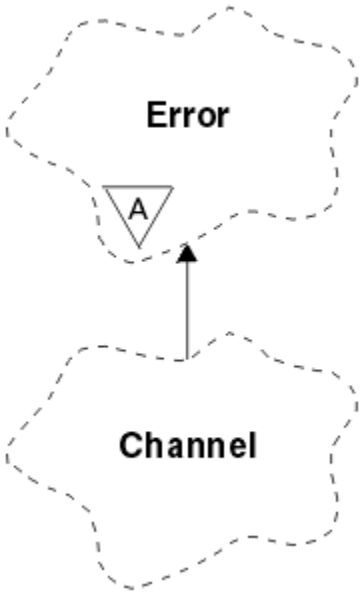


Figure 49. Classe ImqChannel

Pour plus de détails, voir la description de la méthode Manager: :connect et Exemple de programme HELLO WORLD (imqwrld.cpp) . Toutes les méthodes répertoriées ne sont pas applicables à toutes les plateformes ; voir les descriptions des commandes DEFINE CHANNEL et ALTER CHANNEL dans Référence MQSC pour plus de détails. La classe ImqChannel n'est pas prise en charge sur z/OS.

- «Attributs d'objet», à la page 1351
- «Constructeurs», à la page 1352
- «Méthodes d'objet (public)», à la page 1352
- «Codes raison», à la page 1356

Attributs d'objet

signal de présence par lots
Nombre de millisecondes entre les vérifications qu'un canal distant est actif. La valeur initiale est 0.

Nom du canal
Nom du canal. La valeur initiale est null.

Nom de connexion
Nom de la connexion. Par exemple, l'adresse IP d'un ordinateur hôte. La valeur initiale est null.

Compression d'en-tête
Liste des techniques de compression de données d'en-tête prises en charge. Les valeurs initiales sont toutes définies sur MQCOMPRESS_NOT_AVAILABLE.

intervalle des signaux de présence
Nombre de secondes entre les vérifications qu'une connexion fonctionne toujours. La valeur initiale est 300.

intervalle d'attente
Nombre de secondes passées à la pile de communications en spécifiant le délai de signal de présence pour le canal. La valeur initiale est MQKAI_AUTO.

adresse locale
Adresse de communication locale du canal.

longueur maximale des messages
Longueur maximale de message prise en charge par le canal dans une seule communication. La valeur initiale est 4 194 304.

Compression de message
Liste des techniques de compression de données de message prises en charge. Les valeurs initiales sont toutes définies sur MQCOMPRESS_NOT_AVAILABLE.

Nom de mode
Nom du mode. La valeur initiale est null.

password
Mot de passe fourni pour l'authentification de la connexion. La valeur initiale est null.

nombre d'exits de réception
Nombre d'exits de réception. La valeur initiale est zéro. Cet attribut est en lecture seule.

noms d'exit de réception
Noms des exits de réception.

réception des données utilisateur
Données associées aux exits de réception.

Nom d'exit de sécurité
Nom d'un exit de sécurité à appeler côté serveur de la connexion. La valeur initiale est null.

données utilisateur de sécurité
Données à transmettre à l'exit de sécurité. La valeur initiale est null.

nombre d'exits d'émission
Nombre d'exits d'émission. La valeur initiale est zéro. Cet attribut est en lecture seule.

noms d'exit d'émission
Noms des exits d'émission.

envoyer des données utilisateur
Données associées aux exits d'envoi.

SSL CipherSpec
CipherSpec à utiliser avec SSL.

Type d'authentification du client SSL
Type d'authentification client à utiliser avec SSL.

Nom de l'homologue SSL
Nom d'homologue à utiliser avec SSL.

Nom de programme transactionnel
Nom du programme de transaction. La valeur initiale est null.

Type de transfert
Type de transport de la connexion. La valeur initiale est MQXPT_LU62.

ID utilisateur
Identificateur utilisateur fourni pour l'autorisation. La valeur initiale est null.

Constructeurs

ImqChannel( ) ;
Constructeur par défaut.

ImqChannel(const ImqChannel & *canal*) ;
Constructeur de copie.

Méthodes d'objet (public)

void operator = (const ImqChannel & *canal*) ;
Copie les données d'instance à partir du *canal*, en remplaçant les données d'instance existantes.

MQLONG batchHeartBeat () const ;
Renvoie le **signal de présence du lot**.

ImqBoolean setBatchHeartBeat(const MQLONG *heartbeat* = 0L) ;
Définit le **signal de présence du lot** . Cette méthode renvoie TRUE en cas de succès.

ImqString channelName() const ;
Renvoie le **nom de canal**.

ImqBoolean setChannelNom (const car * *nom* = 0) ;
Définit le **nom de canal**. Cette méthode renvoie TRUE en cas de succès.

ImqString connectionName() const ;
Renvoie le **nom de connexion**.

ImqBoolean setConnectionNom (const char * *nom* = 0) ;
Définit le **nom de connexion**. Cette méthode renvoie TRUE en cas de succès.

size_t headerCompressionNombre () const ;
Renvoie le nombre de techniques de compression de données d'en-tête prises en charge.

ImqBoolean headerCompression(nombre de tailles de const, compression MQLONG [ ]) const ;
Renvoie des copies des techniques de compression de données d'en-tête prises en charge dans **compress**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setHeaderCompression (nombre de tailles de const, compression MQLONG de const [ ]) ;

Définit les techniques de compression de données d'en-tête prises en charge sur **compress**.

Définit le nombre de techniques de compression de données d'en-tête prises en charge sur **count**.

Cette méthode renvoie TRUE en cas de succès.

Intervalle heartBeatMQLONG () const ;
Renvoie l' **intervalle des signaux de présence**.

ImqBoolean setHeartBeatInterval(const MQLONG *intervalle* = 300L) ;
Définit l' **intervalle des signaux de présence**. Cette méthode renvoie TRUE en cas de succès.

Intervalle keepAliveMQLONG () const ;
Renvoie l' **intervalle de signal de présence**.

ImqBoolean setKeepAliveInterval(const MQLONG *intervalle* = MQKAI_AUTO) ;
Définit l' **intervalle de signal de présence**. Cette méthode renvoie TRUE en cas de succès.

ImqString localAddress() const ;
Renvoie l' **adresse locale**.

ImqBoolean setLocalAdresse (const char * *adresse* = 0) ;
Définit l' **adresse locale**. Cette méthode renvoie TRUE en cas de succès.

MQLONG maximumMessageLongueur () const ;
Renvoie la **longueur maximale des messages**.

ImqBoolean setMaximumMessageLength(const MQLONG *longueur* = 4194304L) ;
Définit la **longueur maximale des messages**. Cette méthode renvoie TRUE en cas de succès.

size_t messageCompressionNombre () const ;
Renvoie le nombre de techniques de compression de données de message prises en charge.

ImqBoolean messageCompression(nombre de tailles de const, MQLONG compress [ ]) const ;
Renvoie des copies des techniques de compression de données de message prises en charge dans **compress**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setMessageCompression (const size_t count, const MQLONG compress [ ]) ;

Définit les techniques de compression de données de message prises en charge à compresser.

Définit le comptage des techniques de compression de données de message prises en charge.

Cette méthode renvoie TRUE en cas de succès.

ImqString modeName() const ;
Renvoie le **nom de mode**.

ImqBoolean setModeNom (const char * *name* = 0) ;
Définit le **nom de mode**. Cette méthode renvoie TRUE en cas de succès.

Mot de passe ImqString () const ;
Renvoie le **mot de passe**.

ImqBoolean setPassword(const car * *password* = 0) ;
Définit le **mot de passe**. Cette méthode renvoie TRUE en cas de succès.

size_t receiveExitNombre () const ;
Renvoie le **nombre d'exits de réception**.

ImqString receiveExitNom () ;
Renvoie le premier des **noms d'exit de réception**, le cas échéant. Si le **nombre d'exits de réception** est égal à zéro, il renvoie une chaîne vide.

ImqBoolean receiveExitNoms (const size_t *count*, ImqString * *noms* [ ]) ;
Renvoie des copies des **noms d'exit de réception** dans *noms*. Définit tous les *noms* supérieurs à **nombre d'exits de réception** sur des chaînes nulles. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setReceiveExitName(const char * *name* = 0) ;
Définit les **noms d'exit de réception** sur le nom unique . *name* peut être vide ou null. Définit le **nombre d'exits de réception** sur 1 ou sur zéro. Efface la **réception des données utilisateur**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setReceiveExitNames(const size_t *count*, const char * *names* [ ]) ;
Définit les **noms d'exit de réception** sur *noms*. Les valeurs individuelles *names* ne doivent pas être vides ou nulles. Définit le **nombre d'exits de réception** sur *nombre*. Efface la **réception des données utilisateur**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setReceiveExitNames(const size_t *count*, const ImqString * *names* [ ]) ;
Définit les **noms d'exit de réception** sur *noms*. Les valeurs individuelles *names* ne doivent pas être vides ou nulles. Définit le **nombre d'exits de réception** sur *nombre*. Efface la **réception des données utilisateur**. Cette méthode renvoie TRUE en cas de succès.

ImqString receiveUserData () ;
Renvoie le premier des éléments **receive user data** , le cas échéant. Si le **nombre d'exits de réception** est égal à zéro, renvoie une chaîne vide.

ImqBoolean receiveUserDonnées (const size_t *count*, ImqString * *données* [ ]) ;
Renvoie des copies des éléments **receive user data** dans *data*. Définit une valeur de *données* supérieure à **nombre de sorties de réception** sur des chaînes nulles. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setReceiveUserData(const char * *data* = 0) ;
Définit les **données utilisateur de réception** sur l'élément unique *données*. Si *data* n'est pas null, le **nombre d'exits de réception** doit être au moins égal à 1. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setReceiveUserData(const size_t *count*, const char * *données* [ ]) ;
Définit **receive user data** sur *data*. *nombre* ne doit pas être supérieur au **nombre d'exits de réception**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setReceiveUserData(const size_t *count*, const ImqString * *data* [ ]) ;
Définit **receive user data** sur *data*. *nombre* ne doit pas être supérieur au **nombre d'exits de réception**. Cette méthode renvoie TRUE en cas de succès.

ImqString securityExitNom () const ;
Renvoie le **nom de l'exit de sécurité**.

ImqBoolean setSecurityExitName(const char * *name* = 0) ;
Définit le **nom de l'exit de sécurité**. Cette méthode renvoie TRUE en cas de succès.

ImqString securityUserDonnées () const ;
Renvoie les **données utilisateur de sécurité**.

ImqBoolean setSecurityUserData(const char * *data* = 0) ;
Définit les **données utilisateur de sécurité**. Cette méthode renvoie TRUE en cas de succès.

size_t sendExitNombre () const ;
Renvoie le **nombre d'exits d'émission**.

ImqString sendExitNom () ;
Renvoie le premier des **noms d'exit d'émission**, le cas échéant. Renvoie une chaîne vide si le **nombre d'exits d'émission** est égal à zéro.

ImqBoolean sendExitNames (const size_t *count*, ImqString * *names* [ ]) ;
Renvoie des copies des **noms d'exit d'émission** dans *noms*. Définit tous les *noms* au-delà de **nombre d'exits d'émission** sur des chaînes nulles. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setSendExitName(const char * *name* = 0) ;
Définit les **noms d'exit d'émission** sur le seul *nom*. *name* peut être vide ou null. Définit le **nombre d'exits d'émission** sur 1 ou sur zéro. Efface les **données utilisateur d'envoi**. Cette méthode renvoie TRUE en cas de succès

ImqBoolean setSendExitNames(const size_t *count*, const char * *names* [ ]) ;
Définit les **noms d'exit d'émission** sur *noms*. Les valeurs individuelles *names* ne doivent pas être vides ou nulles. Définit le **nombre d'exits d'émission** sur *nombre*. Efface les **données utilisateur d'envoi**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setSendExitNames(const size_t *count*, const ImqString * *names* [ ]) ;
Définit les **noms d'exit d'émission** sur *noms*. Les valeurs individuelles *names* ne doivent pas être vides ou nulles. Définit le **nombre d'exits d'émission** sur *nombre*. Efface les **données utilisateur d'envoi**. Cette méthode renvoie TRUE en cas de succès.

ImqString sendUserData () ;
Renvoie le premier des **éléments d'envoi de données utilisateur** , le cas échéant. Renvoie une chaîne vide si le **nombre d'exits d'émission** est égal à zéro.

ImqBoolean sendUserDonnées (const size_t *count*, ImqString * *données* [ ]) ;
Renvoie des copies des éléments **send user data** dans *data*. Définit une valeur de *données* supérieure à **Nombre d'exits d'envoi** sur des chaînes nulles. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setSendUserData(const char * *data* = 0) ;
Définit **send user data** sur l'élément unique *data*. Si *data* n'est pas null, **nombre d'exits d'émission** doit être au moins égal à 1. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setSendUserData(const size_t *count*, const char * *données* [ ]) ;
Définit **send user data** sur *data*. *nombre* ne doit pas être supérieur au **nombre d'exits d'émission**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setSendUserData(const size_t *count*, const ImqString * *data* [ ]) ;
Définit **send user data** sur *data*. *nombre* ne doit pas être supérieur au **nombre d'exits d'émission**. Cette méthode renvoie TRUE en cas de succès.

Spécification ImqString sslCipher() const ;
Renvoie la spécification de chiffrement SSL.

ImqBoolean setSslCipherSpecification(const char * *nom* = 0) ;
Définit la spécification de chiffrement SSL. Cette méthode renvoie TRUE en cas de succès.

Authentification MQLONG sslClient() const ;
Renvoie le type d'authentification du client SSL.

ImqBoolean setSslClientAuthentication(const MQLONG *auth* = MQSCA_REQUIRED) ;
Définit le type d'authentification du client SSL. Cette méthode renvoie TRUE en cas de succès.

ImqString sslPeerNom () const ;
Renvoie le nom de l'homologue SSL.

ImqBoolean setSslPeerName(const char * *nom* = 0) ;
Définit le nom de l'homologue SSL. Cette méthode renvoie TRUE en cas de succès.

ImqString transactionProgramNom () const ;
Renvoie le **nom du programme de transaction**.

ImqBoolean setTransactionProgramName(const char * *name* = 0) ;
Définit le **nom du programme de transaction**. Cette méthode renvoie TRUE en cas de succès.

MQLONG transportType() const ;
Renvoie le **type de transport**.

ImqBoolean setTransportType (const MQLONG *type* = MQXPT_LU62) ;
Définit le **type de transport**. Cette méthode renvoie TRUE en cas de succès.

ImqString userId() const ;
Renvoie l' **ID utilisateur**.

ImqBoolean setUserId (const car * *id* = 0) ;
Définit l' **ID utilisateur**. Cette méthode renvoie TRUE en cas de succès.

Codes raison

- MQRC_DATA_LENGTH_ERROR
- MQRC_ITEM_COUNT_ERREUR
- POINT_NULL_MQRC_NULL
- MQRC_SOURCE_BUFFER_ERROR

Classe C++ d'en-tête ImqCICSBridge

Cette classe encapsule des fonctions spécifiques de la structure de données MQCIH.

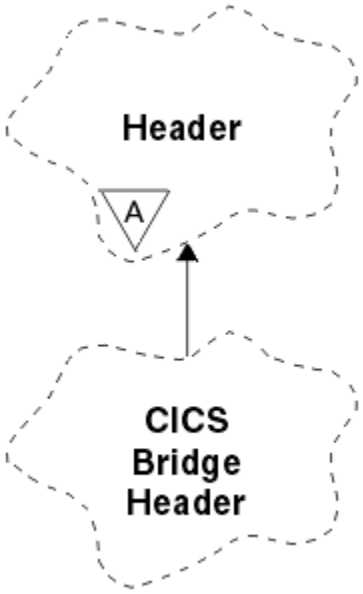


Figure 50. Classe d'en-tête ImqCICSBridge

Les objets de cette classe sont utilisés par les applications qui envoient des messages au pont CICS via WebSphere MQ for z/OS.

- «Attributs d'objet», à la page 1356
- «Constructeurs», à la page 1359
- «Méthodes ImqItem surchargées», à la page 1359
- «Méthodes d'objet (public)», à la page 1359
- «Données d'objet (protégées)», à la page 1361
- «Codes raison», à la page 1362
- «Codes retour», à la page 1362

Attributs d'objet

Descripteur ADS
Descripteur ADS d'envoi/réception. Cette valeur est définie à l'aide de MQCADSD_NONE. La valeur initiale est MQCADSD_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQCADSD_NONE

- MQCADSD_SEND
- MQCADSD_RECV
- MQCADSD_MSGFORMAT

identificateur d'attention
Touche AID. La longueur de la zone doit être MQ_ATTENTION_ID_LENGTH.

authentificateur
Mot de passe ou mot de passe RACF . La valeur initiale contient des blancs de longueur MQ_AUTHENTICATOR_LENGTH.

code de fin anormale de pont
Code de fin anormale du pont, de longueur MQ_ABEND_CODE_LENGTH. La valeur initiale est de quatre caractères blancs. La valeur renvoyée dans cette zone dépend du code retour. Pour plus d'informations, voir Tableau 614, à la page 1362.

code d'annulation de pont
Code de transaction de fin anormale de pont. La zone est réservée, doit contenir des blancs et doit avoir une longueur de MQ_CANCEL_CODE_LENGTH.

code achèvement de pont
Code achèvement, qui peut contenir le code achèvement WebSphere MQ ou la valeur CICS EIBRESP. La valeur initiale de la zone est MQCC_OK. La valeur renvoyée dans cette zone dépend du code retour. Pour plus d'informations, voir Tableau 614, à la page 1362.

décalage d'erreur de pont
Décalage d'erreur de pont. La valeur initiale est zéro. Cet attribut est en lecture seule.

code anomalie de pont
Code anomalie. Cette zone peut contenir la raison WebSphere MQ ou la valeur CICS EIBRESP2 . La valeur initiale de la zone est MQRC_NONE. La valeur renvoyée dans cette zone dépend du code retour. Pour plus d'informations, voir Tableau 614, à la page 1362.

code retour de pont
Code retour du pont CICS . La valeur initiale est MQCRC_OK.

tâche conversationnelle
Indique si la tâche peut être conversationnelle. La valeur initiale est MQCCT_NO. Les valeurs supplémentaires suivantes sont possibles:

- MQCCT_OUI
- MQCCT_NO

Position de curseur
Position du curseur. La valeur initiale est zéro.

durée de conservation de la fonction
Heure d'édition de la fonction de pont CICS .

fonction telle que
Attribut de terminal émulé. La longueur de la zone doit être MQ_FACILITY_LIKE_LENGTH.

jeton de fonction
Valeur du jeton BVT. La longueur de la zone doit être MQ_FACILITY_LENGTH. La valeur initiale est MQCFAC_NONE.

Fonction
Fonction, qui peut contenir le nom d'appel WebSphere MQ ou la fonction CICS EIBFN. La zone a la valeur initiale MQCFUNC_NONE, avec la longueur MQ_FUNCTION_LENGTH. La valeur renvoyée dans cette zone dépend du code retour. Pour plus d'informations, voir Tableau 614, à la page 1362.

Les valeurs supplémentaires suivantes sont possibles lorsque la **fonction** contient un nom d'appel WebSphere MQ :

- MQCFUNC_MQCONN
- MQCFUNC_MQGET

- MQCFUNC_MQINQ
- MQCFUNC_NONE
- MQCFUNC_MQOPEN
- MQCFUNC_PUT
- MQCFUNC_MQPUT1

intervalle d'attente d'obtention
Intervalle d'attente d'un appel MQGET émis par la tâche de pont CICS . La valeur initiale est MQCGWI_DEFAULT. La zone s'applique uniquement lorsque **uow control** a la valeur MQCUOWC_FIRST. Les valeurs supplémentaires suivantes sont possibles:

- MQCGWI_VALEUR PAR DEFAUT
- MQWI_ILLIMITÉ

Type de lien
Type de lien. La valeur initiale est MQCLT_PROGRAM. Les valeurs supplémentaires suivantes sont possibles:

- PROGRAMME MQCL
- TRANSACTION MQCL

ID transaction suivante
ID de la transaction suivante à joindre. La longueur de la zone doit être MQ_TRANSACTION_ID_LENGTH.

longueur des données de sortie
Longueur des données COMMAREA. La valeur initiale est MQCODL_AS_INPUT.

format de réponse
Nom de format du message de réponse. La valeur initiale est MQFMT_NONE avec la longueur MQ_FORMAT_LENGTH.

code de début
Code de début de transaction. La longueur de la zone doit être MQ_START_CODE_LENGTH. La valeur initiale est MQCSC_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQCSC_DEBUT
- Données MQCSC_STARTDATA
- PUT MQCSC_TERMINE
- MQCSC_NONE

statut de fin de tâche
Statut de fin de la tâche. La valeur initiale est MQCTES_NOSYNC. Les valeurs supplémentaires suivantes sont possibles:

- MQCTES_COMMIT
- MQCTES_BACKOUT
- MQCTES_ENDTASK
- MQCTES_NOSYNC

Identifiant de transaction
ID de la transaction à associer. La valeur initiale doit contenir des blancs et doit avoir une longueur de MQ_TRANSACTION_ID_LENGTH. La zone s'applique uniquement lorsque le **contrôle d'unité de travail** a la valeur MQCUOWC_FIRST ou MQCUOWC_ONLY.

Contrôle UOW
Contrôle de l'unité de travail. La valeur initiale est MQCUOWC_ONLY. Les valeurs supplémentaires suivantes sont possibles:

- MQCUOWC_FIRST
- MQCUOWC_MILIEU

- MQCUOWC_LAST
- MQCUOWC_ONLY
- MQCUOWC_COMMIT
- MQCUOWC_BACKOUT
- MQCUOWC_CONTINUE

version
Numéro de version MQCIH. La valeur initiale est MQCIH_VERSION_2. La seule autre valeur prise en charge est MQCIH_VERSION_1.

Constructeurs

ImqCICSBridgeHeader () ;
Constructeur par défaut.

ImqCICSBridgeHeader (const ImqCICSBridgeHeader & *header*) ;
Constructeur de copie.

Méthodes ImqItem surchargées

virtual ImqBoolean copyOut(ImqMessage & *msg*) ;
Insère une structure de données MQCIH dans la mémoire tampon de message au début, en déplaçant les données de message existantes plus loin et en définissant le format de message sur MQFMT_CICS.

Pour plus de détails, voir la description de la méthode de classe parent.

virtual ImqBoolean pasteIn(ImqMessage & *msg*) ;
Lit une structure de données MQCIH à partir de la mémoire tampon de messages. Pour que l'opération aboutisse, le codage de l'objet *msg* doit être MQENC_NATIVE. Extrayez les messages avec MQGMO_CONVERT en MQENC_NATIVE. Pour que l'opération aboutisse, le format ImqMessage doit être MQFMT_CICS.

Pour plus de détails, voir la description de la méthode de classe parent.

Méthodes d'objet (public)

void operator = (const ImqCICSBridgeHeader & *header*) ;
Copie les données d'instance à partir de l'en-tête , en remplaçant les données d'instance existantes.

MQLONG ADSDescripteur () const ;
Renvoie une copie du **descripteur ADS**.

void setADSDescriptor(const MQLONG *descripteur* = MQCADSD_NONE) ;
Définit le **descripteur ADS**.

ImqString attentionIdentifier() const ;
Renvoie une copie de l' **identificateur d'attention**, complétée avec des blancs de fin jusqu'à la longueur MQ_ATTENTION_ID_LENGTH.

void setAttentionIdentifier (const char * *données* = 0) ;
Définit l' **identificateur d'attention**, complété par des blancs de fin sur la longueur MQ_ATTENTION_ID_LENGTH. Si aucune *donnée* n'est fournie, réinitialise l' **identificateur d'attention** à la valeur initiale.

ImqString authentificateur () const ;
Renvoie une copie de l' **authentificateur**, complétée avec des blancs de fin jusqu'à la longueur MQ_AUTHENTICATOR_LENGTH.

void setAuthenticator(const char * *données* = 0) ;
Définit l' **authentificateur**, complété par des blancs de fin de longueur MQ_AUTHENTICATOR_LENGTH. Si aucune *donnée* n'est fournie, réinitialise l' **authentificateur** à la valeur initiale.

ImqString bridgeAbendCode () const ;
Renvoie une copie du **code de fin anormale de pont**, complété par des blancs de fin jusqu'à la longueur MQ_ABEND_CODE_LENGTH.

ImqString bridgeCancelCode () const ;
Renvoie une copie du **code d'annulation du pont**, complété par des blancs de fin jusqu'à la longueur MQ_CANCEL_CODE_LENGTH.

void setBridgeCancelCode(const char * *data* = 0) ;
Définit le **code d'annulation de pont**, complété par des blancs de fin de longueur MQ_CANCEL_CODE_LENGTH. Si aucune *donnée* n'est fournie, réinitialise le **code d'annulation de pont** à la valeur initiale.

Code bridgeCompletionMQLONG () const ;
Renvoie une copie du **code achèvement de pont**.

MQLONG bridgeErrorDécalage () const ;
Renvoie une copie du **décalage d'erreur de pont**.

Code bridgeReasonMQLONG () const ;
Renvoie une copie du **code anomalie du pont**.

Code bridgeReturnMQLONG () const ;
Renvoie le **code retour de pont**.

MQLONG conversationalTask() const ;
Renvoie une copie de la **tâche conversationnelle**.

void setConversationalTask (const MQLONG *tâche* = MQCCT_NO) ;
Définit la **tâche conversationnelle**.

MQLONG cursorPosition() const ;
Renvoie une copie de la **position du curseur**.

void setCursorPosition (const MQLONG *position* = 0) ;
Définit la **position du curseur**.

Durée facilityKeepMQLONG () const ;
Renvoie une copie de la durée de conservation de la fonction .

void setFacilityKeepTime(const MQLONG *time* = 0) ;
Définit la durée de conservation de la fonction .

ImqString facilityLike() const ;
Renvoie une copie de la fonction **, telle que**, complétée avec des blancs de fin jusqu'à la longueur MQ_FACILITY_LIKE_LENGTH.

void setFacilityLike (const char * *nom* = 0) ;
Définit la fonction **comme**, complétée avec des blancs de fin sur la longueur MQ_FACILITY_LIKE_LENGTH. Si aucun *nom* n'est fourni, réinitialise la fonction **comme** la valeur initiale.

ImqBinary facilityToken() const ;
Renvoie une copie du **jeton de fonction**.

ImqBoolean setFacilityToken (const ImqBinary & *token*) ;
Définit le **jeton de fonction**. La **longueur des données** du *jeton* doit être égale à zéro ou MQ_FACILITY_LENGTH. Elle renvoie TRUE en cas de succès.

void setFacilityToken (const MQBYTE8 *jeton* = 0) ;
Définit le **jeton de fonction**. *token* peut être zéro, ce qui revient à spécifier MQCFAC_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ_FACILITY_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQCFAC_NONE, vous devrez peut-être effectuer un transtypage pour garantir une correspondance de signature. Par exemple, (MQBYTE *) MQCFAC_NONE.

Fonction ImqString () const ;
Renvoie une copie de la **fonction**, complétée avec des blancs de fin jusqu'à la longueur MQ_FUNCTION_LENGTH.

Intervalle getWaitMQLONG () const ;
Renvoie une copie de l' **intervalle d'attente d'obtention**.

void setGetWaitInterval(const MQLONG *intervalle* = MQCGWI_DEFA
Définit l' **intervalle d'attente d'obtention**.

MQLONG linkType() const ;
Renvoie une copie du **type de lien**.

void setLinkType (const MQLONG *type* = MQCLT_PROGRAM) ;
Définit le **type de lien**.

ImqString nextTransactionIdentificateur () const ;
Renvoie une copie des données de l' **identificateur de transaction suivant** , complétée avec des blancs de fin jusqu'à la longueur MQ_TRANSACTION_ID_LENGTH.

MQLONG outputDataLongueur () const ;
Renvoie une copie de la **longueur des données de sortie**.

void setOutputDataLength(const MQLONG *longueur* = MQCODL_AS_INPUT) ;
Définit la **longueur des données de sortie**.

ImqString replyToFormat () const ;
Renvoie une copie du nom **reply-to format** , complétée avec des blancs de fin jusqu'à la longueur MQ_FORMAT_LENGTH.

void setReplyToFormat(const char * *name* = 0) ;
Définit le **format de réponse**, complété avec des blancs de fin sur la longueur MQ_FORMAT_LENGTH. Si aucun *nom* n'est fourni, réinitialise le **format de réponse** à la valeur initiale.

ImqString startCode() const ;
Renvoie une copie du **code de début**, complétée avec des blancs de fin jusqu'à la longueur MQ_START_CODE_LENGTH.

void setStartCode (const char * *données* = 0) ;
Définit les données du **code de démarrage** , complétées par des blancs de fin sur la longueur MQ_START_CODE_LENGTH. Si aucune *donnée* n'est fournie, réinitialise le **code de début** à la valeur initiale.

MQLONG taskEndStatut () const ;
Renvoie une copie du **statut de fin de tâche**.

ImqString transactionIdentifier() const ;
Renvoie une copie des données de l' **identificateur de transaction** , complétée avec des blancs de fin jusqu'à la longueur MQ_TRANSACTION_ID_LENGTH.

void setTransactionIdentifier (const char * *data* = 0) ;
Définit l' **identificateur de transaction**, rempli avec des blancs de fin sur la longueur MQ_TRANSACTION_ID_LENGTH. Si aucune *donnée* n'est fournie, réinitialise l' **identificateur de transaction** à la valeur initiale.

MQLONG UOWControl () const ;
Renvoie une copie du **contrôle d'unité de travail**.

void setUOWControl(const MQLONG *contrôle* = MQCUOWC_ONLY) ;
Définit le **contrôle d'unité de travail**.

Version MQLONG () const ;
Renvoie le numéro de **version** .

ImqBoolean setVersion(const MQLONG *version* = MQCIH_VERSION_2) ;
Définit le numéro de **version** . Elle renvoie TRUE en cas de succès.

Données d'objet (protégées)

MQLONG *olVersion*
Numéro de version MQCIH maximal pouvant être utilisé dans le stockage alloué pour *opcih*.

PMQCIH *opcih*
Adresse d'une structure de données MQCIH. La quantité de mémoire allouée est indiquée par *olVersion*.

Codes raison

- MQRC_BINARY_DATA_LENGTH_ERROR
- VERSION MQRC_WRONG_VERSION

Codes retour

Tableau 614. Codes retour de la classe d'en-tête ImqCICSBridge

Code retour	Function	CompCode	Motif	Code de fin anormale
MQCRC_OK				
MQCRC_BRIDGE_ERREUR			MQFB_CICS	
MQCRC_MQ_API_ERREUR	Nom d'appel WebSphere MQ	WebSphere MQ CompCode	WebSphere MQ Raison	
MQCRC_DÉLAI_BRIDGE_ATTENTE	Nom d'appel WebSphere MQ	WebSphere MQ CompCode	WebSphere MQ Raison	
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_NON DISPONIBLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_BRIDGE_ABEND				CICS -ABCODE
MQCRC_APPLICATION_ABEND				CICS -ABCODE

Classe C++ ImqDeadLetterHeader

Cette classe encapsule les fonctions de la structure de données MQDLH.

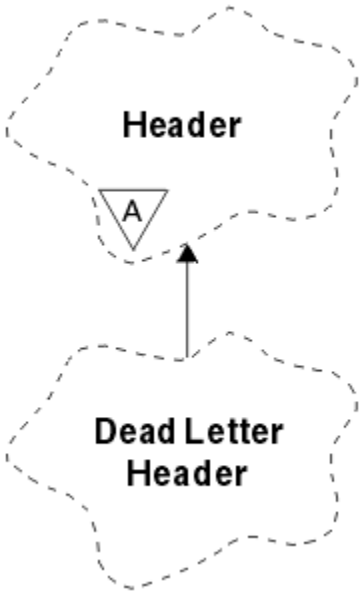


Figure 51. Classe ImqDeadLetterHeader

Les objets de cette classe sont généralement utilisés par une application qui rencontre un message qui ne peut pas être traité. Un nouveau message comprenant un en-tête de rebut et le contenu du message est placé dans la file d'attente de rebut et le message est supprimé.

- «Attributs d'objet», à la page 1363
- «Constructeurs», à la page 1364
- «Méthodes ImqItem surchargées», à la page 1364
- «Méthodes d'objet (public)», à la page 1364
- «Données d'objet (protégées)», à la page 1365
- «Codes raison», à la page 1365

Attributs d'objet

code raison de rebut
Raison pour laquelle le message est arrivé dans la file d'attente des messages non livrés. La valeur initiale est MQRC_NONE.

Nom du gestionnaire de files d'attente de destination
Nom du gestionnaire de files d'attente de destination d'origine. Le nom est une chaîne de longueur MQ_Q_MGR_NAME_LENGTH. Sa valeur initiale est null.

nom de la file d'attente de destination
Nom de la file d'attente de destination d'origine. Le nom est une chaîne de longueur MQ_Q_NAME_LENGTH. Sa valeur initiale est null.

Nom d'application d'insertion
Cet attribut indique le nom de l'application ayant inséré le message dans la file d'attente de rebut. Le nom est une chaîne de longueur MQ_PUT_APPL_NAME_LENGTH. Sa valeur initiale est null.

Type d'application d'insertion
Type d'application qui a inséré le message dans la file d'attente de rebut. La valeur initiale est zéro.

Date d'insertion
Date à laquelle le message a été inséré dans la file d'attente de rebut. La date est une chaîne de longueur MQ_PUT_DATE_LENGTH. Sa valeur initiale est une chaîne nulle.

Heure d'insertion
Heure à laquelle le message a été inséré dans la file d'attente de rebut. L'heure est une chaîne de longueur MQ_PUT_TIME_LENGTH. Sa valeur initiale est une chaîne nulle.

Constructeurs

ImqDeadLetterHeader( );
Constructeur par défaut.

ImqDeadLetterHeader(const ImqDeadLetterHeader & *header*) ;
Constructeur de copie.

Méthodes ImqItem surchargées

virtual ImqBoolean copyOut(ImqMessage & *msg*) ;
Insère une structure de données MQDLH dans la mémoire tampon du message au début, ce qui déplace les données de message existantes plus loin. Définit le *msg* **format** sur MQFMT_DEAD_LETTER_HEADER.

Voir la description de la méthode de classe ImqHeader sur la page «Classe C++ ImqHeader», à la page 1371 pour plus de détails.

virtuel ImqBoolean pasteIn(ImqMessage & *msg*) ;
Lit une structure de données MQDLH à partir de la mémoire tampon de messages.

Pour que l'opération aboutisse, le **format** ImqMessage doit être MQFMT_DEAD_LETTER_HEADER.

Voir la description de la méthode de classe ImqHeader sur la page «Classe C++ ImqHeader», à la page 1371 pour plus de détails.

Méthodes d'objet (public)

void operator = (const ImqDeadLetterHeader & *header*) ;
Les données d'instance de copie sont copiées à partir de l'en-tête , remplaçant les données d'instance existantes.

MQLONG deadLetterReasonCode() const ;
Renvoie le **code raison de rebut**.

void setDeadLetterReasonCode(const MQLONG *reason*) ;
Définit le **code raison de rebut**.

ImqString destinationQueueManagerName() const ;
Renvoie le **nom du gestionnaire de files d'attente de destination**, sans les blancs de fin.

void setDestinationQueueManagerNom(const char * *nom*) ;
Définit le **nom du gestionnaire de files d'attente de destination**. Tronque les données dont la longueur est supérieure à MQ_Q_MGR_NAME_LENGTH (48 caractères).

ImqString destinationQueueNom() const ;
Renvoie une copie du **nom de la file d'attente de destination**, sans les blancs de fin.

void setDestinationQueueName(const char * *name*) ;
Définit le **nom de la file d'attente de destination**. Tronque les données dont la longueur est supérieure à MQ_Q_NAME_LENGTH (48 caractères).

ImqString putApplicationNom() const ;
Renvoie une copie du **nom de l'application d'insertion**, sans les blancs de fin.

void setPutApplicationName(const char * *name* = 0) ;
Définit le **nom de l'application d'insertion**. Tronque les données dont la longueur est supérieure à MQ_PUT_APPL_NAME_LENGTH (28 caractères).

MQLONG putApplicationType() const ;
Renvoie le **type d'application d'insertion**.

void setPutApplicationType(const MQLONG *type* = MQAT_NO_CONTEXT) ;
Définit le **type d'application d'insertion**.

ImqString putDate() const ;
Renvoie une copie de la **date d'insertion**, sans les blancs de fin.

void setPutDate(const char * *date* = 0) ;
Définit la **date d'insertion**. Tronque les données dont la longueur est supérieure à MQ_PUT_DATE_LENGTH (8 caractères).

ImqString putTime() const ;
Renvoie une copie de l' **heure d'insertion**, débarrassée des blancs de fin.

void setPutTime(const char * *time* = 0) ;
Définit l' **heure d'insertion**. Tronque les données dont la longueur est supérieure à MQ_PUT_TIME_LENGTH (8 caractères).

Données d'objet (protégées)

MQDLH *omqdlh*
Structure de données MQDLH.

Codes raison

- MQRC_INCONSISTENT_FORMAT
- MQRC_STRUC_ID_ERROR
- ERREUR_CODAGE_MQRC_ERREUR

ImqDistributionRépertorie la classe C++

Cette classe encapsule une liste de distribution dynamique qui fait référence à une ou plusieurs files d'attente dans le but d'envoyer un ou plusieurs messages à plusieurs destinations.

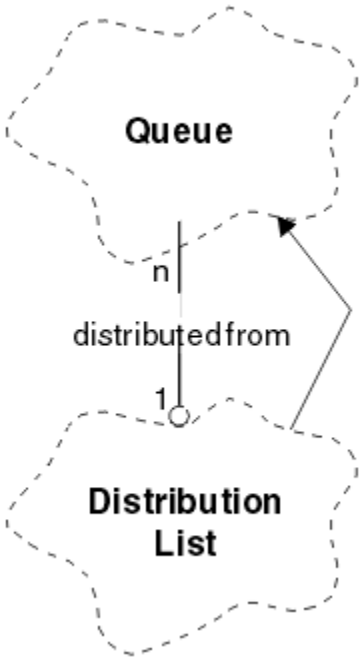


Figure 52. Classe de liste ImqDistribution

- «Attributs d'objet», à la page 1366
- «Constructeurs», à la page 1366
- «Méthodes d'objet (public)», à la page 1366
- «Méthodes d'objet (protégées)», à la page 1366

Attributs d'objet

première file d'attente répartie
Premier d'un ou de plusieurs objets de classe, sans ordre particulier, dans lequel la **référence de liste de distribution** traite cet objet.

Au départ, il n'y a pas d'objets de ce type. Pour que l'ouverture d'une liste ImqDistributionaboutisse, il doit y avoir au moins un objet de ce type.

Remarque : Lorsqu'un objet de liste ImqDistributionest ouvert, tous les objets ouverts qui y font référence sont automatiquement fermés.

Constructeurs

ImqDistributionList () ;
Constructeur par défaut.

ImqDistributionList (const ImqDistributionList & *list*) ;
Constructeur de copie.

Méthodes d'objet (public)

void operator = (const ImqDistributionList & *list*) ;
Tous les objets qui font référence à **cet objet** sont déréférencés avant la copie. Aucun objet ne fera référence à **cet** objet après l'appel de cette méthode.

*** firstDistributedQueue() const ;**
Renvoie la **première file d'attente répartie**.

Méthodes d'objet (protégées)

void setFirstDistributedQueue(* *file d'attente* = 0) ;
Définit la **première file d'attente répartie**.

Classe C++ ImqError

Cette classe abstraite fournit des informations sur les erreurs associées à un objet.

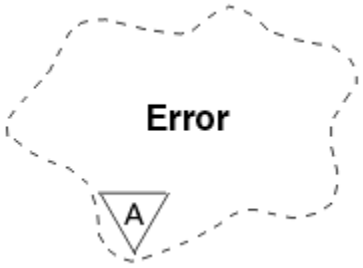


Figure 53. Classe ImqError

- «Attributs d'objet», à la page 1366
- «Constructeurs», à la page 1367
- «Méthodes d'objet (public)», à la page 1367
- «Méthodes d'objet (protégées)», à la page 1367
- «Codes raison», à la page 1367

Attributs d'objet

code achèvement
Code achèvement le plus récent. La valeur initiale est zéro. Les valeurs supplémentaires suivantes sont possibles:

- MQCC_OK
- MQCC_WARNING
- MQCC_FAILED

code anomalie
Code anomalie le plus récent. La valeur initiale est zéro.

Constructeurs

ImqError( );
Constructeur par défaut.

ImqError(const ImqError & ***error*** **) ;**
Constructeur de copie.

Méthodes d'objet (public)

void operator = (const ImqError & ***error*** **) ;**
Copie les données d'instance à partir de *erreur*, en remplaçant les données d'instance existantes.

void clearErrorCodes() ;
Définit le **code achèvement** et le **code anomalie** sur zéro.

MQLONG completionCode() const ;
Renvoie le **code achèvement**.

MQLONG reasonCode() const ;
Renvoie le **code anomalie**.

Méthodes d'objet (protégées)

ImqBoolean checkReadPointeur(const void * ***pointeur*****, const size_t** ***longueur*** **) ;**
Vérifie que la combinaison du pointeur et de la longueur est valide pour l'accès en lecture seule et renvoie TRUE si l'opération aboutit.

ImqBoolean checkWritePointer(const void * ***pointer*****, const size_t** ***longueur*** **) ;**
Vérifie que la combinaison du pointeur et de la longueur est valide pour l'accès en lecture-écriture et renvoie TRUE en cas de réussite.

void setCompletionCode(const MQLONG ***code*** **= 0) ;**
Définit le **code achèvement**.

void setReasonCode(const MQLONG ***code*** **= 0) ;**
Définit le **code anomalie**.

Codes raison

- MQRC_BUFFER_ERROR

Classe C++ ImqGetMessageOptions

Cette classe encapsule la structure de données MQGMO

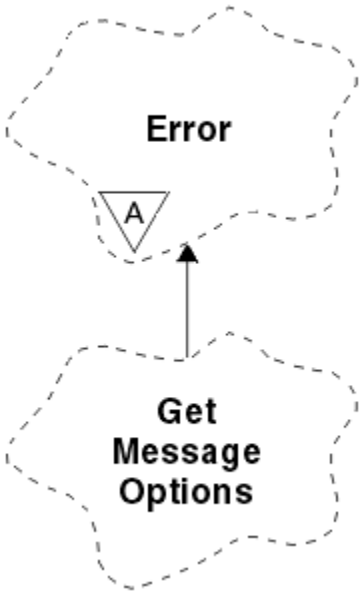


Figure 54. Classe ImqGetMessageOptions

Attributs d'objet

statut de groupe
Statut d'un message pour un groupe de messages. La valeur initiale est MQGS_NOT_IN_GROUP. Les valeurs supplémentaires suivantes sont possibles:

- MQGS_MSG_IN_GROUPE
- MQGS_LAST_MSG_IN_GROUP

options de mise en correspondance
Options de sélection des messages entrants. La valeur initiale est MQMO_MATCH_MSG_ID | MQMO_MATCH_CORREL_ID. Les valeurs supplémentaires suivantes sont possibles:

- ID_GROUPE_MQMO
- NUMERO MQMO_MATCH_MSG_SEQ_NO
- MQMO_MATCH_OFFSET
- JETON_MSG_MQMO_TOKEN
- MQMO_AUCUN

jeton de message
Jeton de message. Une valeur binaire (MQBYTE16) de longueur MQ_MSG_TOKEN_LENGTH. La valeur initiale est MQMTOK_NONE.

Options
Options applicables à un message. La valeur initiale est MQGMO_NO_WAIT. Les valeurs supplémentaires suivantes sont possibles:

- MQGMO_WAIT
- MQGMO_SYNCPOINT

- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_NO_SYNCPOINT
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_LOCK
- MQGMO_UNLOCK
- MQGMO_ACCEPT_TRUNCATED_MSG
- MQGMO_SET_SIGNAL
- MQGMO_FAIL_IF_QUIESCING
- MQGMO_CONVERT
- MQGMO_LOGICAL_ORDER
- MQGMO_COMPLETE_MSG
- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_NONE

Nom de file d'attente résolu
Nom de file d'attente résolu. Cet attribut est en lecture seule. Les noms ne comportent jamais plus de 48 caractères et peuvent être complétés par des valeurs nulles. La valeur initiale est une chaîne nulle.

longueur renvoyée
Longueur renvoyée. La valeur initiale est MQRL_UNDEFINED. Cet attribut est en lecture seule.

Segmentation
Possibilité de segmenter un message. La valeur initiale est MQSEG_INHIBÉE. La valeur supplémentaire, MQSEG_ALLOWED, est possible.

statut du segment
Statut de segmentation d'un message. La valeur initiale est MQSS_NOT_A_SEGMENT. Les valeurs supplémentaires suivantes sont possibles:

- SEGMENT_MQSS_SEGMENT
- SEGMENT_LAST_MQSS_

participation au point de synchronisation
TRUE lorsque les messages sont extraits sous le contrôle de point de synchronisation.

Intervalle d'attente
Durée pendant laquelle la méthode **get** de la classe s'interrompt en attendant l'arrivée d'un message approprié, s'il n'en existe pas déjà un. La valeur initiale est zéro, ce qui entraîne une attente indéfinie. La valeur supplémentaire, MQWI_UNLIMITED, est possible. Cet attribut est ignoré sauf si les **options** incluent MQGMO_WAIT.

Constructeurs

ImqGetMessageOptions( );
Constructeur par défaut.

ImqGetMessageOptions(const ImqGetMessageOptions & ***gmo*** **) ;**
Constructeur de copie.

Méthodes d'objet (public)

void operator = (const ImqGetMessageOptions &*gmo*) ;
Copie les données d'instance à partir de *gmo*, en remplaçant les données d'instance existantes.

MQCHAR groupStatus() const ;
Renvoie le **statut du groupe**.

void setGroupStatus(const MQCHAR *status*) ;
Définit le **statut du groupe**.

MQLONG matchOptions() const ;
Renvoie les **options de correspondance**.

void setMatchOptions(const MQLONG *options*) ;
Définit les **options de correspondance**.

ImqBinary messageToken() const ;
Renvoie le **jeton de message**.

ImqBoolean setMessageToken (const ImqBinary &*jeton*) ;
Définit le **jeton de message**. La **longueur des données** du jeton doit être zéro ou MQ_MSG_TOKEN_LENGTH. Cette méthode renvoie TRUE en cas de succès.

void setMessageToken (const MQBYTE16 *jeton* = 0) ;
Définit le **jeton de message**. *token* peut être égal à zéro, ce qui revient à spécifier MQMTOK_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ_MSG_TOKEN_LENGTH des données binaires.

Lorsque vous utilisez des valeurs prédéfinies, telles que MQMTOK_NONE, vous n'avez peut-être pas besoin d'effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE *) MQMTOK_NONE.

MQLONG options() const ;
Renvoie les **options**.

void setOptions(const MQLONG *options*) ;
Définit les **options**, y compris la valeur **syncpoint participation** .

ImqString resolvedQueueNom() const ;
Renvoie une copie du **nom de file d'attente résolue**.

MQLONG returnedLength() const ;
Renvoie la **longueur renvoyée**.

MQCHAR segmentation() const ;
Renvoie la **segmentation**.

void setSegmentation(const MQCHAR *valeur*) ;
Définit la **segmentation**.

MQCHAR segmentStatus() Const ;
Renvoie le **statut du segment**.

void setSegmentStatus(const MQCHAR *status*) ;
Définit le **statut du segment**.

ImqBoolean syncPointParticipation() const ;
Renvoie la valeur **syncpoint participation** , qui est TRUE si les **options** incluent MQGMO_SYNCPOINT ou MQGMO_SYNCPOINT_IF_PERSISTENT.

void setSyncPointParticipation(const ImqBoolean *sync*) ;
Définit la valeur de **participation de point de synchronisation** . Si *sync* est TRUE, modifie les **options** pour inclure MQGMO_SYNCPOINT et exclure à la fois MQGMO_NO_SYNCPOINT et MQGMO_SYNCPOINT_IF_PERSISTENT. Si *sync* a pour valeur FALSE, modifie les **options** pour inclure MQGMO_NO_SYNCPOINT et exclure à la fois MQGMO_SYNCPOINT et MQGMO_SYNCPOINT_IF_PERSISTENT.

MQLONG waitInterval() const ;
Renvoie l' **intervalle d'attente**.

void setWaitInterval(const MQLONG *intervalle*) ;
Définit l' **intervalle d'attente**.

Méthodes d'objet (protégées)

static void setVersionSupported(const MQLONG) ;
Définit la version de **MQGMO** . La valeur par défaut est **MQGMO_VERSION_3**.

Données d'objet (protégées)

MQGMO *omqgmo*
Structure de données MQGMO version 2. Accès aux zones MQGMO prises en charge pour MQGMO_VERSION_2 uniquement.

PMQGMO *opgmo*
Adresse d'une structure de données MQGMO. Le numéro de version de cette adresse est indiqué dans *olVersion*. Vérifiez le numéro de version avant d'accéder aux zones MQGMO pour vous assurer qu'elles sont présentes.

MQLONG *olVersion*
Numéro de version de la structure de données MQGMO traitée par *opgmo*.

Codes raison

- MQRC_BINARY_DATA_LENGTH_ERROR

Classe C++ ImqHeader

Cette classe abstraite encapsule les fonctions communes de la structure de données MQDLH.

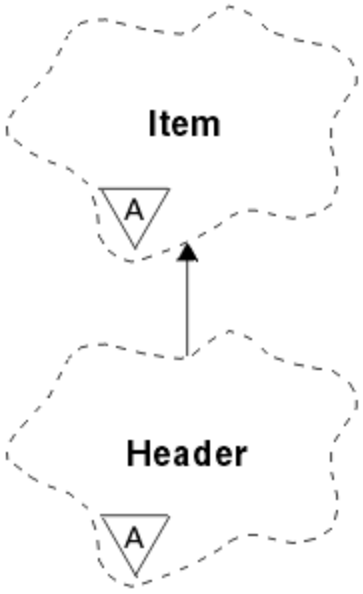


Figure 55. Classe ImqHeader

- «Attributs d'objet», à la page 1371
- «Constructeurs», à la page 1372
- «Méthodes d'objet (public)», à la page 1372

Attributs d'objet

jeu de caractères
Identificateur du jeu de caractères codés d'origine. Initialement MQCCSI_Q_MGR.

Codage
Codage d'origine. Initialement MQENC_NATIVE.

Format
Format d'origine. Initialement MQFMT_NONE.

indicateurs d'en-tête
Les valeurs initiales sont les suivantes:

- Zéro pour les objets de la classe ImqDeadLetterHeader
- MQIIH_NONE pour les objets de la classe d'en-tête ImqIMSBridge
- MQRMHF_LAST pour les objets de la classe d'en-tête ImqReference
- MQCIH_NONE pour les objets de la classe d'en-tête ImqCICSBridge
- MQWIH_NONE pour les objets de la classe d'en-tête ImqWork

Constructeurs

ImqHeader( );
Constructeur par défaut.

ImqHeader(const ImqHeader & ***header*** **) ;**
Constructeur de copie.

Méthodes d'objet (public)

void operator = (const ImqHeader & ***header*** **) ;**
Copie les données d'instance à partir de l'en-tête , en remplaçant les données d'instance existantes.

virtual MQLONG characterSet() const ;
Renvoie le **jeu de caractères**.

virtual void setCharacterSet(const MQLONG ***ccsid*** **= MQCCSI_Q_MGR) ;**
Définit le **jeu de caractères**.

codage() virtuel MQLONG const ;
Renvoie le **codage**.

vide virtuel setEncoding(const MQLONG ***codage*** **= MQENC_NATIVE) ;**
Définit le **codage**.

virtual ImqString format() const ;
Renvoie une copie du **format**, y compris les blancs de fin.

virtual void setFormat(const char * ***nom*** **= 0) ;**
Définit le **format**, complété par 8 caractères avec des blancs de fin.

virtual MQLONG headerFlags() const ;
Renvoie les **indicateurs d'en-tête**.

virtual void setHeaderFlags(const MQLONG ***indicateurs*** **= 0) ;**
Définit les **indicateurs d'en-tête**.

Classe C++ d'en-tête ImqIMSBridge

Cette classe encapsule les fonctions de la structure de données MQIIH.

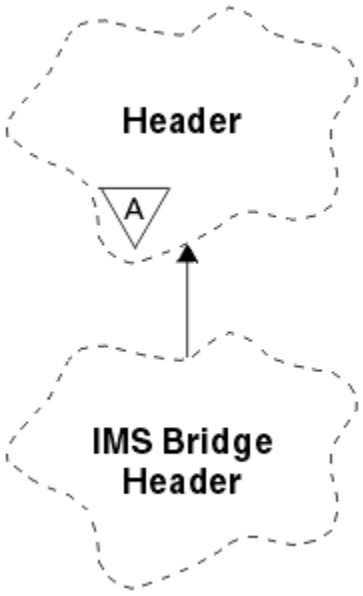


Figure 56. Classe d'en-tête ImqIMSBridge

Les objets de cette classe sont utilisés par les applications qui envoient des messages au pont IMS via WebSphere MQ for z/OS.

Remarque : Le jeu de caractères ImqHeader et le **codage** doivent avoir des valeurs par défaut et ne doivent pas être définis sur d'autres valeurs.

- «Attributs d'objet», à la page 1373
- «Constructeurs», à la page 1374
- «Méthodes ImqItem surchargées», à la page 1374
- «Méthodes d'objet (public)», à la page 1374
- «Données d'objet (protégées)», à la page 1375
- «Codes raison», à la page 1375

Attributs d'objet

authentificateur
Mot de passe ou mot de passe RACF, de longueur MQ_AUTHENTICATOR_LENGTH. La valeur initiale est MQIAUT_NONE.

Mode de validation
Mode de validation. Pour plus d'informations sur les modes de validation IMS , voir *OTMA-Guide d'utilisation* . La valeur initiale est MQICM_COMMIT_THEN_SEND. La valeur supplémentaire, MQICM_SEND_THEN_COMMIT, est possible.

Substitution de terminal logique
Remplacement de terminal logique, de longueur MQ_LTERM_OVERRIDE_LENGTH. La valeur initiale est une chaîne nulle.

Nom de mappe des services de structuration de messages
Nom de mappe MFS, de longueur MQ_MFS_MAP_NAME_LENGTH. La valeur initiale est une chaîne nulle.

format de réponse
Format de toute réponse, de longueur MQ_FORMAT_LENGTH. La valeur initiale est MQFMT_NONE.

Etendue de la sécurité
Portée du traitement de la sécurité IMS . La valeur initiale est MQISS_CHECK. La valeur supplémentaire, MQISS_FULL, est possible.

ID instance de transaction
Identité de l'instance de transaction, valeur binaire (MQBYTE16) de longueur MQ_TRAN_INSTANCE_ID_LENGTH. La valeur initiale est MQITII_NONE.

ETAT DE TRANSACTION
Etat de la conversation IMS . La valeur initiale est MQITS_NOT_IN_CONVERSATION. La valeur supplémentaire, MQITS_IN_CONVERSATION, est possible.

Constructeurs

ImqIMSBridgeHeader () ;
Constructeur par défaut.

ImqIMSBridgeHeader (const ImqIMSBridgeHeader & *header*) ;
Constructeur de copie.

Méthodes ImqItem surchargées

virtual ImqBoolean copyOut(ImqMessage & *msg*) ;
Insère une structure de données MQIIH dans la mémoire tampon de message au début, déplaçant les données de message existantes plus loin. Définit le *msg* **format** sur MQFMT_IMS.

Pour plus de détails, voir la description de la méthode de la classe parent.

virtuel ImqBoolean pasteIn(ImqMessage & *msg*) ;
Lit une structure de données MQIIH à partir de la mémoire tampon de messages.

Pour que l'opération aboutisse, le **codage** de l'objet *msg* doit être MQENC_NATIVE. Extrayez les messages avec MQGMO_CONVERT en MQENC_NATIVE.

Pour que l'opération aboutisse, le **format** ImqMessage doit être MQFMT_IMS.

Pour plus de détails, voir la description de la méthode de la classe parent.

Méthodes d'objet (public)

void operator = (const ImqIMSBridgeHeader & *header*) ;
Copie les données d'instance à partir de l'en-tête , en remplaçant les données d'instance existantes.

ImqString authentificateur() const ;
Renvoie une copie de l' **authentificateur**, complétée avec des blancs de fin jusqu'à la longueur MQ_AUTHENTICATOR_LENGTH.

void setAuthenticator(const char * *nom*) ;
Définit l' **authentificateur**.

MQCHAR commitMode() const ;
Renvoie le **mode de validation**.

void setCommitMode(const MQCHAR *mode*) ;
Définit le **mode de validation**.

ImqString logicalTerminalRemplacement() const ;
Renvoie une copie de la **substitution de terminal logique**.

void setLogicalTerminalOverride(const char * *override*) ;
Définit la **substitution de terminal logique**.

ImqString messageFormatServicesMapNom() const ;
Renvoie une copie du **nom de mappe des services de format de message**.

void setMessageFormatServicesMapName(const char * *name*) ;
Définit le **nom de mappe des services de format de message**.

ImqString replyToFormat() const ;
Renvoie une copie du **format de réponse**, complétée avec des blancs de fin jusqu'à la longueur MQ_FORMAT_LENGTH.

void setReplyToFormat(const char * *format*) ;
Définit le **format de réponse**, complété avec des blancs de fin sur la longueur MQ_FORMAT_LENGTH.

MQCHAR securityScope() const ;
Renvoie la **portée de sécurité**.

void setSecurityScope(const MQCHAR *scope*) ;
Définit la **portée de sécurité**.

ImqBinary transactionInstance() const ;
Renvoie une copie de l' **ID instance de transaction**.

ImqBoolean setTransactionInstanceId(const ImqBinary & *id*) ;
Définit l' **ID d'instance de transaction**. La **longueur de données** du *jeton* doit être zéro ou MQ_TRAN_INSTANCE_ID_LENGTH. Cette méthode renvoie TRUE en cas de succès.

void setTransactionInstanceId(const MQBYTE16 *id* = 0) ;
Définit l' **ID d'instance de transaction**. *id* peut être égal à zéro, ce qui revient à spécifier MQITII_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ_TRAN_INSTANCE_ID_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQITII_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE *) MQITII_NONE.

MQCHAR transactionState() const ;
Renvoie l' **état de la transaction**.

void setTransactionState(const MQCHAR *state*) ;
Définit l' **état de la transaction**.

Données d'objet (protégées)

MQIIH *omqiih*
Structure de données MQIIH.

Codes raison

- MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_INCONSISTENT_FORMAT
- ERREUR_CODAGE_MQRC_ERREUR
- MQRC_STRUC_ID_ERROR

Classe C++ ImqItem

Cette classe abstraite représente un élément, peut-être un parmi plusieurs, dans un message.

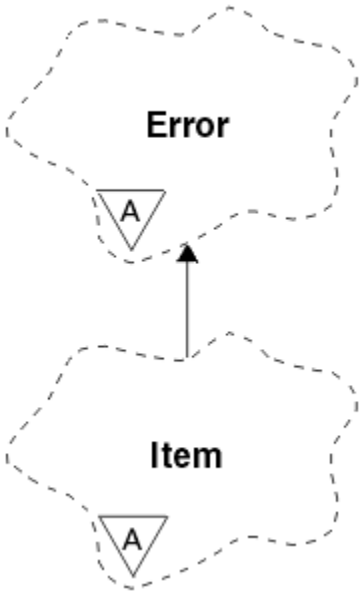


Figure 57. Classe ImqItem

Les éléments sont concaténés dans une mémoire tampon de messages. Chaque spécialisation est associée à une structure de données particulière qui commence par un ID de structure.

Les méthodes polymorphes de cette classe abstraite permettent de copier des éléments vers et depuis des messages. La classe ImqMessage **readItem** et les méthodes **writeItem** fournissent un autre style d'appel de ces méthodes polymorphes plus naturel pour les programmes d'application.

- «Attributs d'objet», à la page 1376
- «Constructeurs», à la page 1376
- «Méthodes de classe (public)», à la page 1376
- «Méthodes d'objet (public)», à la page 1377
- «Codes raison», à la page 1377

Attributs d'objet

ID structure
Chaîne de quatre caractères au début de la structure de données. Cet attribut est en lecture seule. Tenez compte de cet attribut pour les classes dérivées. Il n'est pas inclus automatiquement.

Constructeurs

ImqItem( );
Constructeur par défaut.

ImqItem(const ImqItem & ***item*** **) ;**
Constructeur de copie.

Méthodes de classe (public)

static ImqBoolean structureIdIs(const char * ***structure-id-to-test*****, const ImqMessage &** ***msg*** **) ;**
Renvoie TRUE si l' **ID de structure** de l'élément ImqItem suivant dans le *msg* entrant est identique à *structure-id-to-test*. L'élément suivant est identifié comme partie de la mémoire tampon de messages actuellement traitée par le **pointeur de données** ImqCache . Cette méthode s'appuie sur l' **ID de structure** et n'est donc pas garantie de fonctionner pour toutes les classes dérivées ImqItem .

Méthodes d'objet (public)

void operator = (const ImqItem & ***item*** **) ;**
Copie les données d'instance à partir de *élément*, en remplaçant les données d'instance existantes.

virtual ImqBoolean copyOut(ImqMessage & ***msg*** **) = 0 ;**
Ecrit cet objet en tant qu'élément suivant dans une mémoire tampon de messages sortants, en l'ajoutant à tous les éléments existants. Si l'opération d'écriture aboutit, augmente la **longueur de données** ImqCache . Cette méthode renvoie TRUE en cas de succès.

Remplacez cette méthode pour utiliser une sous-classe spécifique.

virtuel ImqBoolean pasteIn(ImqMessage & ***msg*** **) = 0 ;**
Lit cet objet *de façon destructive* à partir de la mémoire tampon des messages entrants. La lecture est destructive dans la mesure où le **pointeur de données** ImqCache est déplacé. Toutefois, le contenu de la mémoire tampon reste le même, de sorte que les données peuvent être relectées en réinitialisant le **pointeur de données** ImqCache .

La (sous-) classe de cet objet doit être cohérente avec l' **ID de structure** qui se trouve ensuite dans la mémoire tampon de messages de l'objet *msg* .

Le **codage** de l'objet *msg* doit être MQENC_NATIVE. Il est recommandé d'extraire les messages avec l'option ImqMessage **encoding** définie sur MQENC_NATIVE et avec les options ImqGetMessageOptions incluant MQGMO_CONVERT.

Si l'opération de lecture aboutit, la **longueur de données** ImqCache est réduite. Cette méthode renvoie TRUE en cas de succès.

Remplacez cette méthode pour utiliser une sous-classe spécifique.

Codes raison

- ERREUR_CODAGE_MQRC_ERREUR
- MQRC_STRUC_ID_ERROR
- MQRC_INCONSISTENT_FORMAT
- MQRC_INSUFFISENT_BUFFER
- MQRC_DONNÉES_INSUFFISANTES

Classe C++ ImqMessage

Cette classe encapsule une structure de données MQMD et gère également la construction et la reconstruction des données de message.

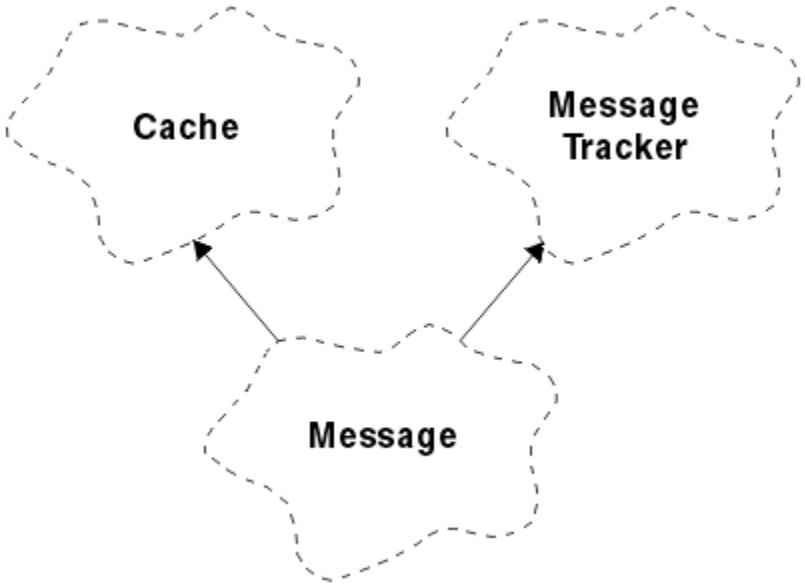


Figure 58. Classe ImqMessage

- «Attributs d'objet», à la page 1378
- «Constructeurs», à la page 1382
- «Méthodes d'objet (public)», à la page 1382
- «Méthodes d'objet (protégées)», à la page 1384
- «Données d'objet (protégées)», à la page 1384

Attributs d'objet

Données d'ID application
Informations d'identité associées à un message. La valeur initiale est une chaîne nulle.

Données d'origine de l'application
Informations d'origine associées à un message. La valeur initiale est une chaîne nulle.

Nombre d'annulations
Nombre de fois où un message a été provisoirement extrait, puis annulé. La valeur initiale est zéro. Cet attribut est en lecture seule.

jeu de caractères
ID de jeu de caractères codés. La valeur initiale est MQCCSI_Q_MGR. Les valeurs supplémentaires suivantes sont possibles:

- MQCCSI_HÉRITER
- MQCCSI_IMBRIQUÉ

Vous pouvez également utiliser un ID de jeu de caractères codés de votre choix. Pour plus d'informations à ce sujet, voir «Conversion de page de code», à la page 930.

Codage
Codage machine des données de message. La valeur initiale est MQENC_NATIVE.

expiration
Quantité avec contrainte horaire qui contrôle la durée pendant laquelle WebSphere MQ conserve un message non extrait avant de le supprimer. La valeur initiale est MQEI_UNLIMITED.

Format
Nom du format (modèle) qui décrit la présentation des données dans la mémoire tampon. Les noms de plus de huit caractères sont tronqués à huit caractères. Les noms sont toujours complétés avec des blancs à huit caractères. La valeur de la constante initiale est MQFMT_NONE. Les constantes supplémentaires suivantes sont possibles:

- 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

Vous pouvez également utiliser une chaîne spécifique à l'application de votre choix. Pour plus d'informations à ce sujet, voir la zone «Format (MQCHAR8)», à la page 410 du descripteur de message (MQMD).

Indicateurs de message
Informations de contrôle de segmentation. La valeur initiale est MQMF_SEGMENTATION_INHIBÉE. Les valeurs supplémentaires suivantes sont possibles:

- MQMF_SEGMENTATION_ALLOWED
- GROUPE MQMF_MSG_IN_GROUP
- MQMF_LAST_MSG_IN_GROUP
- SEGMENT_MQMF
- SEGMENT_DERNIER_MQMF_SEGMENT
- MQMF_AUCUN

Type de message
Catégorisation générale d'un message. La valeur initiale est MQMT_DATAGRAM. Les valeurs supplémentaires suivantes sont possibles:

- MQMT_SYSTEM_FIRST
- MQMT_SYSTEM_LAST
- MQMT_DATAGRAM
- MQMT_REQUEST
- MQMT_REPLY
- MQMT_REPORT
- MQMT_APPL_FIRST
- MQMT_APPL_LAST

Vous pouvez également utiliser une valeur spécifique à l'application de votre choix. Pour plus d'informations à ce sujet, voir la zone «MsgType (MQLONG)», à la page 422 du descripteur de message (MQMD).

offset
Informations de décalage. La valeur initiale est zéro.

Longueur d'origine
Longueur d'origine d'un message segmenté. La valeur initiale est MQOL_UNDEFINED.

Persistance
Indique que le message est important et qu'il doit être sauvegardé à tout moment à l'aide du stockage de persistance. Cette option implique une baisse des performances. La valeur initiale est MQPER_PERSISTENCE_AS_Q_DEF. Les valeurs supplémentaires suivantes sont possibles:

- MQPER_PERSISTANT
- MQPER_NON_PERSISTENT

priority
Priorité relative pour la transmission et la distribution. Les messages de même priorité sont généralement distribués dans la même séquence qu'ils ont été fournis (bien qu'il existe plusieurs critères qui doivent être remplis pour garantir cela). La valeur initiale est MQPRI_PRIORITY_AS_Q_DEF.

Validation de propriété
Indique si la validation des propriétés doit avoir lieu lorsqu'une propriété du message est définie. La valeur initiale est **MQCMHO_DEFAULT_VALIDATION**. Les valeurs supplémentaires suivantes sont possibles:

- MQCMHO_VALIDATION
- MQCMHO_NO_VALIDATION

Les méthodes suivantes agissent sur la **validation de propriété**:

MQLONG propertyValidation() const ;
Renvoie l'option **property validation** .

void setPropertyValidation (const MQLONG *option*) ;
Définit l'option **property validation** .

Nom d'application d'insertion
Nom de l'application qui a inséré un message. La valeur initiale est une chaîne nulle.

Type d'application d'insertion
Type d'application qui a inséré un message. La valeur initiale est MQAT_NO_CONTEXT. Les valeurs supplémentaires suivantes sont possibles:

- MQAT_AIX
- MQAT_CICS
- MQAT_CICS_BRIDGE
- MQAT_DOS
- MQAT_IMS
- MQAT_IMS_BRIDGE
- MQAT_MVS
- MQAT_NOTES_AGENT
- MQAT_OS2
- MQAT_OS390
- MQAT_OS400
- MQAT_QMGR
- MQAT_UNIX
- MQAT_WINDOWS
- MQAT_WINDOWS_NT
- MQAT_XCF
- MQAT_PAR DEFAUT

- MQAT_INCONNU
- MQAT_USER_FIRST
- MQAT_USER_LAST

Vous pouvez également utiliser une chaîne spécifique à l'application de votre choix. Pour plus d'informations à ce sujet, voir la zone «Type PutAppl(MQLONG)», à la page 427 du descripteur de message (MQMD).

Date d'insertion
Date à laquelle un message a été inséré. La valeur initiale est une chaîne nulle.

Heure d'insertion
Heure à laquelle un message a été inséré. La valeur initiale est une chaîne nulle.

nom du gestionnaire de files d'attente de réponse
Nom du gestionnaire de files d'attente auquel toute réponse doit être envoyée. La valeur initiale est une chaîne nulle.

Nom de file d'attente de réponses
Nom de la file d'attente à laquelle toute réponse doit être envoyée. La valeur initiale est une chaîne nulle.

report
Informations de retour associées à un message. La valeur initiale est MQRO_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQRO_EXCEPTION
- MQRO_EXCEPTION_WITH_DATA
- MQRO_EXCEPTION_WITH_FULL_DATA *
- MQRO_EXPIRATION
- MQRO_EXPIRATION_WITH_DATA
- MQRO_EXPIRATION_AVEC_DONNÉES_COMPLET*
- 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

où * indique des valeurs qui ne sont pas prises en charge sur WebSphere MQ for z/OS.

numéro de séquence
Informations de séquence identifiant un message au sein d'un groupe. La valeur initiale est 1.

longueur totale des messages
Nombre d'octets disponibles lors de la dernière tentative de lecture d'un message. Ce nombre sera supérieur à la **longueur de message** ImqCache si le dernier message a été tronqué ou si le dernier message n'a pas été lu en raison d'une troncature. Cet attribut est en lecture seule. La valeur initiale est zéro.

Cet attribut peut être utile dans toute situation impliquant des messages tronqués.

ID utilisateur
Identité d'utilisateur associée à un message. La valeur initiale est une chaîne nulle.

Constructeurs

ImqMessage( );
Constructeur par défaut.

ImqMessage(const ImqMessage & ***msg*** **) ;**
Constructeur de copie. Voir la méthode **operator =** pour plus de détails.

Méthodes d'objet (public)

void operator = (const ImqMessage & ***msg*** **) ;**
Copie le MQMD et les données de message à partir de *msg*. Si une mémoire tampon a été fournie par l'utilisateur pour cet objet, la quantité de données copiées est limitée à la taille de mémoire tampon disponible. Sinon, le système s'assure qu'une mémoire tampon de taille adéquate est mise à disposition pour les données copiées.

ImqString applicationIdDonnées() const ;
Renvoie une copie des **données d'ID d'application**.

void setApplicationIdData(const char * ***data*** **= 0) ;**
Définit les **données d'ID application**.

ImqString applicationOriginDonnées() const ;
Renvoie une copie des **données d'origine de l'application**.

void setApplicationOriginData(const char * ***data*** **= 0) ;**
Définit les **données d'origine de l'application**.

MQLONG backoutCount() const ;
Renvoie le **nombre d'annulations**.

MQLONG characterSet() const ;
Renvoie le **jeu de caractères**.

void setCharacterSet(const MQLONG ***ccsid*** **= MQCCSI_Q_MGR) ;**
Définit le **jeu de caractères**.

MQLONG codage() const ;
Renvoie le **codage**.

void setEncoding(const MQLONG ***encoding*** **= MQENC_NATIVE) ;**
Définit le **codage**.

MQLONG expiration() const ;
Renvoie l' **expiration**.

void setExpiry(const MQLONG ***expiration*** **) ;**
Définit l' **expiration**.

ImqString format() const ;
Renvoie une copie du **format**, y compris les blancs de fin.

ImqBoolean formatIs(const char * ***format-to-test*** **) const ;**
Renvoie TRUE si le **format** est identique à *format-to-test*.

void setFormat(const char * ***name*** **= 0) ;**
Définit le **format**, complété par huit caractères avec des blancs de fin.

MQLONG messageFlags() const ;
Renvoie les **indicateurs de message**.

void setMessageFlags(const MQLONG ***indicateurs*** **) ;**
Définit les **indicateurs de message**.

MQLONG messageType() const ;
Renvoie le **type de message**.

void setMessageType(const MQLONG *type*) ;
Définit le **type de message**.

MQLONG décalage() const ;
Renvoie le **décalage**.

void setOffset(const MQLONG *décalage*) ;
Définit le **décalage**.

MQLONG originalLength() const ;
Renvoie la **longueur d'origine**.

void setOriginalLength(const MQLONG *longueur*) ;
Définit la **longueur d'origine**.

MQLONG persistance() const ;
Renvoie la **persistance**.

void setPersistence(const MQLONG *persistance*) ;
Définit la **persistance**.

MQLONG priorité() const ;
Renvoie la **priorité**.

void setPriority(const MQLONG *priorité*) ;
Définit la **priorité**.

ImqString putApplicationNom() const ;
Renvoie une copie du **nom de l'application d'insertion**.

void setPutApplicationName(const char * *name* = 0) ;
Définit le **nom de l'application d'insertion**.

MQLONG putApplicationType() const ;
Renvoie le **type d'application d'insertion**.

void setPutApplicationType(const MQLONG *type* = MQAT_NO_CONTEXT) ;
Définit le **type d'application d'insertion**.

ImqString putDate() const ;
Renvoie une copie de la **date d'insertion**.

void setPutDate(const char * *date* = 0) ;
Définit la **date d'insertion**.

ImqString putTime() const ;
Renvoie une copie de l' **heure d'insertion**.

void setPutTime(const char * *time* = 0) ;
Définit l' **heure d'insertion**.

ImqBoolean readItem(ImqItem & *élément*) ;
Lit dans l'objet *item* à partir de la mémoire tampon de messages, à l'aide de la méthode ImqItem **pasteIn** . Elle renvoie TRUE en cas de succès.

ImqString replyToQueueManagerNom() const ;
Renvoie une copie du **nom du gestionnaire de files d'attente de réponse**.

void setReplyToQueueManagerName(const char * *name* = 0) ;
Définit le **nom du gestionnaire de files d'attente de réponse**.

ImqString replyToQueueName() const ;
Renvoie une copie du **nom de la file d'attente de réponse**.

void setReplyToQueueNom(const char * *nom* = 0) ;
Définit le **nom de la file d'attente de réponse**.

MQLONG rapport() const ;
Renvoie le **rapport**.

void setReport(const MQLONG *rapport*) ;
Définit le **rapport**.

MQLONG sequenceNumber() const ;
Renvoie le **numéro de séquence**.

void setSequenceNumber(const MQLONG *nombre*) ;
Définit le **numéro de séquence**.

size_t totalMessageLongueur() const ;
Renvoie la **longueur totale des messages**.

ImqString userId() const ;
Renvoie une copie de l' **ID utilisateur**.

void setUserId(const char * *id* = 0) ;
Définit l' **ID utilisateur**.

ImqBoolean writeItem(ImqItem & *élément*) ;
Ecrit à partir de l'objet *item* dans la mémoire tampon du message, à l'aide de la méthode ImqItem **copyOut** . L'écriture peut prendre la forme d'une insertion, d'un remplacement ou d'un ajout: cela dépend de la classe de l'objet *item* . Cette méthode renvoie TRUE en cas de succès.

Méthodes d'objet (protégées)

static void setVersionSupported(const MQLONG) ;
Définit la **version MQMD**. La valeur par défaut est **MQMD_VERSION_2**.

Données d'objet (protégées)

MQMD1 *omqmd*
(WebSphere MQ for z/OS uniquement.) Structure de données MQMD.

MQMD2 *omqmd*
(Plateformes autres que z/OS.) Structure de données MQMD.

Classe C++ de la fonction de suivi ImqMessage

Cette classe encapsule les attributs d'un objet ImqMessage ou ImqQueue qui peuvent être associés à l'un ou l'autre objet.

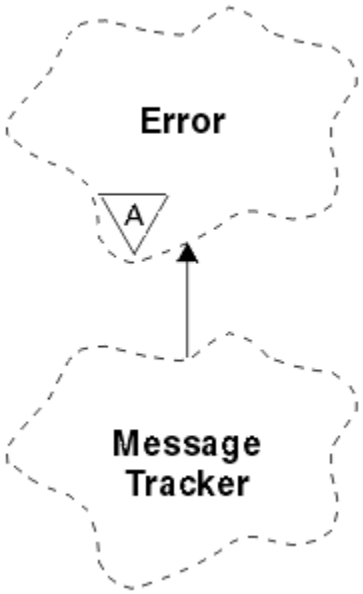


Figure 59. Classe ImqMessageTracker

Cette classe est liée aux appels MQI répertoriés dans le «Référence croisée de la fonction de suivi ImqMessage», à la page 1333.

Attributs d'objet

Jeton de comptabilité
Valeur binaire (MQBYTE32) de longueur MQ_ACCOUNTING_TOKEN_LENGTH. La valeur initiale est MQACT_NONE.

ID de corrélation
Une valeur binaire (MQBYTE24) de longueur MQ_CORREL_ID_LENGTH que vous affectez à la corrélation des messages. La valeur initiale est MQCI_NONE. La valeur supplémentaire, MQCI_NEW_SESSION, est possible.

Retour d'informations
Informations de retour à envoyer avec un message. La valeur initiale est MQFB_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQFB_SYSTEM_FIRST
- MQFB_SYSTEM_LAST
- MQFB_APPL_FIRST
- MQFB_APPL_LAST
- MQFB_COA
- MQFB_COD
- EXPIRATION_MQFB
- MQFB_PAN
- MQFB_NAN
- MQFB_QUIT
- MQFB_DATA_LENGTH_ZERO
- MQFB_DONNÉES_LONGUEUR_NÉGATIVE
- MQFB_DATA_LENGTH_TOO_BIG
- MQFB_BUFFER_OVERFLOW
- MQFB_LENGTH_OFF_BY_ONE
- MQFB_IIH_ERREUR
- MQFB_NOT_AUTHORIZED_FOR_IMS
- MQFB_IMS_ERREUR
- MQFB_IMS_FIRST
- MQFB_IMS_LAST
- MQFB_CICS_APPL_ABENDED
- MQFB_CICS_APPL_NOT_STARTED
- MQFB_CICS_BRIDGE_FAILURE
- MQFB_CICS_CCSID_ERREUR
- MQFB_CICS_CIH_ERREUR
- MQFB_CICS_COMMAREA_ERREUR
- MQFB_CICS_CORREL_ID_ERREUR

- MQFB_CICS_DLQ_ERREUR
- MQFB_CICS_ENCODING_ERROR
- MQFB_CICS_INTERNAL_ERROR
- MQFB_CICS_NOT_AUTHORIZED
- MQFB_CICS_UOW_BACKED_OUT
- MQFB_ERREUR UOW_CICS_

Vous pouvez également utiliser une chaîne spécifique à l'application de votre choix. Pour plus d'informations à ce sujet, voir la zone «Commentaires en retour (MQLONG)», à la page 406 du descripteur de message (MQMD).

ID groupe
Valeur binaire (MQBYTE24) de longueur MQ_GROUP_ID_LENGTH unique dans une file d'attente. La valeur initiale est MQGI_NONE.

ID message
Valeur binaire (MQBYTE24) de longueur MQ_MSG_ID_LENGTH unique dans une file d'attente. La valeur initiale est MQMI_NONE.

Constructeurs

ImqMessageTracker () ;
Constructeur par défaut.

ImqMessageTracker (const ImqMessageTracker & *tracker*) ;
Constructeur de copie. Voir la méthode **operator =** pour plus de détails.

Méthodes d'objet (public)

void operator = (const ImqMessageTracker & *tracker*) ;
Copie les données d'instance à partir de *tracker*, en remplaçant les données d'instance existantes.

ImqBinary accountingToken() const ;
Renvoie une copie du **jeton de comptabilité**.

ImqBoolean setAccountingToken(const ImqBinary & *jeton*) ;
Définit le **jeton de comptabilité**. La **longueur des données** du jeton doit être égale à zéro ou à MQ_ACCOUNTING_TOKEN_LENGTH. Cette méthode renvoie TRUE en cas de succès.

void setAccountingToken(const MQBYTE32 *token* = 0) ;
Définit le **jeton de comptabilité**. *token* peut être égal à zéro, ce qui revient à spécifier MQACT_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ_ACCOUNTING_TOKEN_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQACT_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature ; par exemple, (MQBYTE *) MQACT_NONE.

ImqBinary correlationId() const ;
Renvoie une copie de l' **ID corrélation**.

ImqBoolean setCorrelationId(const ImqBinary & *jeton*) ;
Définit l' **ID de corrélation**. La **longueur des données** du jeton doit être zéro ou MQ_CORREL_ID_LENGTH. Cette méthode renvoie TRUE en cas de succès.

void setCorrelationId(const MQBYTE24 *id* = 0) ;
Définit l' **ID de corrélation**. *id* peut être égal à zéro, ce qui revient à spécifier MQCI_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ_CORREL_ID_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQCI_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature ; par exemple, (MQBYTE *) MQCI_NONE.

commentaires en retourMQLONG () const ;
Renvoie les **commentaires en retour**.

void setFeedback(const MQLONG *feedback*) ;
Définit les **commentaires en retour**.

ImqBinary groupId() const ;
Renvoie une copie de l' **ID groupe**.

ImqBoolean setGroupId(const ImqBinary &*jeton*) ;
Définit l' **ID groupe**. La **longueur des données** du *jeton* doit être égale à zéro ou à MQ_GROUP_ID_LENGTH. Cette méthode renvoie TRUE en cas de succès.

void setGroupId(const MQBYTE24 *id* = 0) ;
Définit l' **ID groupe**. *id* peut être égal à zéro, ce qui revient à spécifier MQGI_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ_GROUP_ID_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQGI_NONE, vous devrez peut-être effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE *) MQGI_NONE.

ImqBinary messageId() const ;
Renvoie une copie de l' **ID message**.

ImqBoolean setMessageId(const ImqBinary &*jeton*) ;
Définit l' **ID message**. La **longueur de données** du *jeton* doit être zéro ou MQ_MSG_ID_LENGTH. Cette méthode renvoie TRUE en cas de succès.

void setMessageId(const MQBYTE24 *id* = 0) ;
Définit l' **ID message**. *id* peut être égal à zéro, ce qui revient à spécifier MQMI_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ_MSG_ID_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQMI_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE *) MQMI_NONE.

Codes raison

- MQRC_BINARY_DATA_LENGTH_ERROR

Classe C++ ImqNamelist

Cette classe encapsule une liste de noms.

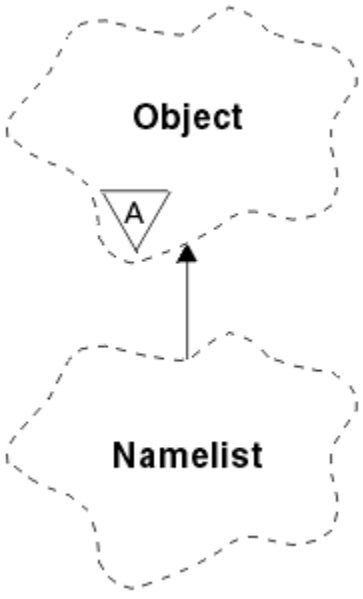


Figure 60. Classe ImqNamelist

Cette classe est liée aux appels MQI répertoriés dans le «Référence croisée ImqNamelist», à la page 1333.

- «Attributs d'objet», à la page 1388
- «Constructeurs», à la page 1388
- «Méthodes d'objet (public)», à la page 1388

- «Codes raison», à la page 1388

Attributs d'objet

Nombre de noms
Nombre de noms d'objet dans les **noms de liste de noms**. Cet attribut est en lecture seule.

noms de liste de noms
Noms d'objet, dont le nombre est indiqué par le **nombre de noms**. Cet attribut est en lecture seule.

Constructeurs

ImqNamelist( );
Constructeur par défaut.

ImqNamelist(const ImqNamelist & ***liste*** **) ;**
Constructeur de copie. Le **statut d'ouverture** ImqObject est false.

ImqNamelist(const car * ***nom*****) ;**
Définit le nom ImqObject sur **name**.

Méthodes d'objet (public)

void operator = (const ImqNamelist & ***liste*** **) ;**
Copie les données d'instance à partir de *liste*, en remplaçant les données d'instance existantes. Le **statut d'ouverture** ImqObject est false.

ImqBoolean nameCount(MQLONG & ***nombre*** **) ;**
Fournit une copie du **nombre de noms**. Elle renvoie TRUE en cas de succès.

MQLONG nameCount () ;
Renvoie le **nombre de noms** sans aucune indication d'erreurs possibles.

ImqBoolean namelistName (const MQLONG ***index*****, ImqString &** ***nom*** **) ;**
Fournit une copie de l'un des **noms de liste de noms** par index à base de zéro. Elle renvoie TRUE en cas de succès.

ImqString namelistName (const MQLONG ***index*** **) ;**
Renvoie l'un des **noms de liste de noms** par index à base de zéro sans aucune indication d'erreurs possibles.

Codes raison

- ERREUR MQRC_INDEX_ERREUR
- MQRC_INDEX_NOT_PRESENT

Classe C++ ImqObject

Cette classe est abstraite. Lorsqu'un objet de cette classe est détruit, il est automatiquement fermé et sa connexion au gestionnaire ImqQueueest interrompue.

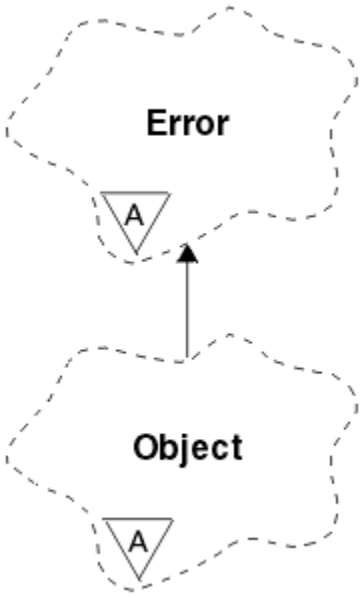


Figure 61. Classe ImqObject

Cette classe est liée aux appels MQI répertoriés dans le «Référence croisée ImqObject», à la page 1333.

Attributs de classe

comportement
Contrôle le comportement de l'ouverture implicite.

IMQ_IMPL_OPEN (8L)
L'ouverture implicite est autorisée. Il s'agit de l'option par défaut.

Attributs d'objet

Date de modification
Date de modification. Cet attribut est en lecture seule.

Heure de modification
Heure de modification. Cet attribut est en lecture seule.

Autre ID utilisateur
ID utilisateur de remplacement, jusqu'à MQ_USER_ID_LENGTH caractères. La valeur initiale est une chaîne nulle.

Autre ID de sécurité
ID de sécurité de remplacement. Une valeur binaire (MQBYTE40) de longueur MQ_SECURITY_ID_LENGTH. La valeur initiale est MQSID_NONE.

options de fermeture
Options qui s'appliquent lorsqu'un objet est fermé. La valeur initiale est MQCO_NONE. Cet attribut est ignoré lors des opérations de réouverture implicites, où la valeur MQCO_NONE est toujours utilisée.

référence de connexion
Référence à un objet gestionnaire ImqQueuequi fournit la connexion requise à un gestionnaire de files d'attente (local). Pour un objet gestionnaire ImqQueue, il s'agit de l'objet lui-même. La valeur initiale est zéro.

Remarque : Ne le confondez pas avec le **nom de gestionnaire de files d'attente** qui identifie un gestionnaire de files d'attente (éventuellement distant) pour une file d'attente nommée.

description
Nom descriptif (jusqu'à 64 caractères) du gestionnaire de files d'attente, de la file d'attente, de la liste de noms ou du processus. Cet attribut est en lecture seule.

name
Nom (jusqu'à 48 caractères) du gestionnaire de files d'attente, de la file d'attente, de la liste de noms ou du processus. La valeur initiale est une chaîne nulle. Le nom d'une file d'attente modèle change après une **ouverture** et devient le nom de la file d'attente dynamique résultante.

Remarque : Un gestionnaire ImqQueuepeut avoir un nom null, représentant le gestionnaire de files d'attente par défaut. Le nom change pour le gestionnaire de files d'attente réel après une **ouverture**réussie. Une liste ImqDistributionest dynamique et doit avoir un nom null.

objet géré suivant
Il s'agit de l'objet suivant de cette classe, sans ordre particulier, ayant la même **référence de connexion** que cet objet. La valeur initiale est zéro.

Options d'ouverture
Options qui s'appliquent lorsqu'un objet est ouvert. La valeur initiale est MQOO_INQUIRE. Il existe deux façons de définir les valeurs appropriées:

1. Ne définissez pas les **options d'ouverture** et n'utilisez pas la méthode **open** . WebSphere MQ ajuste automatiquement les **options d'ouverture** et ouvre, rouvre et ferme automatiquement les objets selon les besoins. Cela peut entraîner des opérations de réouverture inutiles, car WebSphere MQ utilise la méthode **openFor** et ajoute des **options d'ouverture** de manière incrémentielle uniquement.
2. Définissez les **options d'ouverture** avant d'utiliser les méthodes qui génèrent un appel MQI (voir «Références croisées C++ et MQI», à la page 1326). Cela permet de s'assurer que des opérations de réouverture inutiles ne se produisent pas. Définissez les options d'ouverture de manière explicite si l'un des problèmes de réouverture potentiels est susceptible de se produire (voir Rouvrir).

   Si vous utilisez la méthode **open** , vous ***devez*** vous assurer que les **options d'ouverture** sont appropriées en premier. Toutefois, l'utilisation de la méthode **open** n'est pas obligatoire ; WebSphere MQ présente toujours le même comportement que dans le cas 1, mais dans ce cas, le comportement est efficace.

Zéro n'est pas une valeur valide ; définissez la valeur appropriée avant de tenter d'ouvrir l'objet. Cette opération peut être effectuée à l'aide des **optionssetOpen**(*lOpenOptions*) suivi de **open**() ou **openFor**(*lRequiredOpenOption*).

Remarque :

1. MQOO_OUTPUT est remplacé par MQOO_INQUIRE lors de la méthode **open** pour une liste de distribution, car MQOO_OUTPUT est la seule **option d'ouverture** valide pour le moment. Toutefois, il est recommandé de toujours définir MQOO_OUTPUT explicitement dans les programmes d'application qui utilisent la méthode **open** .
2. Spécifiez MQOO_RESOLVE_NAMES si vous souhaitez utiliser les attributs **nom du gestionnaire de files d'attente résolu** et **nom de la file d'attente résolue** de la classe.

statut d'ouverture
Indique si l'objet est ouvert (TRUE) ou fermé (FALSE). La valeur initiale est FALSE. Cet attribut est en lecture seule.

objet géré précédent
Objet précédent de cette classe, sans ordre particulier, ayant la même **référence de connexion** que cet objet. La valeur initiale est zéro.

Identificateur du gestionnaire de files
Identificateur du gestionnaire de files d'attente. Cet attribut est en lecture seule.

Constructeurs

ImqObject( );
Constructeur par défaut.

ImqObject(const ImqObject & ***objet*** **) ;**
Constructeur de copie. Le **statut d'ouverture** est FALSE.

Méthodes de classe (public)

comportement MQLONG statique () ;
Renvoie le **comportement**.

void setBehavior(const MQLONG ***comportement*** **= 0) ;**
Définit le **comportement**.

Méthodes d'objet (public)

void operator = (const ImqObject & ***objet*** **) ;**
Effectue une opération de fermeture si nécessaire et copie les données d'instance à partir de l'objet . Le **statut d'ouverture** est FALSE.

ImqBoolean alterationDate(ImqString & ***date*** **) ;**
Fournit une copie de la **date de modification**. Elle renvoie TRUE en cas de succès.

ImqString alterationDate( );
Renvoie la **date d'altération** sans aucune indication d'erreurs possibles.

ImqBoolean alterationTime(ImqString & ***heure*** **) ;**
Fournit une copie de l' **heure d'altération**. Elle renvoie TRUE en cas de succès.

ImqString alterationTime( );
Renvoie l' **heure d'altération** sans aucune indication d'erreurs possibles.

ImqString alternateUserId() const ;
Renvoie une copie de l' **autre ID utilisateur**.

ImqBoolean setAlternateUserId(const char * ***id*** **) ;**
Définit l' **ID utilisateur de remplacement**. L' **autre ID utilisateur** ne peut être défini que si le **statut d'ouverture** est FALSE. Cette méthode renvoie TRUE en cas de succès.

ID ImqBinary alternateSecurity() const ;
Renvoie une copie de l'ID de sécurité de remplacement **.**

ImqBoolean setAlternateSecurityId(const ImqBinary & ***jeton*** **) ;**
Définit l' **autre ID de sécurité**. L' **autre ID de sécurité** ne peut être défini que si le **statut d'ouverture** est FALSE. La longueur des données du *jeton* doit être égale à zéro ou MQ_SECURITY_ID_LENGTH. Elle renvoie TRUE en cas de succès.

ImqBoolean setAlternateSecurityId(const MQBYTE* ***jeton*** **= 0) ;**
Définit l' **autre ID de sécurité**. *token* peut être égal à zéro, ce qui revient à spécifier MQSID_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ_SECURITY_ID_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQSID_NONE, vous pouvez être amené à effectuer un transtypage pour garantir la correspondance de signature ; par exemple, (MQBYTE *) MQSID_NONE.

L' **autre ID de sécurité** ne peut être défini que si le **statut d'ouverture** est TRUE. Elle renvoie TRUE en cas de succès.

ImqBoolean setAlternateSecurityId(const unsigned char * *id* = 0) ;
Définit l' **autre ID de sécurité**.

ImqBoolean close() ;
Définit le **statut d'ouverture** sur FALSE. Elle renvoie TRUE en cas de succès.

MQLONG closeOptions() const ;
Renvoie les **options de fermeture**.

void setCloseOptions(const MQLONG *options*) ;
Définit les **options de fermeture**.

ImqQueueManager * connectionReference() const ;
Renvoie la **référence de connexion**.

void setConnectionReference(ImqQueueManager & *manager*) ;
Définit la **référence de connexion**.

void setConnectionReference(ImqQueueManager * *manager* = 0) ;
Définit la **référence de connexion**.

virtuel ImqBoolean description(ImqString & *description*) = 0 ;
Fournit une copie de la **description**. Elle renvoie TRUE en cas de succès.

ImqString description() ;
Renvoie une copie de la **description** sans aucune indication d'erreurs possibles.

virtual ImqBoolean nom(ImqString & *nom*) ;
Fournit une copie du **nom**. Elle renvoie TRUE en cas de succès.

ImqString nom() ;
Renvoie une copie du **nom** sans aucune indication d'erreurs possibles.

ImqBoolean setName(const char * *name* = 0) ;
Définit le **nom**. Le **nom** peut être défini uniquement lorsque le **statut d'ouverture** est FALSE et, pour un gestionnaire ImqQueue, lorsque le **statut de connexion** est FALSE. Elle renvoie TRUE en cas de succès.

ImqObject * ObjetnextManaged() const ;
Renvoie l' **objet géré suivant**.

ImqBoolean open() ;
Remplace le **statut d'ouverture** par TRUE en ouvrant l'objet si nécessaire, à l'aide, entre autres, des **options d'ouverture** et du **nom**. Cette méthode utilise les informations de **référence de connexion** et la méthode ImqQueueManager **connect** si nécessaire pour s'assurer que le **statut de connexion** du gestionnaire ImqQueueest TRUE. Elle renvoie le **statut d'ouverture**.

ImqBoolean openFor(const MQLONG *obligatoire-options* = 0) ;
Tente de s'assurer que l'objet est ouvert avec des **options d'ouverture**ou avec des **options d'ouverture** qui garantissent le comportement impliqué par la valeur du paramètre *required-options* .

Si *required-options* est égal à zéro, l'entrée est requise et toute option d'entrée suffit. Par conséquent, si les **options d'ouverture** contiennent déjà l'une des options suivantes:

- MQOO_INPUT_AS_Q_DEF
- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIF

les **options d'ouverture** sont déjà satisfaisantes et ne sont pas modifiées ; si les **options d'ouverture** ne contiennent aucune de ces options, MQOO_INPUT_AS_Q_DEF est défini dans les **options d'ouverture**.

Si *required-options* est différent de zéro, les options requises sont ajoutées aux **options d'ouverture**; si *required-options* est l'une de ces options, les autres sont réinitialisées.

Si l'une des **options d'ouverture** est modifiée et que l'objet est déjà ouvert, il est fermé temporairement et rouvert afin d'ajuster les **options d'ouverture**.

Elle renvoie TRUE en cas de succès. La réussite indique que l'objet est ouvert avec les options appropriées.

MQLONG openOptions() const ;
Renvoie les **options d'ouverture**.

ImqBoolean setOpenOptions(const MQLONG *options*) ;
Définit les **options d'ouverture**. Les **options d'ouverture** ne peuvent être définies que si le **statut d'ouverture** est FALSE. Elle renvoie TRUE en cas de succès.

ImqBoolean openStatus() const ;
Renvoie le **statut d'ouverture**.

ImqObject * ObjetpreviousManaged() const ;
Renvoie l' **objet géré précédent**.

ImqBoolean queueManagerIdentificateur (ImqString & *id*) ;
Fournit une copie de l' **identificateur de gestionnaire de files d'attente**. Elle renvoie TRUE en cas de succès.

ImqString queueManagerIdentificateur () ;
Renvoie l' **identificateur de gestionnaire de files d'attente** sans aucune indication d'erreurs possibles.

Méthodes d'objet (protégées)

virtuel ImqBoolean closeTemporarily() ;
Ferme un objet en toute sécurité avant de le rouvrir. Elle renvoie TRUE en cas de succès. Cette méthode suppose que le **statut d'ouverture** est TRUE.

MQHCONN connectionHandle() const ;
Renvoie le MQHCONN associé à la **référence de connexion**. Cette valeur est égale à zéro s'il n'existe aucune **référence de connexion** ou si le gestionnaire n'est pas connecté.

ImqBoolean inquire(const MQLONG *int-attr*, MQLONG & *valeur*) ;
Renvoie une valeur entière dont l'index est une valeur MQIA_ *. En cas d'erreur, la valeur est définie sur MQIAV_UNDEFINED.

ImqBoolean inquire(const MQLONG *char-attr*, char * & *buffer*, const size_t *longueur*) ;
Renvoie une chaîne de caractères dont l'index est une valeur MQCA_ *.

Remarque : Ces deux méthodes ne renvoient qu'une seule valeur d'attribut. Si une *image instantanée* est requise de plusieurs valeurs, où les valeurs sont cohérentes les unes avec les autres pendant un instant, WebSphere MQ C++ ne fournit pas cette fonction et vous devez utiliser l'appel MQINQ avec les paramètres appropriés.

vide virtuel openInformationDisperse() ;
Disperse les informations de la section variable de la structure de données MQOD immédiatement après un appel MQOPEN.

virtuel ImqBoolean openInformationPrepare() ;
Prépare les informations pour la section variable de la structure de données MQOD immédiatement avant un appel MQOPEN et renvoie TRUE si l'opération aboutit.

ImqBoolean set(const MQLONG *int-attr*, const MQLONG *valeur*) ;
Définit un attribut entier WebSphere MQ .

ImqBoolean set(const MQLONG *char-attr*, const char * *buffer*, const size_t *required-length*) ;
Définit un attribut de caractère WebSphere MQ .

void setNextManagedObject(const ImqObject * *objet* = 0) ;
Définit l' **objet géré suivant**.

Attention: N'utilisez cette fonction que si vous êtes sûr qu'elle n'interrompt pas la liste des objets gérés.

void setPreviousManagedObject(const ImqObject * *objet* = 0) ;
Définit l' **objet géré précédent**.

Attention: N'utilisez cette fonction que si vous êtes sûr qu'elle n'interrompt pas la liste des objets gérés.

Données d'objet (protégées)

MQHOBJ *ohobj*
Descripteur d'objet WebSphere MQ (valide uniquement lorsque le **statut d'ouverture** est TRUE).

MQOD *od*
Structure de données MQOD imbriquée. La quantité de mémoire allouée à cette structure de données est celle requise pour un MQOD version 2. Inspectez le numéro de version *(omqod.Version)* et accédez aux autres zones comme suit:

MQOD_VERSION_1
Toutes les autres zones de *omqod* sont accessibles.

MQOD_VERSION_2
Toutes les autres zones de *omqod* sont accessibles.

MQOD_VERSION_3
omqod.pmqod est un pointeur vers un MQOD alloué de manière dynamique, plus grand. Aucune autre zone de *omqod* n'est accessible. Toutes les zones traitées par *omqod.pmqod* sont accessibles.

Remarque : *omqod.pmqod.Version* peut être inférieure à *omqod.Version*, indiquant que le client WebSphere MQ MQI dispose de plus de fonctionnalités que le serveur WebSphere MQ .

Codes raison

- MQRC_ATTRIBUT_LOCKED
- MQRC_INCONSISTENT_OBJECT_STATE
- MQRC_NO_RÉFÉRENCE_CONNEXION
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_REOPEN_SAVED_CONTEXT_ERR
- (codes anomalie de MQCLOSE)
- (codes anomalie de MQCONN)
- (codes anomalie de MQINQ)
- (codes anomalie de MQOPEN)
- (codes anomalie de MQSET)

Classe C++ ImqProcess

Cette classe encapsule un processus d'application (objet WebSphere MQ de type MQOT_PROCESS) qui peut être déclenché par un moniteur de déclenchement.

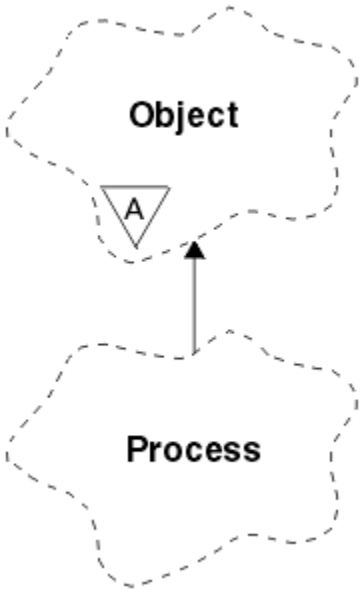


Figure 62. Classe ImqProcess

- «Attributs d'objet», à la page 1395
- «Constructeurs», à la page 1395
- «Méthodes d'objet (public)», à la page 1395

Attributs d'objet

Application ID
Identité du processus d'application. Cet attribut est en lecture seule.

Type d'Application
Type du processus d'application. Cet attribut est en lecture seule.

Données d'environnement
Informations d'environnement pour le processus. Cet attribut est en lecture seule.

données utilisateur
Données utilisateur du processus. Cet attribut est en lecture seule.

Constructeurs

ImqProcess( );
Constructeur par défaut.

ImqProcess(const ImqProcess & *process*) ;
Constructeur de copie. L' **état d'ouverture** ImqObject est FALSE.

ImqProcess(const char * *nom*) ;
Définit le **nom** ImqObject .

Méthodes d'objet (public)

void operator = (const ImqProcess & *process*) ;
Effectue une fermeture si nécessaire, puis copie les données d'instance à partir de *traitement*. L' **état d'ouverture** ImqObject est FALSE.

ImqBoolean applicationId(ImqString & *id*) ;
Fournit une copie de l' **ID application**. Elle renvoie TRUE en cas de succès.

ImqString applicationId( );
Renvoie l' **ID application** sans aucune indication d'erreurs possibles.

ImqBoolean applicationType(MQLONG &*type*) ;
Fournit une copie du **type d'application**. Elle renvoie TRUE en cas de succès.

MQLONG applicationType() ;
Renvoie le **type d'application** sans aucune indication d'erreurs possibles.

ImqBoolean environmentData(ImqString &*données*) ;
Fournit une copie des **données d'environnement**. Elle renvoie TRUE en cas de succès.

ImqString environmentData( );
Renvoie les **données d'environnement** sans indication des erreurs possibles.

ImqBoolean userData(ImqString &*données*) ;
Fournit une copie des **données utilisateur**. Elle renvoie TRUE en cas de succès.

ImqString userData( );
Renvoie les **données utilisateur** sans aucune indication d'erreurs possibles.

Classe C++ ImqPutMessageOptions

Cette classe encapsule la structure de données MQPMO.

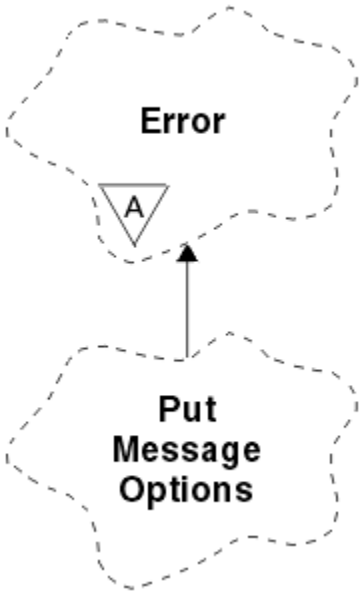


Figure 63. Classe ImqPutMessageOptions

- «Attributs d'objet», à la page 1396
- «Constructeurs», à la page 1397
- «Méthodes d'objet (public)», à la page 1397
- «Données d'objet (protégées)», à la page 1398
- «Codes raison», à la page 1398

Attributs d'objet

référence de contexte
Une file d'attente ImqQueue qui fournit un contexte pour les messages. Au départ, il n'y a pas de référence.

Options
Options du message d'insertion. La valeur initiale est MQPMO_NONE. Les valeurs supplémentaires suivantes sont possibles:

- MQPMO_SYNCPOINT
- MQPMO_NO_SYNCPOINT

- ID MQPMO_NEW_MSG_ID
- MQPMO_NEW_CORREL_ID
- MQPMO_LOGICAL_ORDER
- MQPMO_NO_CONTEXT
- MQPMO_CONTEXTE_PAR_DÉFAUT
- MQPMO_PASS_IDENTITY_CONTEXT
- MQPMO_PASS_ALL_CONTEXT
- MQPMO_SET_IDENTITY_CONTEXT
- MQPMO_SET_ALL_CONTEXT
- MQPMO_ALTERNATE_USER_AUTHORITY
- MQPMO_FAIL_IF_QUIESCING

zones d'enregistrement
Indicateurs qui contrôlent l'inclusion des enregistrements de message d'insertion lorsqu'un message est inséré. La valeur initiale est MQPMRF_NONE. Les valeurs supplémentaires suivantes sont possibles:

- ID MQPMRF_MSG_ID
- ID_CORREL_MQPMRF_
- ID_GROUPE_MQPMRF_ID
- MQPMRF_FEEDBACK
- MQPMRF_COMPTING_TOKEN

Les attributs de la fonction de suivi ImqMessagesont extraits de l'objet pour toute zone spécifiée. Les attributs de la fonction de suivi ImqMessagesont extraits de l'objet ImqMessage pour toute zone *non* spécifiée.

Nom de gestionnaire de files d'attente résolu
Nom d'un gestionnaire de files d'attente de destination déterminé lors d'une insertion. La valeur initiale est null. Cet attribut est en lecture seule.

Nom de file d'attente résolu
Nom d'une file d'attente de destination déterminée lors d'une insertion. La valeur initiale est null. Cet attribut est en lecture seule.

participation au point de synchronisation
TRUE lorsque les messages sont placés sous contrôle de point de synchronisation.

Constructeurs

ImqPutMessageOptions( );
Constructeur par défaut.

ImqPutMessageOptions(const ImqPutMessageOptions & *pmo*) ;
Constructeur de copie.

Méthodes d'objet (public)

void operator = (const ImqPutMessageOptions & *pmo*) ;
Copie les données d'instance à partir de *pmo*, en remplaçant les données d'instance existantes.

ImqQueue * contextReference() const ;
Renvoie la **référence de contexte**.

void setContextReference(const ImqQueue & *queue*) ;
Définit la **référence de contexte**.

void setContextReference(const ImqQueue * *queue* = 0) ;
Définit la **référence de contexte**.

MQLONG options() const ;
Renvoie les **options**.

void setOptions(const MQLONG *options*) ;
Définit les **options**, y compris la valeur **syncpoint participation** .

MQLONG recordFields() const ;
Renvoie les **zones d'enregistrement**.

void setRecordFields(const MQLONG *fields*) ;
Définit les **zones d'enregistrement**.

ImqString resolvedQueueManagerName() const ;
Renvoie une copie du **nom de gestionnaire de files d'attente résolu**.

ImqString resolvedQueueNom() const ;
Renvoie une copie du **nom de file d'attente résolue**.

ImqBoolean syncPointParticipation() const ;
Renvoie la valeur **syncpoint participation** , qui est TRUE si les **options** incluent MQPMO_SYNCPOINT.

void setSyncPointParticipation(const ImqBoolean *sync*) ;
Définit la valeur de **participation de point de synchronisation** . Si *sync* est TRUE, les **options** sont modifiées pour inclure MQPMO_SYNCPOINT et exclure MQPMO_NO_SYNCPOINT. Si *sync* est FALSE, les **options** sont modifiées pour inclure MQPMO_NO_SYNCPOINT et pour exclure MQPMO_SYNCPOINT.

Données d'objet (protégées)

MQPMO *omqpmo*
Structure de données MQPMO.

Codes raison

- MQRC_STORAGE_NOT_AVAILABLE

Classe C++ ImqQueue

Cette classe encapsule une file d'attente de messages (objet WebSphere MQ de type MQOT_Q).

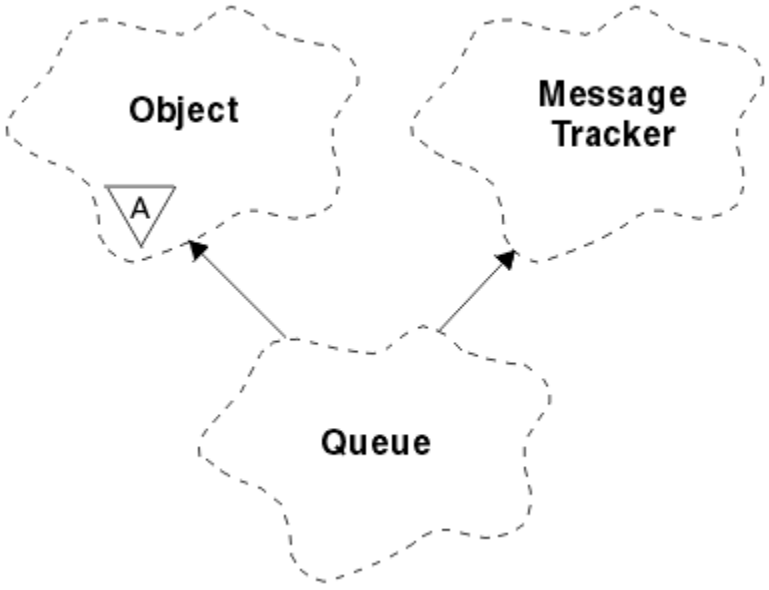


Figure 64. Classe ImqQueue

Cette classe est liée aux appels MQI répertoriés dans le Tableau 612, à la page 1335.

Attributs d'objet

Nom de file d'attente d'annulation
Nom de remise en file d'attente d'annulation trop élevé. Cet attribut est en lecture seule.

Seuil d'annulation
Seuil d'annulation. Cet attribut est en lecture seule.

nom de la file d'attente de base
Nom de la file d'attente dans laquelle l'alias est résolu. Cet attribut est en lecture seule.

Nom du cluster
Nom de cluster. Cet attribut est en lecture seule.

Liste de noms du cluster
Nom de la liste de noms de cluster. Cet attribut est en lecture seule.

Rang charge cluster
Rang charge cluster Cet attribut est en lecture seule.

Priorité charge cluster
Priorité charge cluster Cet attribut est en lecture seule.

File d'attente d'utilisation de charge de travail de cluster
Valeur de file d'attente d'utilisation de charge de travail de cluster Cet attribut est en lecture seule.

date de création
Données de création de file d'attente. Cet attribut est en lecture seule.

Heure de création
Heure de création de la file d'attente. Cet attribut est en lecture seule.

Longueur en cours
Nombre de messages dans la file d'attente. Cet attribut est en lecture seule.

liaison par défaut
Liaison par défaut. Cet attribut est en lecture seule.

Option d'ouverture en entrée par défaut
Option open-for-input par défaut. Cet attribut est en lecture seule.

Persistance par défaut
Persistance de message par défaut. Cet attribut est en lecture seule.

Priorité par défaut
Priorité de message par défaut. Cet attribut est en lecture seule.

Type de définition
Type de définition de file d'attente. Cet attribut est en lecture seule.

événement de profondeur élevée
Attribut de contrôle pour les événements de longueur élevée de file d'attente. Cet attribut est en lecture seule.

limite supérieure de profondeur
Limite supérieure de la longueur de la file d'attente. Cet attribut est en lecture seule.

événement de profondeur faible
Attribut de contrôle pour les événements de longueur faible de la file d'attente. Cet attribut est en lecture seule.

limite inférieure de profondeur
Limite inférieure de la longueur de la file d'attente. Cet attribut est en lecture seule.

événement de profondeur maximale
Attribut de contrôle pour les événements de longueur maximale de la file d'attente. Cet attribut est en lecture seule.

référence de liste de distribution
Référence facultative à une liste ImqDistributionqui peut être utilisée pour distribuer des messages à plusieurs files d'attente, y compris celle-ci. La valeur initiale est null.

Remarque : Lorsqu'un objet ImqQueue est ouvert, tout objet de liste ImqDistributionouvert auquel il fait référence est automatiquement fermé.

listes de distribution
Capacité d'une file d'attente de transmission à prendre en charge les listes de distribution. Cet attribut est en lecture seule.

Nom de la file d'attente dynamique
Nom de la file d'attente dynamique. La valeur initiale est AMQ.* pour toutes les plateformes Windows, UNIXet Linux .

Sauvegarde nb d'annulations
Indique s'il faut renforcer le nombre d'annulations. Cet attribut est en lecture seule.

Type d'index
Type d'index. Cet attribut est en lecture seule.

interdiction de l'obtention
Indique si les opérations d'extraction sont autorisées. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour un alias ou une file d'attente locale.

insertion interdite
Indique si les opérations d'insertion sont autorisées. La valeur initiale dépend de la définition de la file d'attente.

Nom de la file d'attente d'initialisation
Nom de la file d'attente d'initialisation. Cet attribut est en lecture seule.

Profondeur maximale
Nombre maximal de messages admis dans la file d'attente. Cet attribut est en lecture seule.

longueur maximale des messages
Longueur maximale de tout message de cette file d'attente, qui peut être inférieure à la longueur maximale de toute file d'attente gérée par le gestionnaire de files d'attente associé. Cet attribut est en lecture seule.

Séquence livraison messages
Indique si la priorité du message est pertinente. Cet attribut est en lecture seule.

file d'attente distribuée suivante
Objet suivant de cette classe, sans ordre particulier, ayant la même **référence de liste de distribution** que cet objet. La valeur initiale est zéro.

Si un objet d'une chaîne est supprimé, l'objet précédent et l'objet suivant sont mis à jour de sorte que leurs liens de file d'attente répartie ne pointent plus vers l'objet supprimé.

classe de message non persistant
Niveau de fiabilité des messages non persistants insérés dans cette file d'attente. Cet attribut est en lecture seule.

Nombre d'ouvertures de file d'attente en entrée
Nombre d'objets ImqQueue ouverts en entrée. Cet attribut est en lecture seule.

Nombre d'ouvertures en sortie
Nombre d'objets ImqQueue ouverts pour la sortie. Cet attribut est en lecture seule.

file d'attente distribuée précédente
Objet précédent de cette classe, sans ordre particulier, ayant la même **référence de liste de distribution** que cet objet. La valeur initiale est zéro.

Si un objet d'une chaîne est supprimé, l'objet précédent et l'objet suivant sont mis à jour de sorte que leurs liens de file d'attente répartie ne pointent plus vers l'objet supprimé.

Nom de processus
Nom de la définition de processus. Cet attribut est en lecture seule.

Comptabilité file
Niveau des informations de comptabilité pour les files d'attente. Cet attribut est en lecture seule.

nom de gestionnaire de files d'attente
Nom du gestionnaire de files d'attente (éventuellement éloigné) où se trouve la file d'attente.
Ne confondez pas le gestionnaire de files d'attente nommé ici avec la **référence de connexion** ImqObject , qui fait référence au gestionnaire de files d'attente (local) fournissant une connexion. La valeur initiale est null.

Ctrl file d'attente
Niveau de collecte des données de surveillance pour la file d'attente. Cet attribut est en lecture seule.

statistiques de file d'attente
Niveau des données statistiques de la file d'attente. Cet attribut est en lecture seule.

Type de file d'attente
Type de file d'attente. Cet attribut est en lecture seule.

Nom du gestionnaire de files d'attente éloignées
Nom du gestionnaire de files d'attente éloignées. Cet attribut est en lecture seule.

Nom de la file d'attente distante
Nom de la file d'attente éloignée tel qu'il est connu sur le gestionnaire de files d'attente éloignées. Cet attribut est en lecture seule.

Nom de gestionnaire de files d'attente résolu
Nom du gestionnaire de files d'attente résolu. Cet attribut est en lecture seule.

Nom de file d'attente résolu
Nom de file d'attente résolu. Cet attribut est en lecture seule.

Intervalle de conservation
Intervalle de conservation de la file d'attente. Cet attribut est en lecture seule.

Portée
Portée de la définition de file d'attente. Cet attribut est en lecture seule.

intervalle de maintenance
Intervalle de maintenance. Cet attribut est en lecture seule.

événement d'intervalle de maintenance
Attribut de contrôle des événements d'intervalle de service. Cet attribut est en lecture seule.

Partage de file d'attente
Indique si la file d'attente peut être partagée. Cet attribut est en lecture seule.

classe d'archivage
Classe de stockage. Cet attribut est en lecture seule.

Nom de la file d'attente de transmission
Nom de la file d'attente de transmission. Cet attribut est en lecture seule.

Contrôle du déclenchement
Contrôle du déclencheur. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

Données de déclenchement
Données de déclenchement. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

Longueur de déclenchement
Longueur de déclenchement. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

Priorité msg pour déclench.
Priorité de message de seuil pour les déclencheurs. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

type du déclencheur
Type de déclencheur. La valeur initiale dépend de la définition de la file d'attente. Cet attribut est valide uniquement pour une file d'attente locale.

Utilisation
Utilisation. Cet attribut est en lecture seule.

Constructeurs

ImqQueue( );
Constructeur par défaut.

ImqQueue(const ImqQueue & ***file d'attente*** **) ;**
Constructeur de copie. L' **état d'ouverture** ImqObject est FALSE.

ImqQueue(const char * ***nom*** **) ;**
Définit le **nom** ImqObject .

Méthodes d'objet (public)

void operator = (const ImqQueue & ***file d'attente*** **) ;**
Effectue une fermeture si nécessaire, puis copie les données d'instance à partir de *file d'attente*. L' **état d'ouverture** ImqObject est FALSE.

ImqBoolean backoutRequeueNom(ImqString & ***nom*** **) ;**
Fournit une copie du **nom de remise en file d'attente d'annulation**. Elle renvoie TRUE en cas de succès.

ImqString backoutRequeueNom() ;
Renvoie le **nom de remise en file d'attente d'annulation** sans aucune indication d'erreurs possibles.

ImqBoolean backoutThreshold(MQLONG & ***seuil*** **) ;**
Fournit une copie du **seuil d'annulation**. Elle renvoie TRUE en cas de succès.

MQLONG backoutThreshold() ;
Renvoie la valeur de **seuil d'annulation** sans indication d'erreurs possibles.

ImqBoolean baseQueueNom(ImqString & ***nom*** **) ;**
Fournit une copie du **nom de la file d'attente de base**. Elle renvoie TRUE en cas de succès.

ImqString baseQueueNom() ;
Renvoie le **nom de la file d'attente de base** sans indication des erreurs possibles.

ImqBoolean clusterName(ImqString & ***nom*** **) ;**
Fournit une copie du **nom de cluster**. Elle renvoie TRUE en cas de succès.

ImqString clusterName( );
Renvoie le **nom de cluster** sans aucune indication d'erreurs possibles.

ImqBoolean clusterNamelistNom (ImqString & ***nom*** **) ;**
Fournit une copie du **nom de liste de noms de cluster**. Elle renvoie TRUE en cas de succès.

ImqString clusterNamelistNom () ;
Renvoie le **nom de liste de noms de cluster** sans aucune indication d'erreurs.

ImqBoolean clusterWorkLoadPriority (MQLONG & priorité) ;
Fournit une copie de la valeur de priorité de charge de travail du cluster. Elle renvoie TRUE en cas de succès.

MQLONG clusterWorkLoadPriority () ;
Renvoie la valeur de priorité de la charge de travail du cluster sans aucune indication d'erreurs possibles.

ImqBoolean clusterWorkLoadRank (MQLONG & rank) ;
Fournit une copie de la valeur de rang de charge de travail du cluster. Elle renvoie TRUE en cas de succès.

MQLONG clusterWorkLoadRank () ;
Renvoie la valeur de rang de charge de travail du cluster sans aucune indication d'erreurs possibles.

ImqBoolean clusterWorkLoadUseQ (MQLONG & useq) ;
Fournit une copie de la valeur de la file d'attente d'utilisation de la charge de travail du cluster. Elle renvoie TRUE en cas de succès.

MQLONG clusterWorkLoadUseQ () ;
Renvoie la valeur de la file d'attente d'utilisation de la charge de travail du cluster sans aucune indication d'erreurs possibles.

ImqBoolean creationDate(ImqString & ***date*** **) ;**
Fournit une copie de la **date de création**. Elle renvoie TRUE en cas de succès.

ImqString creationDate( );
Renvoie la **date de création** sans aucune indication d'erreurs possibles.

ImqBoolean creationTime(ImqString & ***heure*** **) ;**
Fournit une copie de l' **heure de création**. Elle renvoie TRUE en cas de succès.

ImqString creationTime( );
Renvoie l' **heure de création** sans aucune indication des erreurs possibles.

ImqBoolean currentDepth(MQLONG & ***profondeur*** **) ;**
Fournit une copie de la **profondeur actuelle**. Elle renvoie TRUE en cas de succès.

MQLONG currentDepth() ;
Renvoie la **profondeur en cours** sans aucune indication d'erreurs possibles.

ImqBoolean defaultInputOpenOption(MQLONG & ***option*** **) ;**
Fournit une copie de l' **option d'ouverture d'entrée par défaut**. Elle renvoie TRUE en cas de succès.

MQLONG defaultInputOpenOption() ;
Renvoie l' **option d'ouverture d'entrée par défaut** sans aucune indication d'erreurs possibles.

ImqBoolean defaultPersistence(MQLONG & ***persistance*** **) ;**
Fournit une copie de la **persistance par défaut**. Elle renvoie TRUE en cas de succès.

MQLONG defaultPersistence() ;
Renvoie la **persistance par défaut** sans indication des erreurs possibles.

ImqBoolean defaultPriority(MQLONG & ***priorité*** **) ;**
Fournit une copie de la **priorité par défaut**. Elle renvoie TRUE en cas de succès.

MQLONG defaultPriority() ;
Renvoie la **priorité par défaut** sans aucune indication d'erreurs possibles.

ImqBoolean defaultBind(MQLONG & ***liaison*** **) ;**
Fournit une copie de la **liaison par défaut**. Elle renvoie TRUE en cas de succès.

MQLONG defaultBind() ;
Renvoie la **liaison par défaut** sans aucune indication d'erreurs possibles.

ImqBoolean definitionType(MQLONG & ***type*** **) ;**
Fournit une copie du **type de définition**. Elle renvoie TRUE en cas de succès.

MQLONG definitionType() ;
Renvoie le **type de définition** sans aucune indication d'erreurs possibles.

ImqBoolean depthHighEvent(MQLONG & ***événement*** **) ;**
Fournit une copie de l'état d'activation de l' **événement de profondeur élevée**. Elle renvoie TRUE en cas de succès.

MQLONG depthHighEvénement() ;
Renvoie l'état d'activation de l' **événement de profondeur élevée** sans aucune indication d'erreurs possibles.

ImqBoolean depthHighLimite(MQLONG & ***limite*** **);**
Fournit une copie de la **limite supérieure de profondeur**. Elle renvoie TRUE en cas de succès.

MQLONG depthHighLimite() ;
Renvoie la valeur de la **limite supérieure de profondeur** sans aucune indication d'erreurs possibles.

ImqBoolean depthLowEvénement(MQLONG & ***événement*** **) ;**
Fournit une copie de l'état d'activation de l' **événement de profondeur faible**. Elle renvoie TRUE en cas de succès.

MQLONG depthLowEvénement() ;
Renvoie l'état d'activation de l' **événement de profondeur faible** sans aucune indication d'erreurs possibles.

ImqBoolean depthLowLimite(MQLONG & ***limite*** **);**
Fournit une copie de la **limite inférieure de profondeur**. Elle renvoie TRUE en cas de succès.

MQLONG depthLowLimite() ;
Renvoie la valeur de la **limite inférieure de profondeur** sans aucune indication d'erreurs possibles.

ImqBoolean depthMaximumEvénement(MQLONG & ***événement*** **) ;**
Fournit une copie de l'état d'activation de l' **événement maximal de profondeur**. Elle renvoie TRUE en cas de succès.

MQLONG depthMaximumEvénement() ;
Renvoie l'état d'activation de l' **événement maximal de profondeur** sans aucune indication d'erreurs possibles.

ImqDistributionListe * distributionListRéférence() const ;
Renvoie la **référence de liste de distribution**.

void setDistributionListReference(ImqDistributionList & ***list*** **) ;**
Définit la **référence de liste de distribution**.

void setDistributionListReference(ImqDistributionList * ***list*** **= 0) ;**
Définit la **référence de liste de distribution**.

ImqBoolean distributionLists(MQLONG & ***support*** **) ;**
Fournit une copie de la valeur des **listes de distribution** . Elle renvoie TRUE en cas de succès.

MQLONG distributionLists() ;
Renvoie la valeur de **listes de distribution** sans aucune indication d'erreurs possibles.

ImqBoolean setDistributionLists(const MQLONG ***support*** **) ;**
Définit la valeur des **listes de distribution** . Elle renvoie TRUE en cas de succès.

ImqString dynamicQueueNom() const ;
Renvoie une copie du **nom de file d'attente dynamique**.

ImqBoolean setDynamicQueueName(const char * ***nom*** **) ;**
Définit le **nom de la file d'attente dynamique**. Le **nom de file d'attente dynamique** ne peut être défini que si le **statut d'ouverture** ImqObject est FALSE. Elle renvoie TRUE en cas de succès.

ImqBoolean get(ImqMessage & ***msg*****, ImqGetMessageOptions &** ***options*** **) ;**
Extrait un message de la file d'attente à l'aide des *options*spécifiées. Appelle la méthode ImqObject **openFor** si nécessaire pour s'assurer que les **options d'ouverture** ImqObject incluent l'une des valeurs MQOO_INPUT_ * ou MQOO_BROWSE, en fonction des *options*. Si l'objet *msg* possède une **mémoire tampon automatique** ImqCache , la taille de la mémoire tampon est adaptée à tous les messages extraits. La méthode **clearMessage** est appelée sur l'objet *msg* avant l'extraction.

Cette méthode renvoie TRUE en cas de succès.

Remarque : Le résultat de l'appel de méthode est FALSE si le code anomalie ImqObject est MQRC_TRUNCATED_MSG_FAILED, même si ce **code anomalie** est classé comme un avertissement. Si un message tronqué est accepté, la **longueur de message** ImqCache reflète la longueur tronquée. Dans les deux cas, la **longueur totale du message** ImqMessage indique le nombre d'octets disponibles.

ImqBoolean get(ImqMessage &*msg*) ;
Comme pour la méthode précédente, sauf que les options d'obtention de message par défaut sont utilisées.

ImqBoolean get(ImqMessage &*msg*, ImqGetMessageOptions &*options*, const size_t *buffer-size*) ;
Comme pour les deux méthodes précédentes, sauf qu'une *taille de mémoire tampon* de remplacement est indiquée. Si l'objet *msg* utilise une **mémoire tampon automatique** ImqCache , la méthode **resizeBuffer** est appelée sur l'objet *msg* avant l'extraction des messages et la mémoire tampon ne s'agrandit pas davantage pour accueillir les messages plus volumineux.

ImqBoolean get(ImqMessage &*msg*, const size_t *taille-tampon*) ;
Comme pour la méthode précédente, sauf que les options d'obtention de message par défaut sont utilisées.

ImqBoolean hardenGetBackout(MQLONG &*harden*) ;
Fournit une copie de la valeur **harden get backout** . Elle renvoie TRUE en cas de succès.

MQLONG hardenGetAnnulation() ;
Renvoie la valeur **harden get backout** sans aucune indication d'erreurs possibles.

ImqBoolean indexType(MQLONG &*type*) ;
Fournit une copie du **type d'index**. Elle renvoie TRUE en cas de succès.

MQLONG indexType() ;
Renvoie le **type d'index** sans aucune indication des erreurs possibles.

ImqBoolean inhibitGet(MQLONG &*inhibition*) ;
Fournit une copie de la valeur **inhibition de l'extraction** . Elle renvoie TRUE en cas de succès.

MQLONG inhibitGet() ;
Renvoie la valeur **inhibant l'obtention** sans aucune indication d'erreurs possibles.

ImqBoolean setInhibitGet(const MQLONG *inhibition*) ;
Définit la valeur **inhibition de l'extraction** . Elle renvoie TRUE en cas de succès.

ImqBoolean inhibitPut(MQLONG &*inhiber*) ;
Fournit une copie de la valeur **inhibition de l'insertion** . Elle renvoie TRUE en cas de succès.

MQLONG inhibitPut() ;
Renvoie la valeur **inhibant l'insertion** sans aucune indication d'erreurs possibles.

ImqBoolean setInhibitPut(const MQLONG *inhiber*) ;
Définit la valeur **inhibant l'insertion** . Elle renvoie TRUE en cas de succès.

ImqBoolean initiationQueueNom(ImqString &*nom*) ;
Fournit une copie du **nom de la file d'attente d'initialisation**. Elle renvoie TRUE en cas de succès.

ImqString initiationQueueNom() ;
Renvoie le **nom de la file d'attente d'initialisation** sans aucune indication d'erreurs possibles.

ImqBoolean maximumDepth(MQLONG &*profondeur*) ;
Fournit une copie de la **profondeur maximale**. Elle renvoie TRUE en cas de succès.

MQLONG maximumDepth() ;
Renvoie la **profondeur maximale** sans indication des erreurs possibles.

ImqBoolean maximumMessageLongueur(MQLONG &*longueur*) ;
Fournit une copie de la **longueur maximale de message**. Elle renvoie TRUE en cas de succès.

MQLONG maximumMessageLongueur() ;
Renvoie la **longueur maximale de message** sans aucune indication d'erreurs possibles.

ImqBoolean messageDeliverySéquence(MQLONG &*séquence*) ;
Fournit une copie de la **séquence de distribution des messages**. Elle renvoie TRUE en cas de succès.

MQLONG messageDeliverySéquence() ;
Renvoie la valeur de la **séquence de distribution des messages** sans aucune indication d'erreurs possibles.

ImqQueue * nextDistributedFile d'attente() const ;
Renvoie la **file d'attente distribuée suivante**.

ImqBoolean nonPersistentMessageClass (MQLONG & monq) ;
Fournit une copie de la valeur de classe de message non persistant. Elle renvoie TRUE en cas de succès.

MQLONG nonPersistentMessageClass () ;
Renvoie la valeur de classe de message non persistant sans aucune indication d'erreurs possibles.

ImqBoolean openInputCount(MQLONG & ***nombre*** **) ;**
Fournit une copie du **nombre d'entrées ouvertes**. Elle renvoie TRUE en cas de succès.

MQLONG openInputNombre() ;
Renvoie le **nombre d'entrées ouvertes** sans aucune indication d'erreurs possibles.

ImqBoolean openOutputCount(MQLONG & ***nombre*** **) ;**
Fournit une copie du **nombre de sorties ouvertes**. Elle renvoie TRUE en cas de succès.

MQLONG openOutputNombre() ;
Renvoie le **nombre de sorties ouvertes** sans aucune indication d'erreurs possibles.

ImqQueue * previousDistributedprécédente distribuée() const ;
Renvoie la **file d'attente répartie précédente**.

ImqBoolean processName(ImqString & ***nom*** **) ;**
Fournit une copie du **nom de processus**. Elle renvoie TRUE en cas de succès.

ImqString processName( );
Renvoie le **nom de processus** sans aucune indication des erreurs possibles.

ImqBoolean put(ImqMessage & ***msg*** **) ;**
Place un message dans la file d'attente à l'aide des options de message d'insertion par défaut. Utilise la méthode ImqObject **openFor** si nécessaire pour s'assurer que les **options d'ouverture** ImqObject incluent MQOO_OUTPUT.

Cette méthode renvoie TRUE en cas de succès.

ImqBoolean put(ImqMessage & ***msg*****, ImqPutMessageOptions &** ***pmo*** **) ;**
Place un message dans la file d'attente, en utilisant le *pmo*spécifié. Utilise la méthode ImqObject **openFor** si nécessaire pour s'assurer que ImqObject **options d'ouverture** inclut MQOO_OUTPUT et (si *Pmo* **options** inclut l'une des valeurs MQPMO_PASS_IDENTITY_CONTEXT, MQPMO_PASS_ALL_CONTEXT, MQPMO_SET_IDENTITY_CONTEXT ou MQPMO_ALL_CONTEXT) correspondantes.

Cette méthode renvoie TRUE en cas de succès.

Remarque : Si *pmo* inclut une **référence de contexte**, l'objet référencé est ouvert, si nécessaire, pour fournir un contexte.

ImqBoolean queueAccounting (MQLONG & acctq) ;
Fournit une copie de la valeur de comptabilité de la file d'attente. Elle renvoie TRUE en cas de succès.

MQLONG queueAccounting () ;
Renvoie la valeur de comptabilité de la file d'attente sans indication d'erreurs possibles.

ImqString queueManagerNom() const ;
Renvoie le **nom du gestionnaire de files d'attente**.

ImqBoolean setQueueManagerName(const char * ***nom*** **) ;**
Définit le **nom du gestionnaire de files d'attente**. Le **nom du gestionnaire de files d'attente** ne peut être défini que si le **statut d'ouverture** ImqObject est FALSE. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean queueMonitoring (MQLONG & monq) ;
Fournit une copie de la valeur de surveillance de file d'attente. Elle renvoie TRUE en cas de succès.

MQLONG queueMonitoring () ;
Renvoie la valeur de surveillance de file d'attente sans aucune indication d'erreurs possibles.

ImqBoolean queueStatistics (MQLONG & statq) ;
Fournit une copie de la valeur des statistiques de file d'attente. Elle renvoie TRUE en cas de succès.

MQLONG queueStatistics () ;
Renvoie la valeur des statistiques de file d'attente sans indication d'erreurs possibles.

ImqBoolean queueType(MQLONG &*type*) ;
Fournit une copie de la valeur du **type de file d'attente** . Elle renvoie TRUE en cas de succès.

MQLONG queueType() ;
Renvoie le **type de file d'attente** sans aucune indication d'erreurs possibles.

ImqBoolean remoteQueueManagerName(ImqString &*nom*) ;
Fournit une copie du **nom du gestionnaire de files d'attente éloignées**. Elle renvoie TRUE en cas de succès.

ImqString remoteQueueManagerName( );
Renvoie le **nom du gestionnaire de files d'attente éloignées** sans indiquer d'erreurs possibles.

ImqBoolean remoteQueueNom(ImqString &*nom*) ;
Fournit une copie du **nom de la file d'attente éloignée**. Elle renvoie TRUE en cas de succès.

ImqString remoteQueueNom() ;
Renvoie le **nom de la file d'attente éloignée** sans indiquer d'erreurs possibles.

ImqBoolean resolvedQueueManagerName(ImqString &*nom*) ;
Fournit une copie du **nom du gestionnaire de files d'attente résolu**. Elle renvoie TRUE en cas de succès.

Remarque : Cette méthode échoue sauf si MQOO_RESOLVE_NAMES fait partie des **options d'ouverture** ImqObject .

ImqString resolvedQueueManagerName( ) ;
Renvoie le **nom du gestionnaire de files d'attente résolu**, sans aucune indication d'erreurs possibles.

ImqBoolean resolvedQueueNom (ImqString &*nom*) ;
Fournit une copie du **nom de file d'attente résolue**. Elle renvoie TRUE en cas de succès.

Remarque : Cette méthode échoue sauf si MQOO_RESOLVE_NAMES fait partie des **options d'ouverture** ImqObject .

ImqString resolvedQueueNom () ;
Renvoie le **nom de la file d'attente résolue**, sans aucune indication d'erreurs possibles.

ImqBoolean retentionInterval(MQLONG &*intervalle*) ;
Fournit une copie de l' **intervalle de conservation**. Elle renvoie TRUE en cas de succès.

MQLONG retentionInterval() ;
Renvoie l' **intervalle de conservation** sans aucune indication d'erreurs possibles.

ImqBoolean scope(MQLONG &*portée*) ;
Fournit une copie de la **portée**. Elle renvoie TRUE en cas de succès.

MQLONG portée() ;
Renvoie la **portée** sans indication des erreurs possibles.

ImqBoolean serviceInterval(MQLONG &*intervalle*) ;
Fournit une copie de l' **intervalle de service**. Elle renvoie TRUE en cas de succès.

MQLONG serviceInterval() ;
Renvoie l' **intervalle de service** sans aucune indication d'erreurs possibles.

ImqBoolean serviceIntervalEvénement(MQLONG &*événement*) ;
Fournit une copie de l'état d'activation de l' **événement d'intervalle de service**. Elle renvoie TRUE en cas de succès.

MQLONG serviceIntervalEvénement() ;
Renvoie l'état d'activation de l' **événement d'intervalle de service** sans aucune indication d'erreurs possibles.

ImqBoolean partageabilité(MQLONG &*partageabilité*) ;
Fournit une copie de la valeur **shareability** . Elle renvoie TRUE en cas de succès.

MQLONG partageabilité() ;
Renvoie la valeur **shareability** sans aucune indication des erreurs possibles.

ImqBoolean storageClass(ImqString & *classe*) ;
Fournit une copie de la **classe de stockage**. Elle renvoie TRUE en cas de succès.

ImqString storageClass( );
Renvoie la **classe de stockage** sans aucune indication d'erreurs possibles.

ImqBoolean transmissionQueueNom(ImqString & *nom*) ;
Fournit une copie du **nom de la file d'attente de transmission**. Elle renvoie TRUE en cas de succès.

ImqString transmissionQueueNom() ;
Renvoie le **nom de la file d'attente de transmission** sans aucune indication d'erreurs possibles.

ImqBoolean triggerControl(MQLONG & *contrôle*) ;
Fournit une copie de la valeur **trigger control** . Elle renvoie TRUE en cas de succès.

MQLONG triggerControl() ;
Renvoie la valeur **trigger control** sans aucune indication d'erreurs possibles.

ImqBoolean setTriggerControl(const MQLONG *contrôle*) ;
Définit la valeur du **contrôle de déclencheur** . Elle renvoie TRUE en cas de succès.

ImqBoolean triggerData(ImqString & *données*) ;
Fournit une copie des **données de déclenchement**. Elle renvoie TRUE en cas de succès.

ImqString triggerData( );
Renvoie une copie des **données de déclencheur** sans indication d'erreurs possibles.

ImqBoolean setTriggerData(const char * *données*) ;
Définit les **données de déclenchement**. Elle renvoie TRUE en cas de succès.

ImqBoolean triggerDepth(MQLONG & *profondeur*) ;
Fournit une copie de la **profondeur du déclencheur**. Elle renvoie TRUE en cas de succès.

MQLONG triggerDepth() ;
Renvoie la **profondeur du déclencheur** sans aucune indication d'erreurs possibles.

ImqBoolean setTriggerDepth(const MQLONG *profondeur*) ;
Définit la **profondeur du déclencheur**. Elle renvoie TRUE en cas de succès.

ImqBoolean triggerMessagePriorité(MQLONG & *priorité*) ;
Fournit une copie de la **priorité du message de déclenchement**. Elle renvoie TRUE en cas de succès.

MQLONG triggerMessagePriorité() ;
Renvoie la **priorité de message de déclenchement** sans aucune indication d'erreurs possibles.

ImqBoolean setTriggerMessagePriority(const MQLONG *priorité*) ;
Définit la **priorité des messages de déclenchement**. Elle renvoie TRUE en cas de succès.

ImqBoolean triggerType(MQLONG & *type*) ;
Fournit une copie du **type de déclencheur**. Elle renvoie TRUE en cas de succès.

MQLONG triggerType() ;
Renvoie le **type de déclencheur** sans aucune indication d'erreurs possibles.

ImqBoolean setTriggerType(const MQLONG *type*) ;
Définit le **type de déclencheur**. Elle renvoie TRUE en cas de succès.

ImqBoolean usage(MQLONG & *usage*) ;
Fournit une copie de la valeur **usage** . Elle renvoie TRUE en cas de succès.

MQLONG syntaxe() ;
Renvoie la valeur **usage** sans aucune indication d'erreurs possibles.

Méthodes d'objet (protégées)

void setNextDistributedQueue(ImqQueue * *file d'attente* = 0) ;
Définit la **file d'attente répartie suivante**.

Attention: utilisez cette fonction uniquement si vous êtes sûr qu'elle n'interrompt pas la liste des files d'attente réparties.

void setPreviousDistributedQueue(ImqQueue * *file d'attente* = 0) ;

Définit la **file d'attente distribuée précédente**.

Attention: utilisez cette fonction uniquement si vous êtes sûr qu'elle n'interrompt pas la liste des files d'attente réparties.

Codes raison

- MQRC_ATTRIBUT_LOCKED
- MQRC_CONTEXT_OBJET_NON_VALIDE
- MQRC_CONTEXT_OPEN_ERROR
- MQRC_CURSOR_NOT_VALID
- MQRC_NO_BUFFER
- MQRC_REOPEN_EXCL_INPUT_ERROR
- MQRC_REOPEN_INQUIRE_ERROR
- MQRC_REOPEN_TEMPORARY_Q_ERROR
- (codes anomalie de MQGET)
- (codes anomalie de MQPUT)

Classe C++ du gestionnaire ImqQueue

Cette classe encapsule un gestionnaire de files d'attente (objet WebSphere MQ de type MQOT_Q_MGR).

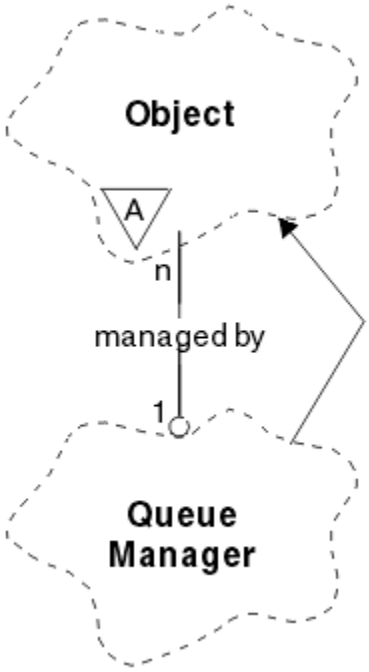


Figure 65. Classe de gestionnaire ImqQueue

Cette classe est liée aux appels MQI répertoriés dans le «Références croisées du gestionnaire ImqQueue», à la page 1337. Toutes les méthodes répertoriées ne sont pas applicables à toutes les plateformes ; voir ALTER QMGR pour plus de détails.

- «Attributs de classe», à la page 1410
- «Attributs d'objet», à la page 1410
- «Constructeurs», à la page 1415
- «Destructeurs», à la page 1416

- «Méthodes de classe (public)», à la page 1416
- «Méthodes d'objet (public)», à la page 1416
- «Méthodes d'objet (protégées)», à la page 1425
- «Données d'objet (protégées)», à la page 1425
- «Codes raison», à la page 1425

Attributs de classe

comportement
Contrôle le comportement de la connexion et de la déconnexion implicites.

IMQ_EXPL_DISC_BACKOUT (0L)
Un appel explicite à la méthode **disconnect** implique l'annulation. Cet attribut et IMQ_EXPL_DISC_COMMIT s'excluent mutuellement.

IMQ_EXPL_DISC_COMMIT (1L)
Un appel explicite à la méthode **disconnect** implique la validation (valeur par défaut). Cet attribut et IMQ_EXPL_DISC_BACKOUT s'excluent mutuellement.

IMQ_IMPL_CONN (2L)
La connexion implicite est autorisée (valeur par défaut).

IMQ_IMPL_DISC_BACKOUT (0L)
Un appel implicite à la méthode **disconnect** , qui peut se produire lors de la destruction d'objet, implique une annulation. Cet attribut est mutuellement exclusif avec IMQ_IMPL_DISC_COMMIT.

IMQ_IMPL_DISC_COMMIT (4L)
Un appel implicite à la méthode **disconnect** , qui peut se produire lors de la destruction d'objet, implique une validation (valeur par défaut). Cet attribut et IMQ_IMPL_DISC_BACKOUT s'excluent mutuellement.

Dans WebSphere MQ V7.0 et versions ultérieures, les applications C++ qui utilisent une connexion implicite doivent spécifier IMQ_IMPL_CONN avec les autres options fournies dans la méthode `setBehavior()` sur un objet de classe `ImqQueueManager`. Si votre application n'utilise pas la méthode `setBehavior()` pour définir explicitement les options de comportement, par exemple,

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

Cette modification ne vous affecte pas car MQ_IMPL_CONN est activé par défaut.

Si votre application définit explicitement les options de comportement, par exemple,

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

vous devez inclure IMQ_IMPL_CONN dans la méthode `setBehavior()` comme suit pour permettre à votre application d'établir une connexion implicite:

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_CONN | IMQ_IMPL_DISC_COMMIT)
```

Attributs d'objet

remplacement des connexions de comptabilité
Permet aux applications de remplacer les valeurs values.This est en lecture seule.

Intervalle de comptabilité
Durée avant l'écriture des enregistrements comptables intermédiaires (en secondes). Cet attribut est en lecture seule.

Enregistrement de l'activité
Ctrl la génération des rapports d'activité. Cet attribut est en lecture seule.

Adoption de la vérification du nouveau MCA
Les éléments vérifiés pour déterminer si un agent MCA doit être adopté lorsqu'un nouveau canal entrant portant le même nom qu'un agent MCA déjà actif est détecté. Cet attribut est en lecture seule.

Adoption du nouveau type de MCA
Indique si une instance orpheline d'un agent MCA d'un type de canal particulier doit être redémarrée automatiquement lorsqu'une nouvelle demande de canal entrant correspondant aux nouveaux paramètres de vérification mca est détectée. Cet attribut est en lecture seule.

Type d'authentification
Indique le type d'authentification en cours d'exécution.

événement d'autorité
Contrôle les événements de droits d'accès. Cet attribut est en lecture seule.

Options de début
Options qui s'appliquent à la méthode **begin** . La valeur initiale est MQBO_NONE.

événement de pont
Indique si les événements de pont IMS sont générés. Cet attribut est en lecture seule.

Définition automatique de canal
Valeur de définition automatique de canal. Cet attribut est en lecture seule.

événement de définition automatique de canal
Valeur d'événement de définition automatique de canal. Cet attribut est en lecture seule.

Exit de définition automatique de canal
Nom de l'exit de définition automatique de canal. Cet attribut est en lecture seule.

événement Canal
Indique si des événements de canal sont générés. Cet attribut est en lecture seule.

Adaptateurs d'initiateur de canal
Nombre de sous-tâches d'adaptateur à utiliser pour le traitement des appels WebSphere MQ . Cet attribut est en lecture seule.

Ctrl init. canal
Indique si l'initialisateur de canal doit être démarré automatiquement au démarrage du gestionnaire de files d'attente. Cet attribut est en lecture seule.

Répartiteurs d'initiateur de canal
Nombre de répartiteurs à utiliser pour l'initiateur de canal. Cet attribut est en lecture seule.

démarrage automatique de la trace de l'initiateur de canal
Indique si la trace de l'initiateur de canal doit démarrer automatiquement ou non. Cet attribut est en lecture seule.

Taille de la table de trace de l'initiateur de canal
Taille de l'espace de données de trace de l'initiateur de canal (en Mo). Cet attribut est en lecture seule.

Contrôle des canaux
Ctrl la collecte des données de surveillance en ligne pour les canaux. Cet attribut est en lecture seule.

référence de canal
Référence à une définition de canal à utiliser lors de la connexion client. Lorsqu'il est connecté, cet attribut peut être défini sur null, mais ne peut pas être remplacé par une autre valeur. La valeur initiale est null.

Statistiques de canal
Ctrl la collecte des données de statistiques pour les canaux. Cet attribut est en lecture seule.

jeu de caractères
ID de jeu de caractères codés (CCSID). Cet attribut est en lecture seule.

Ctrl émetteur cluster
Contrôle la collecte des données de surveillance en ligne pour les canaux émetteurs de cluster définis automatiquement. Cet attribut est en lecture seule.

Stats émetteur cluster
Contrôle la collecte des données statistiques pour les canaux émetteurs de cluster définis automatiquement. Cet attribut est en lecture seule.

Données de charge de travail de cluster
Données d'exit de charge de travail de cluster. Cet attribut est en lecture seule.

Exit de charge de travail du cluster
Nom de l'exit de charge de travail de cluster. Cet attribut est en lecture seule.

Longueur de charge de travail de cluster
Longueur de la charge de travail du cluster. Cet attribut est en lecture seule.

mru de charge de travail de cluster
La charge de travail du cluster a utilisé la valeur de canaux la plus récente. Cet attribut est en lecture seule.

File d'attente d'utilisation de charge de travail de cluster
Valeur de file d'attente d'utilisation de charge de travail de cluster Cet attribut est en lecture seule.

événement de commande
Indique si des événements de commande sont générés. Cet attribut est en lecture seule.

Nom de la file d'attente d'entrée des commandes
Nom de la file d'attente d'entrée des commandes système. Cet attribut est en lecture seule.

Niveau de commande
Niveau de commande pris en charge par le gestionnaire de files d'attente. Cet attribut est en lecture seule.

Contrôle du serveur de commandes
Indique si le serveur de commandes doit être démarré automatiquement lorsque le gestionnaire de files d'attente est démarré. Cet attribut est en lecture seule.

Options de connexion
Options qui s'appliquent à la méthode **connect** . La valeur initiale est MQCNO_NONE. Les valeurs supplémentaires suivantes peuvent être possibles en fonction de la plateforme:

- MQCNO_STANDARD_BINDING
- MQCNO_FASTPATH_BINDING
- MQCNO_HANDLE_SHARE_NONE
- MQCNO_HANDLE_SHARE_BLOCK
- MQCNO_HANDLE_SHARE_NO_BLOCK
- MQCNO_SERIALIZE_CONN_TAG_Q_MGR
- MQCNO_SERIALIZE_CONN_TAG_QSG
- MQCNO_RESTRICT_CONN_TAG_Q_MGR
- MQCNO_RESTRICT_CONN_TAG_QSG

ID de connexion
Identificateur unique qui permet à MQ d'identifier de manière fiable une application.

statut des connexions
TRUE en cas de connexion au gestionnaire de files d'attente. Cet attribut est en lecture seule.

Balise de connexion
Balise à associer à une connexion. Cet attribut ne peut être défini que s'il n'est pas connecté. La valeur initiale est null.

Matériel de cryptographie
Détails de configuration du matériel de cryptographie. Pour les connexions client MQ MQI.

Nom de la file d'attente de rebut
Nom de la file d'attente de rebut. Cet attribut est en lecture seule.

nom de file de transmission par défaut
Nom de la file d'attente de transmission par défaut. Cet attribut est en lecture seule.

listes de distribution
Capacité du gestionnaire de files d'attente à prendre en charge les listes de distribution.

groupe dns
Nom du groupe que le programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente doit rejoindre lors de l'utilisation de la prise en charge des services de nom de domaine dynamique Workload Manager. Cet attribut est en lecture seule.

dns wlm
Indique si le programme d'écoute TCP qui gère les transmissions entrantes pour le groupe de partage de files d'attente doit s'enregistrer auprès de Workload Manager pour Dynamic Domain Name Services. Cet attribut est en lecture seule.

premier enregistrement d'authentification
Le premier d'un ou plusieurs objets de la classe ImqAuthenticationRecord, sans ordre particulier, dans lequel la référence de connexion d'enregistrement ImqAuthenticationadresse cet objet. Pour les connexions client MQ MQI.

premier objet géré
Le premier d'un ou de plusieurs objets de la classe ImqObject, sans ordre particulier, dans lequel la **référence de connexion** ImqObject adresse cet objet. La valeur initiale est zéro.

événement d'interdiction
Contrôle les événements d'interdiction. Cet attribut est en lecture seule.

Version adresse IP
Protocole IP (IPv4 ou IPv6) à utiliser pour une connexion de canal. Cet attribut est en lecture seule.

référentiel de clés
Emplacement du fichier de base de données de clés dans lequel les clés et les certificats sont stockés. Pour les connexions client WebSphere MQ MQI.

nombre de réinitialisations de clé
Nombre d'octets non chiffrés envoyés et reçus dans une conversation SSL avant la renégociation de la clé secrète. Cet attribut s'applique uniquement aux connexions client utilisant MQCONNX. Voir aussi ssl key reset count.

Délai du programme d'écoute
Intervalle (en secondes) entre les tentatives de WebSphere MQ pour redémarrer le programme d'écoute en cas d'échec APPC ou TCP/IP. Cet attribut est en lecture seule.

événement local
Contrôle les événements locaux. Cet attribut est en lecture seule.

Evénement consignateur
Contrôle si les événements du journal de reprise sont générés. Cet attribut est en lecture seule.

Nom de groupe de LU
Nom d'unité logique générique que le programme d'écoute d'unité logique 6.2 qui gère les transmissions entrantes pour le groupe de partage de files d'attente doit utiliser. Cet attribut est en lecture seule.

Nom de LU
Nom de l'unité logique à utiliser pour les transmissions LU 6.2 sortantes. Cet attribut est en lecture seule.

Suffixe de bras lu62
Suffixe de SYS1.PARMLIB APPCPMxx, qui désigne le LUADD pour cet initiateur de canal. Cet attribut est en lecture seule.

Canaux lu62
Nombre maximal de canaux pouvant être en cours ou de clients pouvant être connectés, qui utilisent le protocole de transmission LU 6.2 . Cet attribut est en lecture seule.

nombre maximal de canaux actifs
Le nombre maximal de canaux pouvant être actifs en même temps. Cet attribut est en lecture seule.

Nombre maximal de canaux
Le nombre maximal de canaux pouvant être actifs (notamment les canaux de connexion serveur avec des clients connectés). Cet attribut est en lecture seule.

nombre maximal de descripteurs
Nombre maximal de descripteurs. Cet attribut est en lecture seule.

longueur maximale des messages
Longueur maximale possible pour tout message d'une file d'attente gérée par ce gestionnaire de files d'attente. Cet attribut est en lecture seule.

Priorité maximum
Priorité maximale des messages. Cet attribut est en lecture seule.

Nb max. messages non validés
Nombre maximal de messages non validés dans une unité ou un travail. Cet attribut est en lecture seule.

Comptabilité MQI
Ctrl la collecte des informations comptables pour les données MQI. Cet attribut est en lecture seule.

Statistiques MQI
Ctrl la collecte d'informations de contrôle de statistiques pour le gestionnaire de files d'attente. Cet attribut est en lecture seule.

port de communications sortantes maximum
Extrémité supérieure de la plage de numéros de port à utiliser lors de la liaison de canaux sortants. Cet attribut est en lecture seule.

port de communications sortantes minimum
Extrémité inférieure de la plage de numéros de port à utiliser lors de la liaison des canaux sortants. Cet attribut est en lecture seule.

mot de passe
mot de passe associé à l'ID utilisateur

événement Performances
Contrôle les événements de performances. Cet attribut est en lecture seule.

plateforme
Plateforme sur laquelle réside le gestionnaire de files d'attente. Cet attribut est en lecture seule.

Comptabilité file
Ctrl la collecte des informations comptables pour les files d'attente. Cet attribut est en lecture seule.

Ctrl file d'attente
Ctrl la collecte des données de surveillance en ligne pour les files d'attente. Cet attribut est en lecture seule.

statistiques de file d'attente
Ctrl la collecte des données de statistiques pour les files d'attente. Cet attribut est en lecture seule.

Délai de réception
Durée approximative pendant laquelle un canal de transmission de messages TCP/IP attend la réception de données, y compris les signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. Cet attribut est en lecture seule.

délai d'attente minimal de réception
Durée minimale pendant laquelle un canal TCP/IP attend de recevoir des données, y compris des signaux de présence, de la part de son partenaire, avant de revenir à l'état inactif. Cet attribut est en lecture seule.

Type de délai de réception
Qualificateur appliqué au **délai de réception**. Cet attribut est en lecture seule.

événement distant
Contrôle les événements distants. Cet attribut est en lecture seule.

Nom du référentiel
Nom de référentiel. Cet attribut est en lecture seule.

Liste de noms du référentiel
Nom de la liste de noms de référentiel. Cet attribut est en lecture seule.

nom du gestionnaire de files d'attente partagées
Indique si les MQOPENs d'une file d'attente partagée dans laquelle le nom ObjectQMgrest un autre gestionnaire de files d'attente du groupe de partage de files d'attente doivent être résolus en une ouverture de la file d'attente partagée sur le gestionnaire de files d'attente local. Cet attribut est en lecture seule.

événement ssl
Indique si des événements SSL sont générés. Cet attribut est en lecture seule.

FIPS SSL requis
Indique si seuls les algorithmes certifiés FIPS doivent être utilisés si la cryptographie est exécutée dans le logiciel WebSphere MQ . Cet attribut est en lecture seule.

Nbre avant réinit clé SSL
Nombre d'octets non chiffrés envoyés et reçus dans une conversation SSL avant la renégociation de la clé secrète. Cet attribut est en lecture seule.

événement de démarrage / arrêt
Contrôle les événements de démarrage et d'arrêt. Cet attribut est en lecture seule.

Intervalle de statistiques
Fréquence à laquelle les données de surveillance des statistiques sont écrites dans la file d'attente de surveillance. Cet attribut est en lecture seule.

Prise en charge points sync.
Disponibilité de la participation au point de synchronisation. Cet attribut est en lecture seule.

Remarque : Les unités d'oeuvre globales coordonnées par le gestionnaire de files d'attente ne sont pas prises en charge sur la plateforme IBM i .

canaux tcp
Nombre maximal de canaux pouvant être en cours ou de clients pouvant être connectés, qui utilisent le protocole de transmission TCP/IP. Cet attribut est en lecture seule.

Conservation de TC
Indique si la fonction TCP KEEPALIVE doit être utilisée pour vérifier que l'autre extrémité de la connexion est toujours disponible. Cet attribut est en lecture seule.

Nom TCP
Nom du système TCP/IP unique ou par défaut à utiliser, en fonction de la valeur de **tcp stack type**. Cet attribut est en lecture seule.

Type de pile TCP
Indique si l'initiateur de canal est autorisé à utiliser uniquement l'espace adresse TCP/IP spécifié dans **tcp name** ou s'il peut se connecter à une adresse TCP/IP sélectionnée. Cet attribut est en lecture seule.

Enregistrement trace route
Contrôle l'enregistrement des informations de trace de route. Cet attribut est en lecture seule.

Intervalle de déclenchement
Intervalle de déclenchement. Cet attribut est en lecture seule.

ID utilisateur
Sur les plateformes UNIX and Linux , ID utilisateur réel de l'application. Sur les plateformes Windows, ID utilisateur de l'application.

Constructeurs

ImqQueueManager () ;
Constructeur par défaut.

ImqQueueManager (const ImqQueueManager & ***manager*** **) ;**
Constructeur de copie. Le **statut de connexion** est FALSE.

ImqQueueManager (const char * ***nom*** **) ;**
Définit ImqObject **name** sur *name*.

Destructeurs

Lorsqu'un objet gestionnaire ImqQueueest détruit, il est automatiquement déconnecté.

Méthodes de classe (public)

comportement MQLONG statique () ;
Renvoie le **comportement**.

void setBehavior(const MQLONG *comportement* = 0) ;
Définit le **comportement**.

Méthodes d'objet (public)

void operator = (const ImqQueueManager & *mgr*) ;
Se déconnecte si nécessaire et copie les données d'instance de *mgr*. Le **statut de la connexion** est FALSE.

ImqBoolean accountingConnRemplacer (MQLONG & statint) ;
Fournit une copie de la valeur de substitution des connexions de comptabilité. Elle renvoie TRUE en cas de succès.

MQLONG accountingConnRemplacer () ;
Renvoie la valeur de substitution des connexions de comptabilité sans indication d'erreurs possibles.

ImqBoolean accountingInterval (MQLONG & statint) ;
Fournit une copie de la valeur d'intervalle de comptabilité. Elle renvoie TRUE en cas de succès.

MQLONG accountingInterval () ;
Renvoie la valeur de l'intervalle de comptabilité sans aucune indication d'erreurs possibles.

ImqBoolean activityRecording (MQLONG & rec) ;
Fournit une copie de la valeur d'enregistrement de l'activité. Elle renvoie TRUE en cas de succès.

MQLONG activityRecording () ;
Renvoie la valeur d'enregistrement d'activité sans aucune indication d'erreurs possibles.

ImqBoolean adoptNewMCACheck (MQLONG & check) ;
Fournit une copie de la valeur de vérification de l'adoption d'un nouvel agent MCA. Elle renvoie TRUE en cas de succès.

MQLONG adoptNewMCACheck () ;
Renvoie la valeur de vérification de l'adoption d'un nouvel agent MCA sans indication d'erreurs éventuelles.

ImqBoolean adoptNewMCAType (MQLONG & type) ;
Fournit une copie du nouveau type d'adoption de MCA. Elle renvoie TRUE en cas de succès.

MQLONG adoptNewMCAType () ;
Renvoie le nouveau type d'adoption d'agent MCA sans indication d'erreurs éventuelles.

QLONG authenticationType () const ;
Renvoie le type d'authentification.

void setAuthenticationType (const MQLONG type = MQCSP_AUTH_NONE) ;
Définit le type d'authentification.

ImqBoolean authorityEvent(événement MQLONG &) ;
Fournit une copie de l'état d'activation de l' **événement d'autorité**. Elle renvoie TRUE en cas de succès.

MQLONG authorityEvent() ;
Renvoie l'état d'activation de l' **événement d'autorité** sans aucune indication d'erreurs possibles.

ImqBoolean backout () ;
Annule les modifications non validées. Elle renvoie TRUE en cas de succès.

ImqBoolean begin () ;
Commence une unité de travail. Les **options de début** affectent le comportement de cette méthode. Elle renvoie TRUE en cas de réussite, mais elle renvoie également TRUE même si l'appel MQBEGIN sous-jacent renvoie MQRC_NO_EXTERNAL_PARTICIPANTS ou MQRC_PARTICIPANT_NOT_AVAILABLE (qui sont tous deux associés à MQCC_WARNING).

MQLONG beginOptions() const ;
Renvoie les **options de début**.

void setBeginOptions (const MQLONG *options* = MQBO_NONE) ;
Définit les **options de début**.

ImqBoolean bridgeEvent (MQLONG & événement) ;
Fournit une copie de la valeur d'événement de pont. Elle renvoie TRUE en cas de succès.

MQLONG bridgeEvent () ;
Renvoie la valeur d'événement de pont sans aucune indication d'erreurs possibles.

ImqBoolean channelAutoDéfinition (MQLONG & *valeur*) ;
Fournit une copie de la valeur de **définition automatique de canal** . Elle renvoie TRUE en cas de succès.

MQLONG channelAutoDéfinition () ;
Renvoie la valeur de **définition automatique de canal** sans aucune indication d'erreurs possibles.

ImqBoolean channelAutoDefinitionEvent(MQLONG & *valeur*) ;
Fournit une copie de la valeur d' **événement de définition automatique de canal** . Elle renvoie TRUE en cas de succès.

MQLONG channelAutoDefinitionEvent() ;
Renvoie la valeur de l' **événement de définition automatique de canal** sans aucune indication d'erreurs possibles.

ImqBoolean channelAutoDefinitionExit(ImqString & *nom*) ;
Fournit une copie du nom de l' **exit de définition automatique de canal** . Elle renvoie TRUE en cas de succès.

ImqString channelAutoDefinitionExit( );
Renvoie le nom de l' **exit de définition automatique de canal** sans aucune indication d'erreurs possibles.

ImqBoolean channelEvent (MQLONG & événement) ;
Fournit une copie de la valeur d'événement de canal. Elle renvoie TRUE en cas de succès.

MQLONG channelEvent() ;
Renvoie la valeur d'événement de canal sans indication d'erreurs possibles.

MQLONG channelInitiatorAdapters () ;
Renvoie la valeur des adaptateurs d'initialisateur de canal sans aucune indication d'erreurs possibles.

ImqBoolean channelInitiatorAdaptateurs (MQLONG & adaptateurs) ;
Fournit une copie de la valeur des adaptateurs d'initiateur de canal. Elle renvoie TRUE en cas de succès.

MQLONG channelInitiatorControl () ;
Renvoie la valeur de démarrage de l'initiateur de canal sans aucune indication d'erreurs possibles.

Contrôle ImqBoolean channelInitiator(MQLONG & init) ;
Fournit une copie de la valeur de démarrage du contrôle d'initiateur de canal. Elle renvoie TRUE en cas de succès.

MQLONG channelInitiatorDispatchers () ;
Renvoie la valeur des répartiteurs d'initiateur de canal sans aucune indication d'erreurs possibles.

ImqBoolean channelInitiatorDispatchers (MQLONG & répartiteurs) ;
Fournit une copie de la valeur des répartiteurs de l'initiateur de canal. Elle renvoie TRUE en cas de succès.

MQLONG channelInitiatorTraceAutoStart () ;
Renvoie la valeur de démarrage automatique de la trace de l'initiateur de canal sans aucune indication d'erreurs possibles.

ImqBoolean channelInitiatorTraceAutoStart (MQLONG & auto) ;
Fournit une copie de la valeur de démarrage automatique de la trace de l'initiateur de canal. Elle renvoie TRUE en cas de succès.

MQLONG channelInitiatorTraceTableTaille () ;
Renvoie la valeur de taille de la table de trace de l'initiateur de canal sans aucune indication d'erreurs possibles.

ImqBoolean channelInitiatorTraceTableTaille (MQLONG & size) ;
Fournit une copie de la valeur de taille de la table de trace de l'initiateur de canal. Elle renvoie TRUE en cas de succès.

ImqBoolean channelMonitoring (MQLONG & monchl) ;
Fournit une copie de la valeur de surveillance de canal. Elle renvoie TRUE en cas de succès.

MQLONG channelMonitoring () ;
Renvoie la valeur de surveillance de canal sans aucune indication d'erreurs possibles.

ImqBoolean channelReference(ImqChannel * & ***pchannel*** **) ;**
Fournit une copie de la **référence de canal**. Si la **référence de canal** n'est pas valide, définit *pchannel* sur null. Cette méthode renvoie TRUE en cas de succès.

ImqChannel * channelReference( );
Renvoie la **référence de canal** sans aucune indication d'erreurs possibles.

ImqBoolean setChannelRéférence (ImqChannel & ***channel*** **) ;**
Définit la **référence de canal**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setChannelRéférence (ImqChannel * ***channel*** **= 0) ;**
Définit ou réinitialise la **référence de canal**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean channelStatistics (MQLONG & statchl) ;
Fournit une copie de la valeur des statistiques de canal. Elle renvoie TRUE en cas de succès.

MQLONG channelStatistics () ;
Renvoie la valeur des statistiques de canal sans aucune indication d'erreurs possibles.

ImqBoolean characterSet(MQLONG & ***ccsid*** **) ;**
Fournit une copie du **jeu de caractères**. Elle renvoie TRUE en cas de succès.

MQLONG characterSet() ;
Renvoie une copie du **jeu de caractères**, sans aucune indication d'erreurs possibles.

MQLONG clientSslKeyResetNombre () const ;
Renvoie la valeur du nombre de réinitialisations de clé SSL utilisée sur les connexions client.

void setClientSslKeyResetCount(const MQLONG count) ;
Définit le nombre de réinitialisations de clé SSL utilisées sur les connexions client.

ImqBoolean clusterSenderMonitoring (MQLONG & monacls) ;
Fournit une copie de la valeur par défaut de surveillance de l'émetteur de cluster. Elle renvoie TRUE en cas de succès.

MQLONG clusterSenderMonitoring () ;
Renvoie la valeur par défaut de la surveillance de l'émetteur de cluster sans aucune indication d'erreurs possibles.

ImqBoolean clusterSenderStatistiques (MQLONG & statacls) ;
Fournit une copie de la valeur des statistiques de l'émetteur de cluster. Elle renvoie TRUE en cas de succès.

MQLONG clusterSenderStatistics () ;
Renvoie la valeur des statistiques de l'émetteur de cluster sans aucune indication d'erreurs possibles.

ImqBoolean clusterWorkloadDonnées (ImqString & ***données*** **) ;**
Fournit une copie des **données d'exit de charge de travail de cluster**. Elle renvoie TRUE en cas de succès.

ImqString clusterWorkloadData () ;
Renvoie les **données d'exit de charge de travail de cluster** sans aucune indication d'erreurs possibles.

ImqBoolean clusterWorkloadExit (ImqString & ***name*** **) ;**
Fournit une copie du **nom d'exit de charge de travail de cluster**. Elle renvoie TRUE en cas de succès.

ImqString clusterWorkloadExit () ;
Renvoie le **nom d'exit de charge de travail de cluster** sans aucune indication d'erreurs possibles.

ImqBoolean clusterWorkloadLongueur (MQLONG & ***longueur*** **) ;**
Fournit une copie de la **longueur de la charge de travail du cluster**. Elle renvoie TRUE en cas de succès.

MQLONG clusterWorkloadLength () ;
Renvoie la **longueur de la charge de travail du cluster** sans aucune indication d'erreurs possibles.

ImqBoolean clusterWorkLoadMRU (MQLONG & mru) ;
Fournit une copie de la valeur des canaux utilisés le plus récemment pour la charge de travail du cluster. Elle renvoie TRUE en cas de succès.

MQLONG clusterWorkLoadMRU () ;
Renvoie la valeur de la charge de travail de cluster la plus récemment utilisée pour les canaux sans aucune indication d'erreurs possibles.

ImqBoolean clusterWorkLoadUseQ (MQLONG & useq) ;
Fournit une copie de la valeur de la file d'attente d'utilisation de la charge de travail du cluster. Elle renvoie TRUE en cas de succès.

MQLONG clusterWorkLoadUseQ () ;
Renvoie la valeur de la file d'attente d'utilisation de la charge de travail du cluster sans aucune indication d'erreurs possibles.

ImqBoolean commandEvent (MQLONG & événement) ;
Fournit une copie de la valeur d'événement de commande. Elle renvoie TRUE en cas de succès.

MQLONG commandEvent () ;
Renvoie la valeur d'événement de commande sans aucune indication d'erreurs possibles.

ImqBoolean commandInputQueueName(ImqString & ***nom*** **) ;**
Fournit une copie de la **file d'attente d'entrée de commandes**. Elle renvoie TRUE en cas de succès.

ImqString commandInputQueueName( );
Renvoie le **nom de la file d'attente d'entrée de commandes** sans indiquer d'erreurs possibles.

ImqBoolean commandLevel(MQLONG & ***niveau*** **) ;**
Fournit une copie du **niveau de commande**. Elle renvoie TRUE en cas de succès.

MQLONG commandLevel() ;
Renvoie le **niveau de commande** sans aucune indication d'erreurs possibles.

MQLONG commandServerControl () ;
Renvoie la valeur de démarrage du serveur de commandes sans aucune indication d'erreurs possibles.

ImqBoolean commandServerControl (MQLONG & serveur) ;
Fournit une copie de la valeur de démarrage du contrôle du serveur de commandes. Elle renvoie TRUE en cas de succès.

ImqBoolean commit () ;
Valide les modifications non validées. Elle renvoie TRUE en cas de succès.

ImqBoolean connect () ;
Se connecte au gestionnaire de files d'attente avec le nom ImqObject **indiqué**, par défaut, il s'agit du gestionnaire de files d'attente local. Si vous souhaitez vous connecter à un gestionnaire de files d'attente spécifique, utilisez la méthode ImqObject **setName** avant la connexion. S'il existe une **référence de canal**, elle est utilisée pour transmettre des informations sur la définition de canal à MQCONNX dans un MQCD. ChannelType dans MQCD est défini sur MQCHT_CLNTCONN. Les informations de **référence de canal** , qui ne sont significatives que pour les connexions client, sont ignorées pour les connexions serveur. Les **options de connexion** affectent le comportement de cette

méthode. Cette méthode définit le **statut de connexion** sur TRUE en cas de réussite. Elle renvoie le nouveau statut de connexion.

S'il existe un premier enregistrement d'authentification, la chaîne d'enregistrements d'authentification est utilisée pour authentifier les certificats numériques pour les canaux client sécurisés.

Vous pouvez connecter plusieurs objets de gestionnaire ImqQueueau même gestionnaire de files d'attente. Tous utilisent le même descripteur de connexion MQHCONN et partagent la fonctionnalité UOW pour la connexion associée à l'unité d'exécution. Le premier gestionnaire ImqQueueà se connecter obtient le descripteur MQHCONN. Le dernier gestionnaire ImqQueueà se déconnecter exécute le MQDISC.

Pour un programme à unités d'exécution multiples, il est recommandé d'utiliser un objet ImqQueueManager distinct pour chaque unité d'exécution.

ImqBinary connectionId () const ;
Renvoie l'ID de connexion.

ImqBinary connectionTag () const ;
Renvoie la **balise de connexion**.

ImqBoolean setConnectionTag (const MQBYTE128 *tag* = 0) ;
Définit la **balise de connexion**. Si *tag* est zéro, efface la **balise de connexion**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setConnectionTag (const ImqBinary & *tag*) ;
Définit la **balise de connexion**. La **longueur des données** de la balise doit être égale à zéro (pour effacer la balise de connexion) ou MQ_CONN_TAG_LENGTH. Cette méthode renvoie TRUE en cas de succès.

MQLONG connectOptions() const ;
Renvoie les **options de connexion**.

void setConnectOptions (const MQLONG *options* = MQCNO_NONE) ;
Définit les **options de connexion**.

ImqBoolean connectionStatus() const ;
Renvoie le **statut de connexion**.

ImqString cryptographicHardware ( );
Renvoie le **matériel de cryptographie**.

ImqBoolean setCryptographicMatériel (const char * *matériel* = 0) ;
Définit le **matériel de cryptographie**. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean deadLetterQueueName(ImqString & *nom*) ;
Fournit une copie du **nom de la file d'attente de rebut**. Elle renvoie TRUE en cas de succès.

ImqString deadLetterQueueName( );
Renvoie une copie du **nom de la file d'attente de rebut**, sans indication d'erreurs possibles.

ImqBoolean defaultTransmissionQueueName(ImqString & *nom*) ;
Fournit une copie du **nom de la file d'attente de transmission par défaut**. Elle renvoie TRUE en cas de succès.

ImqString defaultTransmissionQueueName( );
Renvoie le **nom de la file d'attente de transmission par défaut** sans indiquer d'erreurs possibles.

ImqBoolean disconnect () ;
Se déconnecte du gestionnaire de files d'attente et définit le **statut de connexion** sur FALSE. Ferme tous les objets ImqProcess et ImqQueue associés à cet objet et arrête leur **référence de connexion** avant la déconnexion. Si plusieurs objets de gestionnaire ImqQueuesont connectés au même gestionnaire de files d'attente, seule la dernière déconnexion effectue une déconnexion physique ; les autres effectuent une déconnexion logique. Les modifications non validées sont uniquement validées lors de la déconnexion physique.

Cette méthode renvoie TRUE en cas de succès. S'il est appelé alors qu'il n'y a pas de connexion existante, le code retour est également vrai.

ImqBoolean distributionLists(MQLONG & *prise en charge*) ;
Fournit une copie de la valeur des **listes de distribution** . Elle renvoie TRUE en cas de succès.

MQLONG distributionLists() ;
Renvoie la valeur de **listes de distribution** sans aucune indication d'erreurs possibles.

ImqBoolean dnsGroup (ImqString & group) ;
Fournit une copie du nom de groupe DNS. Elle renvoie TRUE en cas de succès.

ImqString dnsGroup ( );
Renvoie le nom de groupe DNS sans aucune indication d'erreurs possibles.

ImqBoolean dnsWlm (MQLONG & wlm) ;
Fournit une copie de la valeur WLM DNS. Elle renvoie TRUE en cas de succès.

MQLONG dnsWlm () ;
Renvoie la valeur WLM DNS sans indication des erreurs possibles.

ImqAuthenticationRecord * firstAuthenticationRecord () const ;
Renvoie le **premier enregistrement d'authentification**.

void setFirstAuthenticationRecord (const ImqAuthenticationRecord * *air* = 0) ;
Définit le **premier enregistrement d'authentification**.

ImqObject * firstManagedObject () const ;
Renvoie le **premier objet géré**.

ImqBoolean inhibitEvent(MQLONG & *événement*) ;
Fournit une copie de l'état d'activation de l' **événement d'interdiction**. Elle renvoie TRUE en cas de succès.

MQLONG inhibitEvent() ;
Renvoie l'état d'activation de l' **événement d'interdiction** sans aucune indication d'erreurs possibles.

ImqBoolean ipAddressVersion (MQLONG & version) ;
Fournit une copie de la valeur de version de l'adresse IP. Elle renvoie TRUE en cas de succès.

MQLONG ipAddressVersion () ;
Renvoie la valeur de version de l'adresse IP sans aucune indication d'erreurs possibles.

ImqBoolean keepAlive (MQLONG & keepalive) ;
Fournit une copie de la valeur de signal de présence. Elle renvoie TRUE en cas de succès.

MQLONG keepAlive () ;
Renvoie la valeur de signal de présence sans aucune indication d'erreurs possibles.

ImqString keyRepository ( );
Renvoie le **référentiel de clés**.

ImqBoolean setKeyRepository (const char * *référentiel* = 0) ;
Définit le **référentiel de clés**. Elle renvoie TRUE en cas de succès.

ImqBoolean listenerTimer (MQLONG & temporisateur) ;
Fournit une copie de la valeur du temporisateur du programme d'écoute. Elle renvoie TRUE en cas de succès.

MQLONG listenerTimer () ;
Renvoie la valeur du temporisateur du programme d'écoute sans aucune indication d'erreurs possibles.

ImqBoolean localEvent(MQLONG & *événement*) ;
Fournit une copie de l'état d'activation de l' **événement local**. Elle renvoie TRUE en cas de succès.

MQLONG localEvent() ;
Renvoie l'état d'activation de l' **événement local** sans aucune indication d'erreurs possibles.

ImqBoolean loggerEvent (MQLONG & nombre) ;
Fournit une copie de la valeur d'événement du consignateur. Elle renvoie TRUE en cas de succès.

MQLONG loggerEvent () ;
Renvoie la valeur d'événement du consignateur sans aucune indication d'erreurs possibles.

ImqBoolean luGroupNom (ImqString & name) ;.
Fournit une copie du nom du groupe de LU. Renvoie TRUE si l'opération aboutit

ImqString luGroupNom () ;
Renvoie le nom du groupe de LU sans indiquer d'erreurs possibles.

ImqBoolean lu62ARMSuffix (ImqString & suffixe) ;
Fournit une copie du suffixe ARM LU62 . Elle renvoie TRUE en cas de succès.

ImqString lu62ARMSuffix ( );
Renvoie le suffixe ARM LU62 sans indication des erreurs possibles

ImqBoolean luName (ImqString & name) ;
Fournit une copie du nom de LU. Elle renvoie TRUE en cas de succès.

ImqString luName ( );
Renvoie le nom d'unité logique sans indication d'erreurs possibles.

ImqBoolean maximumActiveCanaux (MQLONG & canaux) ;
Fournit une copie de la valeur du nombre maximal de canaux actifs. Elle renvoie TRUE en cas de succès.

MQLONG maximumActiveChannels () ;
Renvoie la valeur du nombre maximal de canaux actifs sans aucune indication d'erreurs possibles.

ImqBoolean maximumCurrentCanaux (MQLONG & canaux) ;
Fournit une copie de la valeur du nombre maximal de canaux en cours. Elle renvoie TRUE en cas de succès.

MQLONG maximumCurrentChannels () ;
Renvoie la valeur du nombre maximal de canaux en cours sans aucune indication d'erreurs possibles.

ImqBoolean maximumHandles(MQLONG & ***nombre*** **) ;**
Fournit une copie des **descripteurs maximum**. Elle renvoie TRUE en cas de succès.

MQLONG maximumHandles() ;
Renvoie le **nombre maximal de descripteurs** sans aucune indication d'erreurs possibles.

ImqBoolean maximumLu62Channels (MQLONG & canaux) ;
Fournit une copie de la valeur maximale des canaux LU62 . Elle renvoie TRUE en cas de succès.

MQLONG maximumLu62Channels () ;.
Renvoie la valeur maximale des canaux LU62 sans indication d'erreurs possibles

ImqBoolean maximumMessageLongueur (MQLONG & ***longueur*** **) ;**
Fournit une copie de la **longueur maximale de message**. Elle renvoie TRUE en cas de succès.

MQLONG maximumMessageLongueur () ;
Renvoie la **longueur maximale de message** sans aucune indication d'erreurs possibles.

ImqBoolean maximumPriority(MQLONG & ***priorité*** **) ;**
Fournit une copie de la **priorité maximale**. Elle renvoie TRUE en cas de succès.

MQLONG maximumPriority() ;
Renvoie une copie de la **priorité maximale**, sans indication d'erreurs possibles.

ImqBoolean maximumTcpCanaux (MQLONG & canaux) ;
Fournit une copie de la valeur du nombre maximal de canaux TCP. Elle renvoie TRUE en cas de succès.

MQLONG maximumTcpChannels () ;
Renvoie la valeur du nombre maximal de canaux TCP sans indication d'erreurs possibles.

ImqBoolean maximumUncommittedMessages (MQLONG & ***nombre*** **) ;**
Fournit une copie du **nombre maximal de messages non validés**. Elle renvoie TRUE en cas de succès.

MQLONG maximumUncommittedMessages () ;
Renvoie le **nombre maximal de messages non validés** sans aucune indication d'erreurs possibles.

ImqBoolean mqiAccounting (MQLONG & statint) ;
Fournit une copie de la valeur de comptabilité MQI. Elle renvoie TRUE en cas de succès.

MQLONG mqiAccounting () ;
Renvoie la valeur de comptabilité MQI sans aucune indication d'erreurs possibles.

ImqBoolean mqiStatistics (MQLONG & statmqi) ;
Fournit une copie de la valeur des statistiques MQI. Elle renvoie TRUE en cas de succès.

MQLONG mqiStatistics () ;
Renvoie la valeur des statistiques MQI sans aucune indication d'erreurs possibles.

ImqBoolean outboundPortMax (MQLONG & max) ;
Fournit une copie de la valeur maximale du port de communications sortantes. Elle renvoie TRUE en cas de succès.

MQLONG outboundPortMax () ;
Renvoie la valeur maximale du port de communications sortantes sans aucune indication d'erreurs possibles.

ImqBoolean outboundPortMin (MQLONG & min) ;
Fournit une copie de la valeur minimale du port de communications sortantes. Elle renvoie TRUE en cas de succès.

MQLONG outboundPortMin () ;
Renvoie la valeur minimale du port de communications sortantes sans aucune indication d'erreurs possibles.

Mot de passe ImqBinary () const ;
Renvoie le mot de passe utilisé sur les connexions client.

ImqBoolean setPassword (const ImqString & password) ;
Définit le mot de passe utilisé sur les connexions client.

ImqBoolean setPassword (const car * = 0 mot de passe) ;
Définit le mot de passe utilisé sur les connexions client.

ImqBoolean setPassword (const ImqBinary & password) ;
Définit le mot de passe utilisé sur les connexions client.

ImqBoolean performanceEvent(MQLONG & ***événement*** **) ;**
Fournit une copie de l'état d'activation de l' **événement de performance**. Elle renvoie TRUE en cas de succès.

MQLONG performanceEvent() ;
Renvoie l'état d'activation de l' **événement de performance** sans aucune indication d'erreurs possibles.

Plateforme ImqBoolean (MQLONG & ***plateforme*** **) ;**
Fournit une copie de la plateforme . Elle renvoie TRUE en cas de succès.

plateforme MQLONG () ;
Renvoie la **plateforme** sans indication des erreurs possibles.

ImqBoolean queueAccounting (MQLONG & acctq) ;
Fournit une copie de la valeur de comptabilité de la file d'attente. Elle renvoie TRUE en cas de succès.

MQLONG queueAccounting () ;
Renvoie la valeur de comptabilité de la file d'attente sans indication d'erreurs possibles.

ImqBoolean queueMonitoring (MQLONG & monq) ;
Fournit une copie de la valeur de surveillance de file d'attente. Elle renvoie TRUE en cas de succès.

MQLONG queueMonitoring () ;
Renvoie la valeur de surveillance de file d'attente sans aucune indication d'erreurs possibles.

ImqBoolean queueStatistics (MQLONG & statq) ;
Fournit une copie de la valeur des statistiques de file d'attente. Elle renvoie TRUE en cas de succès.

MQLONG queueStatistics () ;
Renvoie la valeur des statistiques de file d'attente sans indication d'erreurs possibles.

ImqBoolean receiveTimeout (MQLONG & délai d'attente) ;
Fournit une copie de la valeur de délai d'attente de réception. Elle renvoie TRUE en cas de succès.

MQLONG receiveTimeout () ;
Renvoie la valeur du délai d'attente de réception sans aucune indication d'erreurs possibles.

ImqBoolean receiveTimeoutMin (MQLONG & min) ;
Fournit une copie de la valeur de délai d'attente de réception minimale. Elle renvoie TRUE en cas de succès.

MQLONG receiveTimeoutMin () ;
Renvoie la valeur de délai d'attente de réception minimale sans aucune indication d'erreurs possibles.

ImqBoolean receiveTimeoutType (MQLONG & type) ;
Fournit une copie du type de délai d'attente de réception. Elle renvoie TRUE en cas de succès.

MQLONG receiveTimeoutType () ;
Renvoie le type de délai d'attente de réception sans indication d'erreurs possibles.

ImqBoolean remoteEvent(MQLONG & ***événement*** **) ;**
Fournit une copie de l'état d'activation de l' **événement distant**. Elle renvoie TRUE en cas de succès.

MQLONG remoteEvent() ;
Renvoie l'état d'activation de l' **événement distant** sans aucune indication d'erreurs possibles.

ImqBoolean repositoryName(ImqString & ***nom*** **) ;**
Fournit une copie du **nom de référentiel**. Elle renvoie TRUE en cas de succès.

ImqString repositoryName( );
Renvoie le **nom de référentiel** sans aucune indication d'erreurs possibles.

ImqBoolean repositoryNamelistNom (ImqString & ***nom*** **) ;**
Fournit une copie du **nom de liste de noms de référentiel**. Elle renvoie TRUE en cas de succès.

ImqString repositoryNamelistNom () ;
Renvoie une copie du **nom de la liste de noms de référentiel** sans indiquer d'erreurs possibles.

ImqBoolean sharedQueueQueueManagerNom (MQLONG & nom) ;
Fournit une copie de la valeur du nom du gestionnaire de files d'attente partagées. Elle renvoie TRUE en cas de succès.

MQLONG sharedQueueQueueManagerNom () ;
Renvoie la valeur du nom du gestionnaire de files d'attente partagées sans indiquer les erreurs possibles.

ImqBoolean sslEvent (MQLONG & événement) ;
Fournit une copie de la valeur d'événement SSL. Elle renvoie TRUE en cas de succès.

MQLONG sslEvent () ;
Renvoie la valeur d'événement SSL sans aucune indication d'erreurs possibles.

ImqBoolean sslFips (MQLONG & sslfips) ;
Fournit une copie de la valeur SSL FIPS. Elle renvoie TRUE en cas de succès.

MQLONG sslFips () ;
Renvoie la valeur SSL FIPS sans indication d'erreurs possibles.

ImqBoolean sslKeyResetCount (MQLONG & count) ;
Fournit une copie de la valeur du nombre de réinitialisations de clé SSL. Elle renvoie TRUE en cas de succès.

MQLONG sslKeyResetCount () ;
Renvoie la valeur du nombre de réinitialisations de clé SSL sans indication d'erreurs possibles.

ImqBoolean startStopEvénement (MQLONG & ***événement*** **) ;**
Fournit une copie de l'état d'activation de l' **événement de démarrage / arrêt**. Elle renvoie TRUE en cas de succès.

MQLONG startStopEvénement () ;
Renvoie l'état d'activation de l' **événement de démarrage / arrêt** sans aucune indication d'erreurs possibles.

ImqBoolean statisticsInterval (MQLONG & statint) ;
Fournit une copie de la valeur d'intervalle de statistiques. Elle renvoie TRUE en cas de succès.

MQLONG statisticsInterval () ;
Renvoie la valeur d'intervalle des statistiques sans aucune indication d'erreurs possibles.

ImqBoolean syncPointDisponibilité (MQLONG & ***sync*** **) ;**
Fournit une copie de la valeur **syncpoint availability** . Elle renvoie TRUE en cas de succès.

MQLONG syncPointAvailability () ;
Renvoie une copie de la valeur **syncpoint availability** , sans aucune indication d'erreurs possibles.

ImqBoolean tcpName (ImqString & nom) ;
Fournit une copie du nom du système TCP. Elle renvoie TRUE en cas de succès.

ImqString tcpName ( );
Renvoie le nom du système TCP sans aucune indication d'erreurs possibles.

ImqBoolean tcpStackType (MQLONG & type) ;
Fournit une copie du type de pile TCP. Elle renvoie TRUE en cas de succès.

MQLONG tcpStackType () ;
Renvoie le type de pile TCP sans indication d'erreurs possibles.

ImqBoolean traceRouteRecording (MQLONG & routerec) ;
Fournit une copie de la valeur d'enregistrement de la route de trace. Elle renvoie TRUE en cas de succès.

MQLONG traceRouteEnregistrement () ;
Renvoie la valeur d'enregistrement de la route de trace sans aucune indication d'erreurs possibles.

ImqBoolean triggerInterval(MQLONG & ***intervalle*** **) ;**
Fournit une copie de l' **intervalle de déclenchement**. Elle renvoie TRUE en cas de succès.

MQLONG triggerInterval() ;
Renvoie l' **intervalle de déclenchement** sans aucune indication d'erreurs possibles.

ImqBinary userId () const ;
Renvoie l'ID utilisateur utilisé sur les connexions client.

ImqBoolean setUserID (const ImqString & id) ;
Définit l'ID utilisateur utilisé sur les connexions client.

ImqBoolean setUserID (const car * = 0 id) ;
Définit l'ID utilisateur utilisé sur les connexions client.

ImqBoolean setUserId (const ImqBinary & id) ;
Définit l'ID utilisateur utilisé sur les connexions client.

Méthodes d'objet (protégées)

void setFirstManagedObject(const ImqObject * ***objet*** **= 0) ;**
Définit le **premier objet géré**.

Données d'objet (protégées)

MQHCONN ***ohconn***
Descripteur de connexion WebSphere MQ (significatif uniquement lorsque le **statut de connexion** est TRUE).

Codes raison

- MQRC_ATTRIBUT_LOCKED
- MQRC_ENVIRONMENT_ERROR
- MQRC_FUNCTION_NOT_SUPPORTED
- ERREUR MQRC_REFERENCE_ERROR
- (codes anomalie pour MQBACK)
- (codes anomalie pour MQBEGIN)

- (codes anomalie pour MQCMIT)
- (codes anomalie pour MQCONNX)
- (codes anomalie pour MQDISC)
- (codes anomalie pour MQCONN)

Classe C++ d'en-tête ImqReference

Cette classe encapsule les fonctions de la structure de données MQRMH.

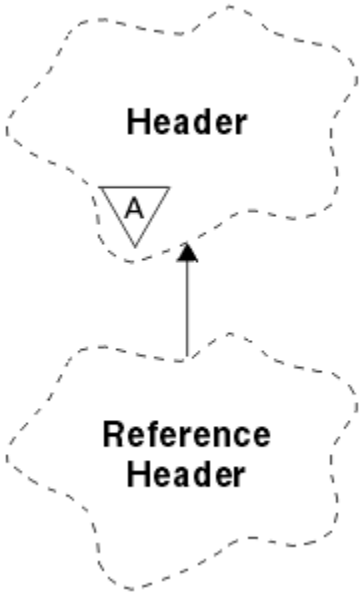


Figure 66. Classe d'en-tête ImqReference

Cette classe est liée aux appels MQI répertoriés dans le «Référence croisée d'en-tête ImqReference», à la page 1342.

Attributs d'objet

Environnement cible
Environnement de la destination. La valeur initiale est une chaîne nulle.

nom de destination
Nom de la destination de données. La valeur initiale est une chaîne nulle.

ID instance
Identificateur de l'instance. Une valeur binaire (MQBYTE24) de longueur MQ_OBJECT_INSTANCE_ID_LENGTH. La valeur initiale est MQOII_NONE.

longueur logique
Longueur logique, ou prévue, des données de message qui suivent cet en-tête. La valeur initiale est zéro.

décalage logique
Décalage logique des données de message qui suivent, à interpréter dans le contexte de l'ensemble des données, à la destination finale. La valeur initiale est zéro.

décalage logique 2
Extension de rang élevé du **décalage logique**. La valeur initiale est zéro.

Type de référence
Type de référence. La valeur initiale est une chaîne nulle.

Environnement source
Environnement de la source. La valeur initiale est une chaîne nulle.

Nom de source
Nom de la source de données. La valeur initiale est une chaîne nulle.

Constructeurs

ImqReferenceEn-tête () ;
Constructeur par défaut.

ImqReferenceHeader (const ImqReferenceHeader & ***header*** **) ;**
Constructeur de copie.

Méthodes ImqItem surchargées

virtual ImqBoolean copyOut(ImqMessage & ***msg*** **) ;**
Insère une structure de données MQRMH dans la mémoire tampon de message au début, en déplaçant davantage les données de message existantes, et définit le format *msg* sur MQFMT_REF_MSG_HEADER.

Voir la description de la méthode de classe ImqHeader sur «Classe C++ ImqHeader», à la page 1371 pour plus de détails.

virtuel ImqBoolean pasteIn(ImqMessage & ***msg*** **) ;**
Lit une structure de données MQRMH à partir de la mémoire tampon de messages.

Pour que l'opération aboutisse, le format ImqMessage doit être MQFMT_REF_MSG_HEADER.

Voir la description de la méthode de classe ImqHeader sur «Classe C++ ImqHeader», à la page 1371 pour plus de détails.

Méthodes d'objet (public)

void operator = (const ImqReferenceHeader & ***header*** **) ;**
Copie les données d'instance à partir de l'en-tête , en remplaçant les données d'instance existantes.

ImqString destinationEnvironment() const ;
Renvoie une copie de l' **environnement de destination**.

void setDestinationEnvironment(const char * ***environment*** **= 0) ;**
Définit l' **environnement de destination**.

ImqString destinationName() const ;
Renvoie une copie du **nom de destination**.

void setDestinationName(const char * ***name*** **= 0) ;**
Définit le **nom de la destination**.

ImqBinary instanceId() const ;
Renvoie une copie de l' **ID d'instance**.

ImqBoolean setInstanceId(const ImqBinary & ***id*** **) ;**
Définit l' **ID d'instance**. La **longueur de données** du *jeton* doit être 0 ou MQ_OBJECT_INSTANCE_ID_LENGTH. Cette méthode renvoie TRUE en cas de succès.

void setInstanceId(const MQBYTE24 *id* = 0) ;
Définit l' **ID d'instance**. *id* peut être égal à zéro, ce qui revient à spécifier MQOII_NONE. Si *id* est différent de zéro, il doit traiter les octets MQ_OBJECT_INSTANCE_ID_LENGTH des données binaires. Lorsque vous utilisez des valeurs prédéfinies telles que MQOII_NONE, vous devrez peut-être effectuer un transtypage pour garantir une correspondance de signature, par exemple (MQBYTE *) MQOII_NONE.

MQLONG logicalLength() const ;
Renvoie la **longueur logique**.

void setLogicalLength(const MQLONG *longueur*) ;
Définit la **longueur logique**.

MQLONG logicalOffset() const ;
Renvoie le **décalage logique**.

void setLogicalOffset(const MQLONG *décalage*) ;
Définit le **décalage logique**.

MQLONG logicalOffset2() const ;
Renvoie le **décalage logique 2**.

void setLogicalOffset2(const MQLONG *décalage*) ;
Définit le **décalage logique 2**.

ImqString referenceType() const ;
Renvoie une copie du **type de référence**.

void setReferenceType(const char * *name* = 0) ;
Définit le **type de référence**.

ImqString sourceEnvironment() const ;
Renvoie une copie de l' **environnement source**.

void setSourceEnvironment(const char * *environment* = 0) ;
Définit l' **environnement source**.

ImqString sourceName() const ;
Renvoie une copie du **nom de la source**.

void setSourceName(const char * *name* = 0) ;
Définit le **nom de la source**.

Données d'objet (protégées)

MQRMH *omqrmh*
Structure de données MQRMH.

Codes raison

- MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_STRUC_LENGTH_ERROR
- MQRC_STRUC_ID_ERROR
- MQRC_DONNÉES_INSUFFISANTES
- MQRC_INCONSISTENT_FORMAT
- ERREUR_CODAGE_MQRC_ERREUR

Classe C++ ImqString

Cette classe fournit un stockage et une manipulation de chaînes de caractères pour les chaînes à terminaison nulle.

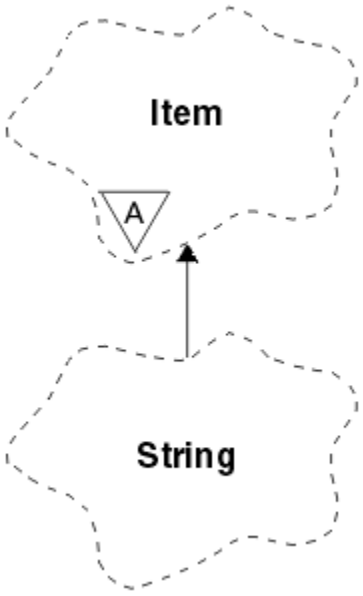


Figure 67. Classe ImqString

Utilisez un ImqString à la place d'un **char *** dans la plupart des cas où un paramètre appelle un **char ***.

- «Attributs d'objet», à la page 1429
- «Constructeurs», à la page 1429
- «Méthodes de classe (public)», à la page 1430
- «Méthodes ImqItem surchargées», à la page 1430
- «Méthodes d'objet (public)», à la page 1430
- «Méthodes d'objet (protégées)», à la page 1433
- «Codes raison», à la page 1433

Attributs d'objet

caractères
Caractères du **stockage** qui précèdent une valeur nulle de fin.

longueur
Nombre d'octets dans les **caractères**. S'il n'y a pas de **stockage**, la **longueur** est égale à zéro. La valeur initiale est zéro.

stockage
Tableau volatil d'octets de taille arbitraire. Une valeur nulle de fin doit toujours être présente dans le **stockage** après les **caractères**, afin que la fin des **caractères** puisse être détectée. Les méthodes garantissent que cette situation est maintenue, mais s'assurent, lors de la définition directe des octets dans le tableau, qu'une valeur nulle de fin existe après la modification. Initialement, il n'existe pas d'attribut **storage** .

Constructeurs

ImqString( );
Constructeur par défaut.

ImqString(const ImqString & *chaîne*) ;
Constructeur de copie.

ImqString(const car *c*) ;
Les **caractères** comprennent *c*.

ImqString(const char * *texte*) ;
Les **caractères** sont copiés à partir de *texte*.

ImqString(const void * *buffer*, const size_t *longueur*) ;
Copie *longueur* octets à partir de la *mémoire tampon* et les affecte aux caractères . La substitution est effectuée pour tous les caractères nuls copiés. Le caractère de substitution est un point (.). Aucune attention particulière n'est accordée aux autres caractères non imprimables ou non affichables copiés.

Méthodes de classe (public)

static ImqBoolean copy (char * *destination-buffer*, const size_t *length*, const char * *source-buffer*, const char *pad* = 0) ;
Copie jusqu'à *longueur* octets depuis *source-buffer* vers *destination-buffer*. Si le nombre de caractères dans *source-buffer* est insuffisant, l'espace restant dans *destination-buffer* est rempli avec des caractères *pad* . *source-buffer* peut avoir la valeur zéro. *destination-buffer* peut être égal à zéro si *length* est également égal à zéro. Tous les codes d'erreur sont perdus. Cette méthode renvoie TRUE en cas de succès.

static ImqBoolean copy (char * *destination-buffer*, const size_t *length*, const char * *source-buffer*, ImqError & *error-object*, const char *pad* = 0) ;
Copie jusqu'à *longueur* octets depuis *source-buffer* vers *destination-buffer*. Si le nombre de caractères dans *source-buffer* est insuffisant, l'espace restant dans *destination-buffer* est rempli avec des caractères *pad* . *source-buffer* peut avoir la valeur zéro. *destination-buffer* peut être égal à zéro si *length* est également égal à zéro. Tous les codes d'erreur sont définis dans *error-object*. Cette méthode renvoie TRUE en cas de succès.

Méthodes ImqItem surchargées

virtual ImqBoolean copyOut(ImqMessage & *msg*) ;
Copie les **caractères** dans la mémoire tampon de messages, en remplaçant tout contenu existant. Définit le *msg* **format** sur MQFMT_STRING.

Pour plus de détails, voir la description de la méthode de la classe parent.

virtuel ImqBoolean pasteIn(ImqMessage & *msg*) ;
Définit les **caractères** en transférant les données restantes de la mémoire tampon de messages, en remplaçant les **caractères**existants.

Pour que l'opération aboutisse, le **codage** de l'objet *msg* doit être MQENC_NATIVE. Extrayez les messages avec MQGMO_CONVERT en MQENC_NATIVE.

Pour que l'opération aboutisse, le **format** ImqMessage doit être MQFMT_STRING.

Pour plus de détails, voir la description de la méthode de la classe parent.

Méthodes d'objet (public)

char & operator [ ] (const size_t *décalage*) const ;
Fait référence au caractère au niveau du décalage *décalage* dans la **mémoire**. Vérifiez que l'octet approprié existe et qu'il est adressable.

Opérateur ImqString () (const size_t *décalage*, const size_t *longueur* = 1) const ;
Renvoie une sous-chaîne en copiant les octets des **caractères** à partir de *décalage*. Si *length* est égal à zéro, renvoie le reste des **caractères**. Si la combinaison de *offset* et *length* ne génère pas de référence dans les **caractères**, renvoie une valeur ImqStringvide.

void operator = (const ImqString & *chaîne*) ;
Copie les données d'instance à partir de *chaîne*, en remplaçant les données d'instance existantes.

Opérateur ImqString + (const car *c*) const ;
Renvoie le résultat de l'ajout de *c* aux caractères .

ImqString operator + (const char * *texte*) const ;
Renvoie le résultat de l'ajout de *texte* aux caractères . Ceci peut également être inversé. Exemple :

```
strOne + "string two" ;
"string one" + strTwo ;
```

Remarque : Bien que la plupart des compilateurs acceptent **strOne + "string two" ;** Microsoft Visual C++ requiert **strOne + (char *) "string two" ;**

Opérateur ImqString + (const ImqString & *string1*) const ;
Renvoie le résultat de l'ajout de *string1* aux caractères .

Opérateur ImqString + (const double *nombre*) const ;
Renvoie le résultat de l'ajout de *nombre* aux **caractères** après la conversion en texte.

Opérateur ImqString + (const long *nombre*) const ;
Renvoie le résultat de l'ajout de *nombre* aux **caractères** après la conversion en texte.

void operator + = (const char *c*) ;
Ajoute *c* aux caractères .

void operator + = (const char * *texte*) ;
Ajoute *texte* aux **caractères**.

void operator + = (const ImqString & *chaîne*) ;
Ajoute *chaîne* aux **caractères**.

void operator + = (const double *nombre*) ;
Ajoute *nombre* aux **caractères** après la conversion en texte.

void operator + = (const long *nombre*) ;
Ajoute *nombre* aux **caractères** après la conversion en texte.

caractère opérateur * () const ;
Renvoie l'adresse du premier octet du **stockage**. Cette valeur peut être égale à zéro et est volatile. Utilisez cette méthode uniquement à des fins de lecture seule.

ImqBoolean opérateur < (const ImqString & *chaîne*) const ;
Compare les **caractères** à ceux de *chaîne* à l'aide de la méthode **compare** . Le résultat est TRUE si inférieur à et FALSE si supérieur ou égal à.

ImqBoolean operator > (const ImqString & *chaîne*) const ;
Compare les **caractères** à ceux de *chaîne* à l'aide de la méthode **compare** . Le résultat est TRUE si supérieur à et FALSE si inférieur ou égal à.

ImqBoolean operator < = (const ImqString & *chaîne*) const ;
Compare les **caractères** à ceux de *chaîne* à l'aide de la méthode **compare** . Le résultat est TRUE si inférieur ou égal à et FALSE si supérieur à.

ImqBoolean operator > = (const ImqString & *chaîne*) const ;
Compare les **caractères** à ceux de *chaîne* à l'aide de la méthode **compare** . Le résultat est TRUE si supérieur ou égal à et FALSE si inférieur à.

opérateur ImqBoolean == (const ImqString & *chaîne*) const ;
Compare les **caractères** à ceux de *chaîne* à l'aide de la méthode **compare** . Elle renvoie TRUE ou FALSE.

Opérateur ImqBoolean ! = (const ImqString & *string*) const ;
Compare les **caractères** à ceux de *chaîne* à l'aide de la méthode **compare** . Elle renvoie TRUE ou FALSE.

short compare (const ImqString & *chaîne*) const ;
Compare les **caractères** à ceux de *chaîne*. Le résultat est zéro si les **caractères** sont égaux, négatif si inférieur à et positif si supérieur à. La comparaison est sensible à la casse. Une valeur ImqString nulle est considérée comme inférieure à une valeur ImqStringnon nulle.

ImqBoolean copyOut(car * *buffer*, const size_t *longueur*, const char *pad* = 0) ;

Copie jusqu'à *longueur* octets des **caractères** vers la *mémoire tampon*. Si le nombre de **caractères** est insuffisant, remplit l'espace restant dans *buffer* avec des caractères *pad* . *buffer* peut être égal à zéro si *length* est également égal à zéro. Elle renvoie TRUE en cas de succès.

size_t copyOut(long & *nombre*) const ;

Définit *nombre* à partir des **caractères** après la conversion à partir du texte et renvoie le nombre de caractères impliqués dans la conversion. Si la valeur est zéro, aucune conversion n'a été effectuée et *nombre* n'est pas défini. Une séquence de caractères convertible doit commencer par les valeurs suivantes:

```
<blank(s)>
<+|->
digit(s)
```

size_t copyOut(ImqString & *jeton*, const car *c* ='') const ;

Si les **caractères** contiennent un ou plusieurs caractères différents de *c*, identifie un jeton comme la première séquence contiguë de ces caractères. Dans ce cas, *token* est défini sur cette séquence et la valeur renvoyée est la somme du nombre de caractères de début *c* et du nombre d'octets dans la séquence. Sinon, renvoie zéro et ne définit pas *token*.

size_t cutOut(long & *nombre*) ;

Définit *nombre* comme pour la méthode **copy** , mais supprime également de **caractères** le nombre d'octets indiqué par la valeur de retour. Par exemple, la chaîne illustrée dans l'exemple suivant peut être coupée en trois nombres à l'aide de **cutOut**(*nombre*) trois fois:

```
strNumbers = "-1 0     +55 "

while ( strNumbers.cutOut( number ) );
number becomes -1, then 0, then 55
leaving strNumbers == " "
```

size_t cutOut(ImqString & *jeton*, const car *c* ='"

Définit *token* comme pour la méthode **copyOut** et supprime de **caractères** les caractères *strToken* ainsi que les caractères *c* qui précèdent les caractères *token* . Si *c* n'est pas un blanc, supprime les caractères *c* qui succèdent directement aux caractères *token* . Renvoie le nombre de caractères supprimés. Par exemple, la chaîne illustrée dans l'exemple suivant peut être découpée en trois jetons à l'aide de **cutOut**(*jeton*) trois fois:

```
strText = "  Program Version 1.1  "

while ( strText.cutOut( token ) );

// token becomes "Program", then "Version",
// then "1.1" leaving strText == "  "
```

L'exemple suivant montre comment analyser un nom de chemin DOS:

```
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 & *chaîne*) ;

Recherche une correspondance exacte pour *chaîne* n'importe où dans les **caractères**. Si aucune correspondance n'est trouvée, elle retourne FALSE. Sinon, elle renvoie TRUE. Si *string* est null, elle renvoie TRUE.

ImqBoolean find (const ImqString & ***chaîne*****, size_t &** ***décalage*** **) ;**
Recherche une correspondance exacte pour *chaîne* quelque part dans les **caractères** à partir du décalage *décalage* . Si *string* est null, elle renvoie TRUE sans mettre à jour *offset*. Si aucune correspondance n'est trouvée, elle renvoie FALSE (la valeur de *offset* peut avoir été augmentée). Si une correspondance est trouvée, elle renvoie TRUE et met à jour *offset* avec le décalage de *string* dans les caractères .

size_t length () const ;
Renvoie la valeur **length**.

ImqBoolean pasteIn(const double ***nombre*****, const car *** ***format*** **= "%f") ;**
Ajoute *nombre* aux **caractères** après la conversion en texte. Elle renvoie TRUE en cas de succès.

La spécification *format* est utilisée pour formater la conversion en virgule flottante. S'il est spécifié, il doit être adapté à une utilisation avec **printf** et des nombres à virgule flottante, par exemple **%.3f**.

ImqBoolean pasteIn(const long ***nombre*** **) ;**
Ajoute *nombre* aux **caractères** après la conversion en texte. Elle renvoie TRUE en cas de succès.

ImqBoolean pasteIn(const void * ***buffer*****, const size_t** ***longueur*** **) ;**
Ajoute *longueur* octets de la *mémoire tampon* aux **caractères**et ajoute une valeur null de fin finale. Remplace tous les caractères nuls copiés. Le caractère de substitution est un point (.). Aucune attention particulière n'est accordée aux autres caractères non imprimables ou non affichables copiés. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean set (const char * ***buffer*****, const size_t** ***longueur*** **) ;**
Définit les **caractères** à partir d'une zone de caractères de longueur fixe, qui peut contenir une valeur null. Ajoute une valeur null aux caractères de la zone de longueur fixe si nécessaire. Cette méthode renvoie TRUE en cas de succès.

ImqBoolean setStorage(const size_t ***longueur*** **) ;**
Alloue (ou réalloue) le **stockage**. Conserve tous les **caractères**d'origine, y compris les caractères nuls de fin, s'il y a encore de la place pour eux, mais n'initialise pas de stockage supplémentaire.

Cette méthode renvoie TRUE en cas de succès.

size_t storage () const ;
Renvoie le nombre d'octets dans le **stockage**.

size_t stripLeading(const car ***c*** **='') ;**
Supprime les caractères de début *c* des **caractères** et renvoie le nombre supprimé.

size_t stripTrailing(const car ***c*** **='') ;**
Supprime les caractères de fin *c* des **caractères** et renvoie le nombre supprimé.

ImqString upperCase() const ;
Renvoie une copie en majuscules des **caractères**.

Méthodes d'objet (protégées)

ImqBoolean assign(const ImqString & ***chaîne*** **) ;**
Equivalent à la méthode **operator =** équivalente, mais non virtuelle. Elle renvoie TRUE en cas de succès.

Codes raison

- MQRC_DATA_TRONQUÉ
- POINT_NULL_MQRC_NULL
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_BUFFER_ERROR
- MQRC_INCONSISTENT_FORMAT

Classe C++ ImqTrigger

Cette classe encapsule la structure de données MQTM (message de déclenchement).

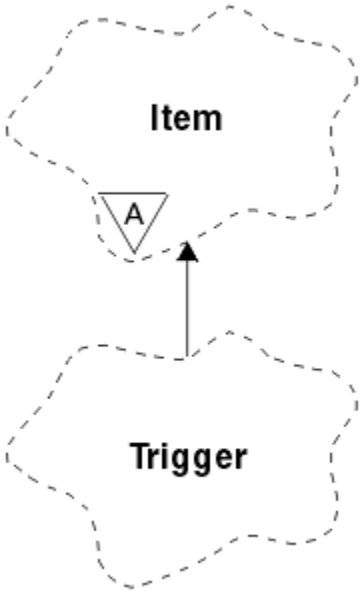


Figure 68. Classe ImqTrigger

Les objets de cette classe sont généralement utilisés par un programme de moniteur de déclenchement. La tâche d'un programme de moniteur de déclenchement consiste à attendre ces messages spécifiques et à les traiter afin de s'assurer que d'autres applications WebSphere MQ sont démarrées lorsque des messages les attendent.

Pour un exemple d'utilisation, voir l'exemple de programme IMQSTRG.

- «Attributs d'objet», à la page 1434
- «Constructeurs», à la page 1435
- «Méthodes ImqItem surchargées», à la page 1435
- «Méthodes d'objet (public)», à la page 1435
- «Données d'objet (protégées)», à la page 1436
- «Codes raison», à la page 1436

Attributs d'objet

Application ID
Identité de l'application qui a envoyé le message. La valeur initiale est une chaîne nulle.

Type d'Application
Type d'application ayant envoyé le message. La valeur initiale est zéro. Les valeurs supplémentaires suivantes sont possibles:

- MQAT_AIX
- MQAT_CICS
- MQAT_DOS
- MQAT_IMS
- MQAT_MVS
- MQAT_NOTES_AGENT
- MQAT_OS2
- MQAT_OS390

- MQAT_OS400
- MQAT_UNIX
- MQAT_WINDOWS
- MQAT_WINDOWS_NT
- MQAT_USER_FIRST
- MQAT_USER_LAST

Données d'environnement
Données d'environnement du processus. La valeur initiale est une chaîne nulle.

Nom de processus
Nom du processus. La valeur initiale est une chaîne nulle.

Nom de la file d'attente
Nom de la file d'attente à démarrer. La valeur initiale est une chaîne nulle.

Données de déclenchement
Données de déclenchement pour le processus. La valeur initiale est une chaîne nulle.

données utilisateur
Données utilisateur du processus. La valeur initiale est une chaîne nulle.

Constructeurs

ImqTrigger( );
Constructeur par défaut.

ImqTrigger(const ImqTrigger & *déclencheur*) ;
Constructeur de copie.

Méthodes ImqItem surchargées

virtual ImqBoolean copyOut(ImqMessage & *msg*) ;
Ecrit une structure de données MQTM dans la mémoire tampon de messages, en remplaçant tout contenu existant. Définit le *msg* **format** sur MQFMT_TRIGGER.

Pour plus de détails, voir la description de la méthode de classe ImqItem à l'adresse «Classe C++ ImqItem», à la page 1375 .

virtuel ImqBoolean pasteIn(ImqMessage & *msg*) ;
Lit une structure de données MQTM à partir de la mémoire tampon de messages.

Pour que l'opération aboutisse, le **format** ImqMessage doit être MQFMT_TRIGGER.

Pour plus de détails, voir la description de la méthode de classe ImqItem à l'adresse «Classe C++ ImqItem», à la page 1375 .

Méthodes d'objet (public)

void operator = (const ImqTrigger & *trigger*) ;
Copie les données d'instance à partir du *déclencheur*, en remplaçant les données d'instance existantes.

ImqString applicationId() const ;
Renvoie une copie de l' **ID application**.

void setApplicationId(const char * *id*) ;
Définit l' **ID application**.

MQLONG applicationType() const ;
Renvoie le **type d'application**.

void setApplicationType(const MQLONG *type*) ;
Définit le **type d'application**.

ImqBoolean copyOut(MQTMC2 * *ptmc2*);
Encapsule la structure de données MQTM, qui est celle reçue dans les files d'attente d'initialisation. Remplit une structure de données MQTMC2 équivalente fournie par l'appelant et définit la zone QMgrName (qui n'est pas présente dans la structure de données MQTM) sur tous les blancs. La structure de données MQTMC2 est traditionnellement utilisée comme paramètre pour les applications démarrées par un moniteur de déclenchement. Cette méthode renvoie TRUE en cas de succès.

ImqString environmentData() const ;
Renvoie une copie des **données d'environnement**.

void setEnvironmentData(const char * *data*) ;
Définit les **données d'environnement**.

ImqString processName() const ;
Renvoie une copie du **nom de processus**.

void setProcessName(const char * *name*) ;
Définit le **nom de processus**, rempli avec des blancs à 48 caractères.

ImqString queueName() const ;
Renvoie une copie du **nom de file d'attente**.

void setQueueName(const char * *name*) ;
Définit le **nom de file d'attente**, en ajoutant des blancs à 48 caractères.

ImqString triggerData() const ;
Renvoie une copie des **données de déclencheur**.

void setTriggerData(const char * *data*) ;
Définit les **données de déclenchement**.

ImqString userData() const ;
Renvoie une copie des **données utilisateur**.

void setUserData(const char * *data*) ;
Définit les **données utilisateur**.

Données d'objet (protégées)

MQTM *omqtm*
Structure de données MQTM.

Codes raison

- POINT_NULL_MQRC_NULL
- MQRC_INCONSISTENT_FORMAT
- ERREUR_CODAGE_MQRC_ERREUR
- MQRC_STRUC_ID_ERROR

Classe C++ d'en-tête ImqWork

Cette classe encapsule des fonctions spécifiques de la structure de données MQWIH.

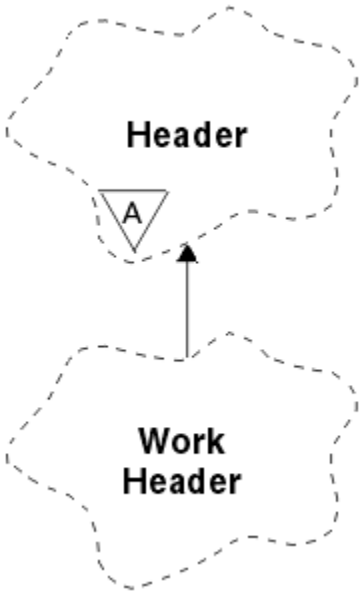


Figure 69. Classe d'en-tête ImqWork

Les objets de cette classe sont utilisés par les applications qui placent des messages dans la file d'attente gérée par z/OS Workload Manager.

- «Attributs d'objet», à la page 1437
- «Constructeurs», à la page 1437
- «Méthodes ImqItem surchargées», à la page 1437
- «Méthodes d'objet (public)», à la page 1438
- «Données d'objet (protégées)», à la page 1438
- «Codes raison», à la page 1438

Attributs d'objet

jeton de message
Jeton de message pour le gestionnaire de charge de travail z/OS , d'une longueur de MQ_MSG_TOKEN_LENGTH. La valeur initiale est MQMTOK_NONE.

nom de service
Nom de 32 caractères d'un processus. Le nom est initialement à blanc.

étape de service
Nom de 8 caractères d'une étape dans le processus. Le nom est initialement à blanc.

Constructeurs

ImqWorkHeader () ;
Constructeur par défaut.

ImqWorkHeader (const ImqWorkHeader & ***header*** **) ;**
Constructeur de copie.

Méthodes ImqItem surchargées

virtual ImqBoolean copyOut(ImqMessage & ***msg*** **) ;**
Insère une structure de données MQWIH au début de la mémoire tampon du message, en déplaçant les données de message existantes plus loin et en définissant le format *msg* sur MQFMT_WORK_INFO_HEADER.

Pour plus de détails, voir la description de la méthode de classe parent.

virtual ImqBoolean pasteIn(ImqMessage & *msg*) ;
Lit une structure de données MQWIH à partir de la mémoire tampon de messages.

Pour que l'opération aboutisse, le codage de l'objet *msg* doit être MQENC_NATIVE. Extrayez les messages avec MQGMO_CONVERT en MQENC_NATIVE.

Le format ImqMessage doit être MQFMT_WORK_INFO_HEADER.

Pour plus de détails, voir la description de la méthode de classe parent.

Méthodes d'objet (public)

void operator = (const ImqWorkHeader & *header*) ;
Copie les données d'instance à partir de l'en-tête , en remplaçant les données d'instance existantes.

ImqBinary messageToken () const ;
Renvoie le **jeton de message**.

ImqBoolean setMessageToken (const ImqBinary & *jeton*) ;
Définit le **jeton de message**. La longueur des données du *jeton* doit être égale à zéro ou à MQ_MSG_TOKEN_LENGTH. Elle renvoie TRUE en cas de succès.

void setMessageToken (const MQBYTE16 *jeton* = 0) ;
Définit le **jeton de message**. *token* peut être égal à zéro, ce qui revient à spécifier MQMTOK_NONE. Si *token* est différent de zéro, il doit traiter les octets MQ_MSG_TOKEN_LENGTH des données binaires.

Lorsque vous utilisez des valeurs prédéfinies telles que MQMTOK_NONE, vous pouvez être amené à effectuer un transtypage pour garantir une correspondance de signature ; par exemple, (MQBYTE *) MQMTOK_NONE.

ImqString serviceName () const ;
Renvoie le **nom de service**, y compris les blancs de fin.

void setServiceName (const char * *name*) ;
Définit le **nom de service**.

ImqString serviceStep () const ;
Renvoie l' **étape de service**, y compris les blancs de fin.

void setServiceStep (const char * *étape*) ;
Définit l' **étape de service**.

Données d'objet (protégées)

Omqwih MQWIH
Structure de données MQWIH.

Codes raison

- MQRC_BINARY_DATA_LENGTH_ERROR

Les classes IBM WebSphere MQ pour les bibliothèques Java

L'emplacement des bibliothèques IBM WebSphere MQ classes for Java varie en fonction de la plateforme. Indiquez cet emplacement lorsque vous démarrez une application.

Pour spécifier l'emplacement des bibliothèques JNI (Java Native Interface), démarrez votre application à l'aide d'une commande **java** au format suivant:

```
java -Djava.library.path=library_path application_name
```

où *chemin_bibliothèque* est le chemin d'accès aux classes WebSphere MQ pour les bibliothèques Java, qui incluent les bibliothèques JNI. Tableau 615, à la page 1439 affiche l'emplacement des classes WebSphere MQ pour les bibliothèques Java pour chaque plateforme.

Tableau 615. Emplacement des bibliothèques WebSphere MQ pour chaque plateforme	
Plateforme	**Répertoire contenant les bibliothèques WebSphere MQ pour Java**
AIX	*MQ_INSTALLATION_PATH*/java/lib (bibliothèques 32 bits) *MQ_INSTALLATION_PATH*/java/lib64 (bibliothèques 64 bits)
HP-UX Linux (POWER, x86-64 et zSeries s390x (plateformes) Solaris (plateformesx86-64 et SPARC)	*MQ_INSTALLATION_PATH*/java/lib (bibliothèques 32 bits) *MQ_INSTALLATION_PATH*/java/lib64 (bibliothèques 64 bits)
Linux (plateformex86)	*MQ_INSTALLATION_PATH*/java/lib
Windows	*MQ_INSTALLATION_PATH*\Java\lib (bibliothèques 32 bits) *MQ_INSTALLATION_PATH*\Java\lib64 (bibliothèques 64 bits)
MQ_INSTALLATION_PATH représente le répertoire de haut niveau dans lequel WebSphere MQ est installé.	

Remarque :

1. Sous AIX, HP-UX, Linux (Power platform) ou Solaris, utilisez les bibliothèques 32 bits ou 64 bits. Utilisez les bibliothèques 64 bits uniquement si vous exécutez votre application dans une machine virtuelle Java (JVM) 64 bits sur une plateforme 64 bits. Sinon, utilisez les bibliothèques 32 bits.
2. Sous Windows, vous pouvez utiliser la variable d'environnement PATH pour spécifier l'emplacement des classes WebSphere MQ pour les bibliothèques Java au lieu de spécifier leur emplacement dans la commande **java** .
3. Pour utiliser WebSphere MQ classes for Java en mode liaisons sous IBM i, vérifiez que la bibliothèque QMQMJAVA figure dans votre liste de bibliothèques.

Tâches associées

Utilisation des classes WebSphere MQ pour Java

Propriétés des objets IBM WebSphere MQ classes for JMS

Tous les objets dans IBM WebSphere MQ classes for JMS ont des propriétés. Différentes propriétés s'appliquent à différents types d'objet. Les différentes propriétés ont des valeurs admises différentes et les valeurs des propriétés symboliques diffèrent entre l'outil d'administration et le code de programme.

IBM WebSphere MQ classes for JMS fournit des fonctions permettant de définir et d'interroger les propriétés des objets à l'aide de l'outil d'administration JMS WebSphere MQ , WebSphere MQ Explorer ou dans une application. De nombreuses propriétés ne sont pertinentes que pour un sous-ensemble spécifique des types d'objet.

Pour plus d'informations sur l'utilisation de l'outil d'administration JMS WebSphere MQ , voir Utilisation de l'outil d'administration JMS WebSphere MQ .

Tableau 616, à la page 1440 fournit une brève description de chaque propriété et indique, pour chaque propriété, les types d'objet auxquels elle s'applique. Les types d'objet sont identifiés à l'aide de mots clés. Pour plus d'informations, voir Types d'objet JMS .

Les numéros se réfèrent aux notes à la fin du tableau. Voir aussi «Dépendances entre les propriétés des objets WebSphere MQ classes for JMS», à la page 1491.

Une propriété se compose d'une paire nom-valeur au format suivant:

```
PROPERTY_NAME(property_value)
```

Les rubriques de cette section répertorient, pour chaque propriété, le nom de la propriété et une brève description, et affichent les valeurs de propriété valides utilisées dans l'outil d'administration. et la méthode set utilisée pour définir la valeur de la propriété dans une application. Les rubriques affichent également les valeurs de propriété valides pour chaque propriété et le mappage entre les valeurs de propriété symboliques utilisées dans l'outil et leurs équivalents programmables.

Les noms de propriété ne sont pas sensibles à la casse et sont limités à l'ensemble des noms reconnus affichés dans ces rubriques.

Tableau 616. Noms de propriété et types d'objet applicables

Propriété	**Forme abrégée**	**Type d'objet**							
		CF	**QCF**	**TCF**	**Q**	**T**	**XACF**	**XAQCF**	**Fonction XATCF**
«APPLICATIONNAME», à la page 1443	APPNAME	Y	Y	Y			Y	Y	Y
«ASYNCEXCEPTION», à la page 1444	AEX	Y	Y	Y			Y	Y	Y
«BROKERCCDURSUBQ», à la page 1445[1]	CCDSUB					Y			
«BROKERCCSUBQ», à la page 1446[1]	CCSUB	Y		Y			Y		Y
«BROKERCONQ», à la page 1446[1]	BCON	Y		Y			Y		Y
«BROKERDURSUBQ», à la page 1446[1]	BDSUB					Y			
«BROKERPUBQ», à la page 1447[1]	BPUB	Y		Y		Y	Y		Y
«BROKERPUBQMGR», à la page 1447[1]	BPQM					Y			
«BROKERQMGR», à la page 1448[1]	BQM	Y		Y			Y		Y
«BROKERSUBQ», à la page 1448[1]	BSUB	Y		Y			Y		Y
«BROKERVER», à la page 1449[1]	BVER	O[2]		O[2]		Y	Y		Y
«CCDTURL», à la page 1449[3]	Table de définition de canal du client (CCDT)	Y	Y	Y			Y	Y	Y
«CCSID», à la page 1450	CCS	Y	Y	Y	Y	Y	Y	Y	Y
«Canal», à la page 1450[3]	CHAN	Y	Y	Y			Y	Y	Y
«CLEANUP», à la page 1451[1]	CL	Y		Y			Y		Y
«CLEANUPINT», à la page 1451[1]	CLINT	Y		Y			Y		Y
«ConnectionNameList», à la page 1452	LISTE DES CNTS	Y	Y	Y					

Tableau 616. Noms de propriété et types d'objet applicables (suite)

Propriété	Forme abrégée	Type d'objet							
		CF	QCF	TCF	Q	T	XACF	XAQCF	Fonction XATCF
«CLIENTRECONNECTOPTIONS», à la page 1452	CROPT	Y	Y	Y					
«CLIENTRECONNECTTIMEOUT», à la page 1453	CRT	Y	Y	Y					
«IDCLIENT», à la page 1453	CID	O[2]	Y	O[2]			Y	Y	Y
«CLONESUPP», à la page 1454	CLS	Y		Y			Y		Y
«COMPHDR», à la page 1454	pays d"origine	Y		Y			Y		Y
«COMPMSG», à la page 1455	MC	Y	Y	Y			Y	Y	Y
«CONNOPT», à la page 1455	CNOPT	Y	Y	Y			Y	Y	Y
«CONNTAG», à la page 1456	CNTAG	Y	Y	Y			Y	Y	Y
«DESCRIPTION», à la page 1457	DECROISSANT	O[2]	Y	O[2]	Y	Y	Y	Y	Y
«DIRECTAUTH», à la page 1457	DAUTH	O[2]		O[2]					
«ENCODING», à la page 1458	ENC				Y	Y			
«EXPIRY», à la page 1459	EXP				Y	Y			
«FAILIFQUIESCE», à la page 1459	FIQ	Y	Y	Y	Y	Y	Y	Y	Y
«HOSTNAME», à la page 1460	HOST	O[2]	Y	O[2]			Y	Y	Y
«LOCALADDRESS», à la page 1460	LA	O[2]	Y	O[2]			Y	Y	Y
«NOM_MAPPE», à la page 1461	MMNST	Y	Y	Y			Y	Y	Y
«MAXBUFFSIZE», à la page 1462	MBSZ	O[2]		O[2]					
«MDREAD», à la page 1462	MDR				Y	Y			
«MDWRITE», à la page 1463	MDW				Y	Y			
«MDMSGCTX», à la page 1463	MDCTX				Y	Y			
«MSGBATCHSZ», à la page 1464[1]	MBS	Y	Y	Y			Y	Y	Y
«MSGBODY», à la page 1464	MBODY				Y	Y			
«MSGRETENTION», à la page 1465	MRET	Y	Y				Y	Y	
«MSGSELECTION», à la page 1465[1]	MSEL	Y		Y			Y		Y
«MULTICAST», à la page 1466	MCAST	O[2]		O[2]		Y			
«OPTIMISTICPUBLICATION», à la page 1467[1]	OPTPUB	Y		Y					
«OUTCOMENOTIFICATION», à la page 1467[1]	NOTIFY	Y		Y					
«PERSISTENCE», à la page 1468	PER				Y	Y			

Tableau 616. Noms de propriété et types d'objet applicables (suite)

Propriété	Forme abrégée	Type d'objet							
		CF	QCF	TCF	Q	T	XACF	XAQCF	Fonction XATCF
«POLLINGINT», à la page 1468[1]	PINT	Y	Y	Y			Y	Y	Y
«PORT», à la page 1469	PORT	O[2]	Y	O[2]			Y	Y	Y
«PRIORITY», à la page 1469	PRI				Y	Y			
«PROCESSDURATION», à la page 1470[1]	PROCDUR	Y		Y					
«PROVIDERVERSION», à la page 1470	PVER	Y	Y	Y			Y	Y	Y
«PROXYHOSTNAME», à la page 1472	PHOST	O[2]		O[2]					
«PROXYPORT», à la page 1472	PPORT	O[2]		O[2]					
«PUBACKINT», à la page 1472[1]	PAI	Y		Y			Y		Y
«PUTASYNCALLOWED», à la page 1473	PAALD				Y	Y			
«QMANAGER», à la page 1474	QMGR	Y	Y	Y	Y		Y	Y	Y
«QUEUE», à la page 1474	QU				Y				
«READAHEADALLOWED», à la page 1474	RAALD				Y	Y			
«READAHEADCLOSEPOLICY», à la page 1475	RACP				Y	Y			
«RECEIVECCSID», à la page 1476	RCCS				Y	Y			
«RECEIVECONVERSION», à la page 1476	RCNV				Y	Y			
«RECEIVEISOLATION», à la page 1477[1]	RCVISOL	Y		Y					
«RECEXIT», à la page 1477	RCX	Y	Y	Y			Y	Y	Y
«RECEXITINIT», à la page 1478	RCXI	Y	Y	Y			Y	Y	Y
«REPLYTOSTYLE», à la page 1478	RTOST				Y	Y			
«RESCANINT», à la page 1479[1]	RINT	Y	Y				Y	Y	
«SECEXIT», à la page 1479	SCX	Y	Y	Y			Y	Y	Y
«SECEXITINIT», à la page 1480	SCXI	Y	Y	Y			Y	Y	Y
«SENDCHECKCOUNT», à la page 1480	SCC	Y	Y	Y			Y	Y	Y
«SENDEXIT», à la page 1481	SDX	Y	Y	Y			Y	Y	Y
«SENDEXITINIT», à la page 1481	SDXI	Y	Y	Y			Y	Y	Y
«SHARECONVALLOWED», à la page 1482	SCALD	Y	Y	Y			Y	Y	Y
«SPARSESUBS», à la page 1482[1]	SSUBS	Y		Y					

Tableau 616. Noms de propriété et types d'objet applicables (suite)

Propriété	Forme abrégée	Type d'objet							
		CF	QCF	TCF	Q	T	XACF	XAQCF	Fonction XATCF
«SSLCIPHERSUITE», à la page 1483	SCPHS	Y	Y	Y			Y	Y	Y
«SSLCRL», à la page 1483	SCRL	Y	Y	Y			Y	Y	Y
«SSLFIPSREQUIRED», à la page 1484	SFIPS	Y	Y	Y			Y	Y	Y
«SSLPEERNAME», à la page 1484	SPEER	Y	Y	Y			Y	Y	Y
«SSLRESETCOUNT», à la page 1485	SRC	Y	Y	Y			Y	Y	Y
«STATREFRESHINT», à la page 1485[1]	SRI	Y		Y			Y		Y
«SUBSTORE», à la page 1486[1]	SS	Y		Y			Y		Y
«SYNCPOINTALLGETS», à la page 1486	SPAG	Y	Y	Y			Y	Y	Y
«TARGCLIENT», à la page 1487	TC				Y	Y			
«TARGCLIENTMATCHING», à la page 1487	TCM	Y	Y				Y	Y	
«TEMPMODEL», à la page 1488	TM	Y	Y				Y	Y	
«TEMPQPREFIX», à la page 1488	TQP	Y	Y				Y	Y	
«TEMPTOPICPREFIX», à la page 1489	TTP	Y		Y			Y		Y
«TOPIC», à la page 1489	TOP					Y			
«TRANSPORT», à la page 1489	TRAN	O[2]	Y	O[2]			Y	Y	Y
«WILDCARDFORMAT», à la page 1490	WCFMT	Y		Y			Y		Y

Remarque :

1. Cette propriété peut être utilisée avec la version 7.0 de WebSphere MQ classes for JMS mais n'a aucun effet pour une application connectée à un gestionnaire de files d'attente version 7.0 sauf si la propriété PROVIDERVERSION de la fabrique de connexions est définie sur un numéro de version inférieur à 7.
2. Seules les propriétés BROKERVER, CLIENTID, DESCRIPTION, DIRECTAUTH, HOSTNAME, LOCALADDRESS, MAXBUFFSIZE, MULTICAST, PORT, PROXYHOSTNAME, PROXYPORT et TRANSPORT sont prises en charge pour un objet ConnectionFactory ou TopicConnectionFactory lors de l'utilisation d'une connexion en temps réel à un courtier.
3. Les propriétés CCDTURL et CHANNEL d'un objet ne doivent pas être définies simultanément.

APPLICATIONNAME

Une application peut définir un nom identifiant sa connexion au gestionnaire de files d'attente. Ce nom d'application est affiché par la commande **DISPLAY CONN MQSC/PCF** (où la zone est appelée **APPLTAG**) ou dans l'affichage **Connexions d'application** de l'explorateur IBM WebSphere MQ (où la zone est appelée **App name**).

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: APPLICATIONNAME

Nom abrégé de l'outil d'administration JMS: APPNAME

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setAppNom ()
- MQConnectionFactory.getAppNom ()

Valeurs

Toute chaîne valide ne dépassant pas 28 caractères. Les noms plus longs sont ajustés pour tenir en supprimant les noms de package de début, si nécessaire. Par exemple, si la classe appelante est `com.example.MainApp`, le nom complet est utilisé, mais si la classe appelante est `com.example.dictionaryAndThesaurus.multilingual.mainApp`, le nom `multilingual.mainApp` est utilisé, car il s'agit de la combinaison la plus longue du nom de classe et du nom de package le plus à droite qui correspond à la longueur disponible.

Si le nom de classe lui-même comporte plus de 28 caractères, il est tronqué pour tenir. Par exemple, `com.example.mainApplicationForSecondTestCase` devient `mainApplicationForSecondTest`.

ASYNCEXCEPTION

Cette propriété détermine si WebSphere MQ classes for JMS informe un programme d'écoute des exceptions (ExceptionListener) uniquement lorsqu'une connexion est interrompue ou lorsqu'une exception se produit de manière asynchrone sur un appel d'API JMS. Cela s'applique à toutes les connexions créées à partir de cette ConnectionFactory pour lesquelles un ExceptionListener est enregistré.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: ASYNCEXCEPTION

Nom abrégé de l'outil d'administration JMS: AEX

Accès par programme

méthodes d'accès set / getters

- Exceptions MQConnectionFactory.setAsync()
- Exceptions MQConnectionFactory.getAsync()

Valeurs

ASYNC_EXCEPTIONS_ALL

Toutes les exceptions détectées de façon asynchrone, en dehors de la portée d'un appel d'API synchrone, et toutes les exceptions de connexion interrompue sont envoyées au programme d'écoute des exceptions.

Environnement	Valeur
Outil d'administration JMS	TOUT
Programmé	WMQCONSTANTS.ASYNC_EXCEPTIONS_ALL = -1

Environnement	Valeur
WebSphere MQ Explorer	Tous

ASYNC_EXCEPTIONS_CONNECTIONBROKEN

Seules les exceptions signalant une connexion interrompue sont envoyées au programme d'écoute des exceptions. Toutes les autres exceptions se produisant au cours d'un traitement asynchrone ne sont pas rapportées au programme d'écoute des exceptions ; l'application n'est donc pas informée de ces exceptions. V 7.5.0.8 Il s'agit de la valeur par défaut de IBM WebSphere MQ Version 7.5.0, groupe de correctifs 8 (voir JMS: Modifications du programme d'écoute des exceptions dans la version 7.5).

Environnement	Valeur
Outil d'administration JMS	CONNEXION interrompue
Programmé	WMQCONSTANTS.ASYNC_EXCEPTIONS_CONNECTIONBROKEN = 1
WebSphere MQ Explorer	Connexion interrompue

La constante supplémentaire suivante est définie: V 7.5.0.8

- Depuis Version 7.5.0, Fix Pack 8: WMQCONSTANTS.ASYNC_EXCEPTIONS_DEFAULT = ASYNC_EXCEPTIONS_CONNECTIONBROKEN
- Avant Version 7.5.0, Fix Pack 8: WMQCONSTANTS.ASYNC_EXCEPTIONS_DEFAULT = ASYNC_EXCEPTIONS_ALL

Concepts associés

Exceptions dans WebSphere MQ classes for JMS

BROKERCCDURSUBQ

Nom de la file d'attente à partir de laquelle les messages d'abonnement durable sont extraits pour un ConnectionConsumer.

Objets applicables

Topic

Nom long de l'outil d'administration JMS: BROKERCCDURSUBQ

Nom abrégé de l'outil d'administration JMS: CCDSUB

Accès par programme

Configurateurs / méthodes d'accès get

- MQTopic.setBrokerCCDurSubQueue()
- MQTopic.getBrokerCCDurSubQueue()

Valeurs

SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE
Il s'agit de la valeur par défaut.

Toute chaîne valide

BROKERCCSUBQ

Nom de la file d'attente à partir de laquelle les messages d'abonnement non durable sont extraits pour un ConnectionConsumer.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: BROKERCCSUBQ

Nom abrégé de l'outil d'administration JMS: CCSUB

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setBrokerCCSubQueue()
- MQConnectionFactory.getBrokerCCSubQueue()

Valeurs

SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE
Il s'agit de la valeur par défaut.

Toute chaîne valide

BROKERCONQ

Nom de la file d'attente de contrôle du courtier.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: BROKERCONQ

Nom abrégé de l'outil d'administration JMS: BCON

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setBrokerControlQueue()
- MQConnectionFactory.getBrokerControlQueue()

Valeurs

SYSTEM.BROKER.CONTROL.QUEUE
Il s'agit de la valeur par défaut.

Toute chaîne valide

BROKERDURSUBQ

Lorsque les classes WebSphere MQ pour JMS sont utilisées en mode de migration du fournisseur de messagerie WebSphere MQ , cette propriété indique le nom de la file d'attente à partir de laquelle les messages d'abonnement durable sont extraits.

Objets applicables

Topic

Nom long de l'outil d'administration JMS: BROKERDURSUBQ

Nom abrégé de l'outil d'administration JMS: BDSUB

Accès par programme

Configurateurs / méthodes d'accès get

- MQTopic.setBrokerDurSubQueue()
- MQTopic.getBrokerDurSubQueue()

Valeurs

SYSTEM.JMS.D.SUBSCRIBER.QUEUE
Il s'agit de la valeur par défaut.

Toute chaîne valide
Démarrage avec SYSTEM.JMS.D

Concepts associés
Règles de sélection du mode de fournisseur de messagerie de WebSphere MQ

BROKERPUBQ

Nom de la file d'attente dans laquelle les messages publiés sont envoyés (file d'attente de flux).

Objets applicables

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: BROKERPUBQ

Nom abrégé de l'outil d'administration JMS: BPUB

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setBrokerPubQueue
- MQConnectionFactory.getBrokerPubQueue

Valeurs

SYSTEM.BROKER.DEFAULT.STREAM
Il s'agit de la valeur par défaut.

Toute chaîne valide

BROKERPUBQMGR

Nom du gestionnaire de files d'attente propriétaire de la file d'attente dans laquelle les messages publiés sur la rubrique sont envoyés.

Objets applicables

Topic

Nom long de l'outil d'administration JMS: BROKERPUBQMGR

Nom abrégé de l'outil d'administration JMS: BPQM

Accès par programme

Configurateurs / méthodes d'accès get

- MQTopic.setBrokerPubQueueManager()
- MQTopic.getBrokerPubQueueManager()

Valeurs

null
Il s'agit de la valeur par défaut.

Toute chaîne valide

BROKERQMGR

Nom du gestionnaire de files d'attente sur lequel le courtier s'exécute.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: BROKERQMGR

Nom abrégé de l'outil d'administration JMS: BQM

Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setBrokerQueueManager()
- MQConnectionFactory.getBrokerQueueManager()

Valeurs

null
Il s'agit de la valeur par défaut.

Toute chaîne valide

BROKERSUBQ

Lorsque les classes WebSphere MQ pour JMS sont utilisées en mode de migration du fournisseur de messagerie WebSphere MQ , cette propriété indique le nom de la file d'attente à partir de laquelle les messages d'abonnement non durable sont extraits.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: BROKERSUBQ

Nom abrégé de l'outil d'administration JMS: BSUB

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setBrokerSubQueue()
- MQConnectionFactory.getBrokerSubQueue()

Valeurs

SYSTEM.JMS.ND.SUBSCRIBER.QUEUE
Il s'agit de la valeur par défaut.

Toute chaîne valide
Démarrage avec SYSTEM.JMS.ND

Concepts associés
Règles de sélection du mode de fournisseur de messagerie de WebSphere MQ

BROKERVER

Version du courtier utilisé.

Objets applicables

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: BROKERVER

Nom abrégé de l'outil d'administration JMS: BVER

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setBrokerVersion ()
- MQConnectionFactory.getBrokerVersion ()

Valeurs

V1
Pour utiliser un courtier WebSphere MQ Publish / Subscribe ou un courtier WebSphere MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker ou WebSphere Business Integration Message Broker en mode compatibilité. Il s'agit de la valeur par défaut si TRANSPORT est défini sur BIND ou CLIENT.

V2
Pour utiliser un courtier WebSphere MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker ou WebSphere Business Integration Message Broker en mode natif. Il s'agit de la valeur par défaut si TRANSPORT est défini sur DIRECT ou DIRECTIVE THTTP.

non spécifié
Une fois que le courtier a migré de V6 vers V7, définissez cette propriété de sorte que les en-têtes RFH2 ne soient plus utilisés. Après la migration, cette propriété n'est plus pertinente.

CCDTURL

Adresse URL (Uniform Resource Locator) identifiant le nom et l'emplacement du fichier contenant la table de définition de canal du client et indiquant comment le fichier est accessible.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CCDTURL

Nom abrégé de l'outil d'administration JMS: CCDT

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setCCDTURL()
- MQConnectionFactory.getCCDTURL()

Valeurs

null
Il s'agit de la valeur par défaut.

Adresse URL (Uniform Resource Locator)

CCSID

ID de jeu de caractères codés à utiliser pour une connexion ou une destination.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CCSID

Nom abrégé de l'outil d'administration JMS: CCS

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setCCSID()
- MQConnectionFactory.getCCSID()

Valeurs

819
Il s'agit de la valeur par défaut pour une fabrique de connexions.

1208
Il s'agit de la valeur par défaut pour une destination.

Tout entier positif

Canal

Nom du canal de connexion client utilisé.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CHANNEL

Nom abrégé de l'outil d'administration JMS: CHAN

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setChannel()
- MQConnectionFactory.getChannel()

Valeurs

SYSTEM.DEF.SVRCONN
Il s'agit de la valeur par défaut.

Toute chaîne valide

CLEANUP

Niveau de nettoyage pour les magasins d'abonnement BROKER ou MIGRATE.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CLEANUP

Nom abrégé de l'outil d'administration JMS: CL

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setCleanupNiveau ()
- MQConnectionFactory.getCleanupNiveau ()

Valeurs

SECURISEE
Utilisez un nettoyage sécurisé. Il s'agit de la valeur par défaut.

PROP
Utilisez un nettoyage sûr, fort ou nul en fonction d'une propriété définie sur la ligne de commande Java.

AUCUN
Ne pas utiliser de nettoyage.

Elevé
Utilisez un nettoyage puissant.

CLEANUPINT

Intervalle, en millisecondes, entre les exécutions en arrière-plan de l'utilitaire de nettoyage de publication / abonnement.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CLEANUPINT

Nom abrégé de l'outil d'administration JMS: CLINT

Accès par programme

Configurateurs / méthodes d'accès get

- Intervalle MQConnectionFactory.setCleanup()
- MQConnectionFactory.getCleanupIntervalle ()

Valeurs

3600000
Il s'agit de la valeur par défaut.

Tout entier positif

ConnectionNameList

Liste des noms de connexion TCP/IP. La liste est tentée dans l'ordre, une fois par tentative de reconnexion.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: CONNECTIONNAMELIST

Nom abrégé de l'outil d'administration JMS: CNLIST

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setconnectionNameListe ()
- MQConnectionFactory.getconnectionNameListe ()

Valeurs

Liste séparée par des virgules de HOSTNAME (PORT). HOSTNAME peut être un nom DNS ou une adresse IP.

La valeur par défaut de PORT est 1414.

CLIENTRECONNECTOPTIONS

Options régissant la reconnexion.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: CLIENTRECONNECTOPTIONS

Nom abrégé de l'outil d'administration JMS: CROPT

Accès par programme

Méthodes d'accès set / getters

- MQConnectionFactory.setClientReconnectOptions()
- MQConnectionFactory.getClientReconnectOptions()

Valeurs

QMGR

L'application peut se reconnecter, mais uniquement au gestionnaire de files d'attente auquel elle était connectée.

Utilisez cette valeur si une application peut être reconnectée, mais qu'il existe une affinité entre les classes WebSphere MQ pour l'application JMS et le gestionnaire de files d'attente auquel elle a établi une connexion pour la première fois.

Choisissez cette valeur si vous souhaitez qu'une application se reconnecte automatiquement à l'instance de secours d'un gestionnaire de files d'attente à haute disponibilité.

Pour utiliser cette valeur à l'aide d'un programme, utilisez la constante WMQConstants.WMQ_CLIENT_RECONNECT_Q_MGR.

ANY

L'application peut se reconnecter à tout gestionnaire de files d'attente.

Utilisez l'option de reconnexion uniquement s'il n'y a pas d'affinité entre les classes WebSphere MQ pour l'application JMS et le gestionnaire de files d'attente avec lequel il a établi une connexion.

Pour utiliser cette valeur à partir d'un programme, utilisez la constante WMQConstants.WMQ_CLIENT_RECONNECT.

DESACTIVE

L'application ne sera pas reconnectée.

Pour utiliser cette valeur à l'aide d'un programme, utilisez la constante WMQConstants.WMQ_CLIENT_RECONNECT_DISABLED.

ADEF

La reconnexion automatique de l'application dépend de la valeur de l'attribut de canal WebSphere MQ DefReconnect.

Pour utiliser cette valeur à partir d'un programme, utilisez la constante WMQConstants.WMQ_CLIENT_RECONNECT_AS_DEF.

CLIENTRECONNECTTIMEOUT

Délai avant l'arrêt des tentatives de reconnexion.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: CLIENTRECONNECTTIMEOUT

Nom abrégé de l'outil d'administration JMS: CRT

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setClientReconnectTimeout()
- MQConnectionFactory.setClientReconnectTimeout()

Valeurs

Intervalle en secondes. Valeur par défaut: 1800 (30 minutes).

IDCLIENT

L'identificateur de client permet d'identifier de manière unique la connexion de l'application pour les abonnements durables.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CLIENTID

Nom abrégé de l'outil d'administration JMS: CID

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory. IDsetClient()
- MQConnectionFactory. IDgetClient()

Valeurs

null
Il s'agit de la valeur par défaut.

Toute chaîne valide

CLONESUPP

Indique si deux instances ou plus du même abonné durable à la rubrique peuvent s'exécuter simultanément.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CLONESUPP

Nom abrégé de l'outil d'administration JMS: CLS

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory. Prise en charge desetClone()
- MQConnectionFactory. Prise en charge degetClone()

Valeurs

DESACTIVE
Une seule instance d'un abonné à une rubrique durable peut être exécutée à la fois. Il s'agit de la valeur par défaut.

Activée
Deux ou plusieurs instances du même abonné durable peuvent s'exécuter simultanément, mais chaque instance doit s'exécuter dans une machine virtuelle Java (JVM) distincte.

COMPHDR

Liste des techniques pouvant être utilisées pour compresser les données d'en-tête sur une connexion.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: COMPHDR

Nom abrégé de l'outil d'administration JMS: HC

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setHdrCompList()

- MQConnectionFactory.getHdrCompList()

Valeurs

AUCUN
Il s'agit de la valeur par défaut.

SYSTEME
La compression de l'en-tête de message RLE est effectuée.

COMPMSG

Liste des techniques pouvant être utilisées pour compresser les données de message sur une connexion.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: COMPMSG

Nom abrégé de l'outil d'administration JMS: MC

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMsgCompList()
- MQConnectionFactory.getMsgCompList()

Valeurs

AUCUN
Il s'agit de la valeur par défaut.

Liste d'une ou de plusieurs des valeurs suivantes, séparées par des caractères blancs:
RLE ZLIBFAST ZLIBHIGH

CONNOPT

Contrôle la façon dont les applications WebSphere MQ classes for JMS qui utilisent le transport de liaisons se connectent au gestionnaire de files d'attente.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CONNOPT

Nom abrégé de l'outil d'administration JMS: CNOPT

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMQConnectionOptions ()
- MQConnectionFactory.getMQConnectionOptions ()

Valeurs

Standard
La nature de la liaison entre l'application et le gestionnaire de files d'attente dépend de la valeur de l'attribut *DefaultBindType* du gestionnaire de files d'attente. La valeur STANDARD est mappée à WebSphere MQ *ConnectOption* MQCNO_STANDARD_BINDING

PARTAGE
L'application et l'agent du gestionnaire de files d'attente local s'exécutent dans des unités d'exécution distinctes mais partagent certaines ressources. Cette valeur est mappée à WebSphere MQ *ConnectOption* MQCNO_SHARED_BINDING.

Isolated
L'application et l'agent du gestionnaire de files d'attente local s'exécutent dans des unités d'exécution distinctes et ne partagent aucune ressource. La valeur ISOLÉ est mappée à WebSphere MQ *ConnectOption* MQCNO_ISOLATED_BINDING.

Fastpath
L'application et l'agent du gestionnaire de files d'attente local s'exécutent dans la même unité d'exécution. Cette valeur est mappée à WebSphere MQ *ConnectOption* MQCNO_FASTPATH_BINDING.

SERIALQM
L'application demande l'utilisation exclusive de la balise de connexion dans la portée du gestionnaire de files d'attente. Cette valeur est mappée à WebSphere MQ *ConnectOption* MQCNO_SERIALIZE_CONN_TAG_Q_MGR.

SERIALQSG
L'application demande l'utilisation exclusive de la balise de connexion dans la portée du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente. La valeur SERIALQSG est mappée à WebSphere MQ *ConnectOption* MQCNO_SERIALIZE_CONN_TAG_QSG.

RESTRICTQM
L'application demande l'utilisation partagée de la balise de connexion, mais il existe des restrictions sur l'utilisation partagée de la balise de connexion dans la portée du gestionnaire de files d'attente. Cette valeur est mappée à WebSphere MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_Q_MGR.

RESTRICTQSG
L'application demande l'utilisation partagée de la balise de connexion, mais il existe des restrictions sur l'utilisation partagée de la balise de connexion dans la portée du groupe de partage de files d'attente auquel appartient le gestionnaire de files d'attente. Cette valeur est mappée à WebSphere MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_QSG.

Pour plus d'informations sur les options de connexion à WebSphere MQ , voir Connexion à un gestionnaire de files d'attente à l'aide de l'appel MQCONNX .

CONNTAG

Balise que le gestionnaire de files d'attente associe aux ressources mises à jour par l'application dans une unité de travail alors que l'application est connectée au gestionnaire de files d'attente.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CONNTAG

Nom abrégé de l'outil d'administration JMS: CNTAG

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setConnBalise ()
- MQConnectionFactory.getConnBalise ()

Valeurs

Un tableau d'octets de 128 éléments, où chaque élément est 0
Il s'agit de la valeur par défaut.

N'importe quelle chaîne
La valeur est tronquée si elle est supérieure à 128 octets.

DESCRIPTION

Description de l'objet stocké.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: DESCRIPTION

Nom abrégé de l'outil d'administration JMS: DESC

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setDescription()
- MQConnectionFactory.getDescription()

Valeurs

null
Il s'agit de la valeur par défaut.

Toute chaîne valide
`

DIRECTAUTH

Indique si l'authentification SSL est utilisée sur une connexion en temps réel à un courtier.

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: DIRECTAUTH

Nom abrégé de l'outil d'administration JMS: DAUTH

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setDirectAuth ()
- MQConnectionFactory.getDirectAuth ()

Valeurs

BASIC
Aucune authentification, authentification par nom d'utilisateur ou authentification par mot de passe. Il s'agit de la valeur par défaut.

CERTIFICATE
Authentification par certificat de clé publique.

ENCODING

Mode de représentation des données numériques dans le corps d'un message lorsque le message est envoyé à cette destination. La propriété spécifie la représentation d'entiers binaires, d'entiers décimaux condensés et de nombres à virgule flottante.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: ENCODING

Nom abrégé de l'outil d'administration JMS: ENC

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setEncoding()
- MQDestination.getEncoding()

Valeurs

propriété ENCODING

Les valeurs valides que la propriété ENCODING peut prendre sont construites à partir des trois sous-propriétés:

Codage d'entier
Normale ou inversée

Codage décimal
Normale ou inversée

codage en virgule flottante
IEEE normal, IEEE inversé ou z/OS

La propriété ENCODING est exprimée sous la forme d'une chaîne de trois caractères avec la syntaxe suivante:

```
{N|R}{N|R}{N|R|3}
```

Dans cette chaîne:

- N indique normal
- R indique que la valeur est inversée
- 3 indique z/OS
- Le premier caractère représente le *codage d'entier*
- Le second caractère représente le *codage décimal*
- Le troisième caractère représente le *codage en virgule flottante*

Fournit un ensemble de douze valeurs possibles pour la propriété ENCODING .

Il existe une valeur supplémentaire, la chaîne NATIVE, qui définit les valeurs de codage appropriées pour la plateforme Java.

Les exemples suivants montrent des combinaisons valides pour ENCODING:

```
ENCODING(NNR)
ENCODING(NATIVE)
ENCODING(RR3)
```

EXPIRY

Heure à laquelle les messages d'une destination arrivent à expiration.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: EXPIRATION

Nom abrégé de l'outil d'administration JMS: EXP

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setExpiry()
- MQDestination.getExpiry()

Valeurs

APP
L'expiration peut être définie par l'application JMS. Il s'agit de la valeur par défaut.

UNLIM
Aucune expiration n'a lieu.

0
Aucune expiration n'a lieu.

Tout entier positif représentant l'expiration en millisecondes.

FAILIFQUIESCE

Cette propriété détermine si les appels à certaines méthodes échouent si le gestionnaire de files d'attente est à l'état de mise au repos ou si une application se connecte à un gestionnaire de files d'attente à l'aide du transport CLIENT et que le canal utilisé par l'application a été mis à l'état de mise au repos, par exemple à l'aide de la commande **STOP CHANNEL** ou **STOP CHANNEL MODE(QUIESCE)** MQSC.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: FAILIFQUIESCE

Nom abrégé de l'outil d'administration JMS: FIQ

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setFailIfQuiesce()
- MQConnectionFactory.getFailIfQuiesce()

Valeurs

YES
Les appels à certaines méthodes échouent si le gestionnaire de files d'attente est à l'état de mise au repos ou si le canal utilisé pour se connecter à un gestionnaire de files d'attente est en cours de mise au repos. Si une application détecte l'une de ces conditions, elle peut effectuer sa tâche immédiate et fermer la connexion, ce qui permet au gestionnaire de files d'attente ou à l'instance de canal de s'arrêter. Il s'agit de la valeur par défaut.

NO
Aucun appel de méthode n'a échoué car le gestionnaire de files d'attente ou le canal utilisé pour se connecter à un gestionnaire de files d'attente est à l'état de mise au repos. Si vous spécifiez cette valeur, une application ne peut pas détecter que le gestionnaire de files d'attente ou le canal est en cours de mise au repos. L'application peut continuer à effectuer des opérations sur le gestionnaire de files d'attente et, par conséquent, empêcher l'arrêt du gestionnaire de files d'attente.

HOSTNAME

Pour une connexion à un gestionnaire de files d'attente, nom d'hôte ou adresse IP du système sur lequel le gestionnaire de files d'attente s'exécute ou, pour une connexion en temps réel à un courtier, nom d'hôte ou adresse IP du système sur lequel le courtier s'exécute.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: HOSTNAME

Nom abrégé de l'outil d'administration JMS: HOST

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setHostNom ()
- MQConnectionFactory.getHostNom ()

Valeurs

système hôte local
Il s'agit de la valeur par défaut.

Toute chaîne valide

LOCALADDRESS

Pour une connexion à un gestionnaire de files d'attente, cette propriété indique soit l'interface réseau locale à utiliser, soit le port local ou la plage de ports locaux à utiliser. Pour une connexion en temps réel à un courtier, cette propriété est pertinente uniquement lorsque la multidiffusion est utilisée et indique l'interface réseau locale à utiliser.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: LOCALADDRESS

Nom abrégé de l'outil d'administration JMS: LA

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setLocalAdresse ()
- MQConnectionFactory.getLocalAdresse ()

Valeurs

"" (chaîne vide)
Il s'agit de la valeur par défaut.

Chaîne au format [ip-addr] [(low-port [, high port])]
Voici quelques exemples :

192.0.2.0

Le canal se lie à l'adresse 192.0.2.0 en local.

192.0.2.0(1000)

Le canal se lie à l'adresse 192.0.2.0 en local et utilise le port 1000.

192.0.2.0(1000,2000)

Le canal se lie à l'adresse 192.0.2.0 en local et utilise un port compris entre 1000 et 2000.

(1000)

Le canal est lié au port 1000 en local.

(1000,2000)

Le canal se lie à un port compris entre 1000 et 2000 localement.

Vous pouvez spécifier un nom d'hôte à la place d'une adresse IP. Pour une connexion en temps réel à un courtier, cette propriété est pertinente uniquement lorsque la multidiffusion est utilisée et la valeur de la propriété ne doit pas contenir de numéro de port ou de plage de numéros de port. Les seules valeurs valides de la propriété dans ce cas sont null, une adresse IP ou un nom d'hôte.

NOM_MAPPE

Permet d'utiliser le style de compatibilité pour les noms d'élément MapMessage .

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: MAPNAMESTYLE

Nom abrégé de l'outil d'administration JMS: MNST

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMapNameStyle()
- MQConnectionFactory.getMapNameStyle()

Valeurs

Standard
Le format de dénomination standard de l'élément com.ibm.jms.JMSMapMessage doit être utilisé. Il s'agit de la valeur par défaut qui permet d'utiliser des identificateurs Java non autorisés comme nom d'élément.

Compatible
L'ancien format de dénomination de l'élément com.ibm.jms.JMSMapMessage doit être utilisé. Seuls les identificateurs Java autorisés peuvent être utilisés comme nom d'élément. Nécessaire uniquement si des messages de mappe sont envoyés à une application qui utilise une version de IBM WebSphere MQ classes for JMS antérieure à la version 5.3.

MAXBUFFSIZE

Nombre maximal de messages reçus qui peuvent être stockés dans une mémoire tampon de messages internes en attendant d'être traités par l'application. Cette propriété s'applique uniquement lorsque TRANSPORT a la valeur DIRECT ou DIRECTION THTTP.

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: MAXBUFFSIZE

Nom abrégé de l'outil d'administration JMS: MBSZ

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMaxBufferSize()
- MQConnectionFactory.getMaxBufferSize()

Valeurs

1 000
Il s'agit de la valeur par défaut.

Tout entier positif

MDREAD

Cette propriété détermine si une application JMS peut extraire les valeurs des zones MQMD.

Objets applicables

Nom long de l'outil d'administration JMS: MDREAD

Nom abrégé de l'outil d'administration JMS: MDR

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setMQMDReadEnabled()
- MQDestination.getMQMDReadEnabled()

Valeurs

NO
Lors de l'envoi de messages, les propriétés JMS_IBM_MQMD* d'un message envoyé ne sont pas mises à jour pour refléter les valeurs de zone mises à jour dans le MQMD. Lors de la réception de messages, aucune des propriétés JMS_IBM_MQMD* n'est disponible sur un message reçu, même si l'émetteur les a définies partiellement ou intégralement. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez False.

Oui
Lors de l'envoi de messages, toutes les propriétés JMS_IBM_MQMD* d'un message envoyé sont mises à jour pour refléter les valeurs de zone mises à jour dans le MQMD, y compris les propriétés que l'expéditeur n'a pas définies explicitement. Lors de la réception de messages, toutes les propriétés JMS_IBM_MQMD* sont disponibles sur un message reçu, y compris les propriétés que l'expéditeur n'a pas définies explicitement.

Pour les programmes, utilisez True.

MDWRITE

Cette propriété détermine si une application JMS peut définir les valeurs des zones MQMD.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: MDWRITE

Nom abrégé de l'outil d'administration JMS: MDR

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setMQMDWriteEnabled()
- MQDestination.getMQMDWriteEnabled()

Valeurs

NO
Toutes les propriétés JMS_IBM_MQMD* sont ignorées et leurs valeurs ne sont pas copiées dans la structure MQMD sous-jacente. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez False.

YES
Les propriétés JMS_IBM_MQMD* sont traitées. Leurs valeurs sont copiées dans la structure sous-jacente.

Pour les programmes, utilisez True.

MDMSGCTX

Niveau de contexte de message à définir par l'application JMS. L'application doit être exécutée avec les droits appropriés pour que cette propriété soit appliquée.

Objets applicables

Nom long de l'outil d'administration JMS: MDMSGCTX

Nom abrégé de l'outil d'administration JMS: MDCTX

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setMQMDMessageContext()
- MQDestination.getMQMDMessageContext()

Valeurs

Valeur par défaut
L'appel d'API MQOPEN et la structure MQPMO ne spécifient aucune option de contexte de message explicite. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQ_MDCTX_DEFAULT.

SET_IDENTITY_CONTEXT
L'appel API MQOPEN API spécifie l'option de contexte de message MQOO_SET_IDENTITY_CONTEXT et la structure MQPMO spécifie MQPMO_SET_IDENTITY_CONTEXT.

Pour les programmes, utilisez WMQ_MDCTX_SET_IDENTITY_CONTEXT.

SET_ALL_CONTEXT
L'appel API MQOPEN API spécifie l'option de contexte de message MQOO_SET_ALL_CONTEXT et la structure MQPMO spécifie MQPMO_SET_ALL_CONTEXT.

Pour les programmes, utilisez WMQ_MDCTX_SET_ALL_CONTEXT.

MSGBATCHSZ

Nombre maximal de messages à prendre dans une file d'attente dans un paquet lors de l'utilisation de la distribution de messages asynchrones.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: MAXBUFFSIZE

Nom abrégé de l'outil d'administration JMS: MBSZ

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMsgBatchSize()
- MQConnectionFactory.getMsgBatchSize()

Valeurs

10
Il s'agit de la valeur par défaut.

Tout entier positif

MSGBODY

Détermine si une application JMS accède à l' MQRFH2 d'un message IBM WebSphere MQ dans le cadre de la charge de message.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: WMQ_MESSAGE_BODY

Nom abrégé de l'outil d'administration JMS: MBODY

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMessageBodyStyle()
- MQConnectionFactory.getMessageBodyStyle()

Valeurs

NON SPECIFIE
Lors de l'envoi, IBM WebSphere MQ classes for JMS génère ou non et inclut un en-tête MQRFH2 , en fonction de la valeur de WMQ_TARGET_CLIENT. Lors de la réception, agit en tant que valeur JMS.

JMS
Lors de l'envoi, IBM WebSphere MQ classes for JMS génère automatiquement un en-tête MQRFH2 et l'inclut dans le message WebSphere MQ .

Lors de la réception, IBM WebSphere MQ classes for JMS définit les propriétés de message JMS en fonction des valeurs dans MQRFH2 (le cas échéant) ; il ne présente pas MQRFH2 comme partie intégrante du corps du message JMS.

MQ
Lors de l'envoi, IBM WebSphere MQ classes for JMS ne génère pas de MQRFH2.

Lors de la réception, IBM WebSphere MQ classes for JMS présente MQRFH2 dans le corps du message JMS.

MSGRETENTION

Indique si le consommateur de connexion conserve les messages non distribués dans la file d'attente d'entrée.

Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory,

Nom long de l'outil d'administration JMS: MSGRETENTION

Nom abrégé de l'outil d'administration JMS: MRET

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMessageRetention ()
- MQConnectionFactory.getMessageConservation ()

Valeurs

Oui
Les messages non distribués restent dans la file d'entrée. Il s'agit de la valeur par défaut.

Non
Les messages non distribués sont traités en fonction de leurs options de disposition.

MSGSELECTION

Détermine si la sélection des messages est effectuée par les classes WebSphere MQ pour JMS ou par le courtier. Si TRANSPORT a la valeur DIRECT, la sélection des messages est toujours effectuée par le courtier et la valeur de MSGSELECTION est ignorée. La sélection des messages par le courtier n'est pas prise en charge lorsque BROKERVER a la valeur V1.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: MSGSELECTION

Nom abrégé de l'outil d'administration JMS: MSEL

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMessageSélection ()
- MQConnectionFactory.getMessageSélection ()

Valeurs

client
La sélection des messages est effectuée par les classes WebSphere MQ pour JMS. Il s'agit de la valeur par défaut.

COURTIER
La sélection de message est faite par le courtier.

MULTICAST

Permet d'activer la multidiffusion sur une connexion en temps réel à un courtier et, si elle est activée, de spécifier la manière précise dont la multidiffusion est utilisée pour distribuer des messages du courtier à un consommateur de message. La propriété n'a aucun effet sur la façon dont un expéditeur de message envoie des messages à un courtier.

Objets applicables

ConnectionFactory, TopicConnectionFactory, Rubrique

Nom long de l'outil d'administration JMS: MULTICAST

Nom abrégé de l'outil d'administration JMS: MCAST

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setMulticast()
- MQConnectionFactory.getMulticast()

Valeurs

DESACTIVE
Les messages ne sont pas distribués à un consommateur de message via le transport multidiffusion. Il s'agit de la valeur par défaut pour les objets ConnectionFactory et TopicConnectionFactory.

ASCF
Les messages sont distribués à un consommateur de message en fonction du paramètre de multidiffusion défini pour la fabrique de connexions associée au consommateur de message. Le paramètre de multidiffusion de la fabrique de connexions est noté lors de la création du consommateur de message. Cette valeur est valide uniquement pour les objets de rubrique. Il s'agit de la valeur par défaut pour les objets de rubrique.

Activée
Si la rubrique est configurée pour la multidiffusion dans le courtier, les messages sont distribués à un consommateur de message à l'aide du transport de multidiffusion. Une qualité de service fiable est utilisée si la rubrique est configurée pour une multidiffusion fiable.

FIABLE.
Si la rubrique est configurée pour une multidiffusion fiable dans le courtier, les messages sont distribués au consommateur de messages à l'aide d'un transport multidiffusion avec une qualité de service fiable. Si la rubrique n'est pas configurée pour la multidiffusion fiable, vous ne pouvez pas créer de consommateur de message pour la rubrique.

NOTR
Si la rubrique est configurée pour la multidiffusion dans le courtier, les messages sont distribués au consommateur de message à l'aide du transport de multidiffusion. Une qualité de service fiable n'est pas utilisée, même si la rubrique est configurée pour une multidiffusion fiable.

OPTIMISTICPUBLICATION

Cette propriété détermine si WebSphere MQ classes for JMS renvoie le contrôle immédiatement à un diffuseur de publications qui a publié un message, ou si elle renvoie le contrôle uniquement après avoir terminé tout le traitement associé à l'appel et peut signaler le résultat au diffuseur de publications.

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: OPTIMISTICPUBLICATION

Nom abrégé de l'outil d'administration JMS: OPTPUB

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setOptimisticPublication ()
- MQConnectionFactory.getOptimisticPublication ()

Valeurs

NO
Lorsqu'un diffuseur publie un message, WebSphere MQ classes for JMS ne renvoie pas le contrôle au diffuseur tant qu'il n'a pas terminé tout le traitement associé à l'appel et peut signaler le résultat au diffuseur. Il s'agit de la valeur par défaut.

YES
Lorsqu'un diffuseur de publications publie un message, WebSphere MQ classes for JMS renvoie le contrôle au diffuseur de publications immédiatement avant qu'il n'ait terminé tout le traitement associé à l'appel et qu'il puisse signaler le résultat au diffuseur de publications. WebSphere MQ classes for JMS signale le résultat uniquement lorsque le diffuseur de publications valide le message.

OUTCOMENOTIFICATION

Cette propriété détermine si WebSphere MQ classes for JMS renvoie immédiatement le contrôle à un abonné qui vient d'accuser réception d'un message ou s'il a validé un message, ou s'il renvoie le contrôle uniquement après avoir terminé tout le traitement associé à l'appel et peut signaler le résultat à l'abonné.

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: OUTCOMENOTIFICATION

Nom abrégé de l'outil d'administration JMS: NOTIFY

Accès par programme

Configurateurs / méthodes d'accès get

- Notification MQConnectionFactory.setOutcome()
- MQConnectionFactory.getOutcomeNotification ()

Valeurs

YES
Lorsqu'un abonné accuse réception ou valide un message, WebSphere MQ classes for JMS ne renvoie pas le contrôle à l'abonné tant qu'il n'a pas terminé tout le traitement associé à l'appel et qu'il peut signaler le résultat à l'abonné. Il s'agit de la valeur par défaut.

NO
Lorsqu'un abonné accuse réception ou valide un message, WebSphere MQ classes for JMS renvoie immédiatement le contrôle à l'abonné, avant qu'il n'ait terminé tout le traitement associé à l'appel et qu'il puisse signaler le résultat à l'abonné.

PERSISTENCE

Persistance des messages envoyés à une destination.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: PERSISTENCE

Nom abrégé de l'outil d'administration JMS: PER

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setPersistence()
- MQDestination.getPersistence()

Valeurs

APP
La persistance est définie par l'application JMS. Il s'agit de la valeur par défaut.

QDEF
La persistance prend la valeur par défaut de la file d'attente.

pers
Les messages sont persistants.

NON
Les messages sont non persistants.

ELEVEE
Pour plus d'informations sur l'utilisation de cette valeur, voir Messages JMS persistants .

POLLINGINT

Si chaque programme d'écoute de messages d'une session n'a pas de message approprié dans sa file d'attente, il s'agit de l'intervalle maximal, en millisecondes, qui s'écoule avant que chaque programme d'écoute de messages tente à nouveau d'extraire un message de sa file d'attente. Si l'absence de message approprié est fréquemment observée pour l'un quelconque des écouteurs de messages au sein d'une session, envisagez d'augmenter la valeur de cette propriété. Cette propriété est pertinente uniquement si TRANSPORT a la valeur BIND ou CLIENT.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: POLLINGINT

Nom abrégé de l'outil d'administration JMS: PINT

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory. IntervallesetPolling()

- MQConnectionFactory.getPollingIntervalle ()

Valeurs

5 000
Il s'agit de la valeur par défaut.

Tout entier positif

PORT

Pour une connexion à un gestionnaire de files d'attente, numéro du port sur lequel le gestionnaire de files d'attente est à l'écoute ou, pour une connexion en temps réel à un courtier, numéro du port sur lequel le courtier est à l'écoute des connexions en temps réel.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: PORT

Nom abrégé de l'outil d'administration JMS: PORT

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setPort()
- MQConnectionFactory.getPort()

Valeurs

1414
Il s'agit de la valeur par défaut si TRANSPORT est défini sur CLIENT.

1506
Il s'agit de la valeur par défaut si TRANSPORT est défini sur DIRECT ou DIRECTIVE THTTP.

Tout entier positif

PRIORITY

Priorité des messages envoyés à une destination.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: PRIORITY

Nom abrégé de l'outil d'administration JMS: PRI

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setPriority()
- MQDestination.getPriority()

Valeurs

APP
La priorité est définie par l'application JMS. Il s'agit de la valeur par défaut.

QDEF
La priorité prend la valeur par défaut de la file d'attente.

Tout entier compris entre 0 et 9
De la plus faible à la plus élevée.

PROCESSDURATION

Cette propriété détermine si un abonné garantit le traitement rapide des messages qu'il reçoit avant de renvoyer le contrôle à WebSphere MQ classes for JMS.

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: PROCESSDURATION

Nom abrégé de l'outil d'administration JMS: PROCDUR

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setProcessDurée ()
- MQConnectionFactory.getProcessDurée ()

Valeurs

INCONNU
Un abonné ne peut donner aucune garantie quant à la rapidité avec laquelle il peut traiter un message qu'il reçoit. Il s'agit de la valeur par défaut.

SHORT
Un abonné garantit le traitement rapide des messages qu'il reçoit avant de renvoyer le contrôle à WebSphere MQ classes for JMS.

PROVIDERVERSION

Cette propriété différencie les deux modes de fonctionnement de la messagerie WebSphere MQ : WebSphere MQ en mode normal du fournisseur de messagerie et WebSphere MQ en mode de migration du fournisseur de messagerie.

Le mode normal du fournisseur de messagerie WebSphere MQ utilise toutes les fonctions des gestionnaires de files d'attente WebSphere MQ Version 7.0 pour implémenter JMS. Ce mode est utilisé uniquement pour la connexion à un gestionnaire de files d'attente WebSphere MQ et peut se connecter à des gestionnaires de files d'attente WebSphere MQ Version 7.0 en mode client ou liaison. Ce mode est optimisé pour utiliser la nouvelle fonction WebSphere MQ Version 7.0 . Si vous n'utilisez pas WebSphere MQ Real-Time Transport, le mode de fonctionnement utilisé est déterminé principalement par la propriété PROVIDERVERSION de la fabrique de connexions.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnection Factory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: PROVIDERVERSION

Nom abrégé de l'outil d'administration JMS: PVER

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setProviderVersion ()
- MQConnectionFactory.getProviderVersion ()

Valeurs

Vous pouvez affecter à **PROVIDERVERSION** la valeur 7, 6 ou *unspecified*. Cependant, **PROVIDERVERSION** peut être une chaîne dans l'un des formats suivants :

- V.R.M.F
- V.R.M
- V.R
- V

où V, R, M et F sont des entiers strictement positifs.

7
Utilise le mode normal du fournisseur de messagerie WebSphere MQ.

Si vous définissez PROVIDERVERSION sur 7, seul le mode de fonctionnement normal du fournisseur de messagerie WebSphere MQ est disponible. Si le gestionnaire de files d'attente connecté en raison des autres paramètres de la fabrique de connexions n'est pas un gestionnaire de files d'attente version 7.0 , la méthode createConnection() échoue avec une exception.

Le mode normal du fournisseur de messagerie WebSphere MQ utilise la fonction de partage des conversations et le nombre de conversations pouvant être partagées est contrôlé par la propriété SHARECNV () sur le canal de connexion serveur. Si cette propriété est définie sur 0, vous ne pouvez pas utiliser le mode normal du fournisseur de messagerie WebSphere MQ et la méthode createConnection() échoue avec une exception.

6
Utilise le mode de migration du fournisseur de messagerie WebSphere MQ.

Les classes WebSphere MQ pour JMS utilisent les fonctions et les algorithmes fournis avec WebSphere MQ version 6.0. Si vous souhaitez vous connecter à WebSphere Event Broker ou WebSphere Message Broker à l'aide de WebSphere MQ Enterprise Transport, vous devez utiliser ce mode. Vous pouvez vous connecter à un gestionnaire de files d'attente WebSphere MQ Version 7.0 à l'aide de ce mode, mais aucune des nouvelles fonctions d'un gestionnaire de files d'attente version 7.0 n'est utilisée, par exemple, la lecture anticipée ou la diffusion en flux.

non spécifié
Il s'agit de la valeur par défaut et le texte réel est "non spécifié".

Cette valeur est affectée à une fabrique de connexions créée à l'aide d'une version antérieure des classes WebSphere MQ for JMS dans JNDI lorsque la fabrique de connexions est utilisée avec la nouvelle version des classes WebSphere MQ for JMS. L'algorithme ci-dessous sert à déterminer le mode de fonctionnement utilisé. Cet algorithme est utilisé lorsque la méthode createConnection() est appelée et utilise d'autres aspects de la fabrique de connexions pour déterminer si le mode normal du fournisseur de messagerie WebSphere MQ ou WebSphere MQ est requis.

- Tout d'abord, une tentative d'utilisation du mode normal du fournisseur de messagerie WebSphere MQ est effectuée.
- Si le gestionnaire de files d'attente connecté n'est pas WebSphere MQ Version 7.0, la connexion est fermée et le mode de migration du fournisseur de messagerie WebSphere MQ est utilisé à la place.
- Si la propriété SHARECNV () sur le canal de connexion serveur est définie sur 0, la connexion est fermée et le mode de migration du fournisseur de messagerie WebSphere MQ est utilisé à la place.
- Si BROKERVER est défini sur 1 ou sur la nouvelle valeur "unspecified" par défaut, le mode normal du fournisseur de messagerie WebSphere MQ continue d'être utilisé. Par conséquent, toute opération de publication / abonnement utilise les nouvelles fonctions WebSphere MQ V7.0 . Si WebSphere

Event Broker ou WebSphere Message Broker sont utilisés en mode compatibilité (et que vous souhaitez utiliser la fonction de publication / abonnement de la version 6.0 plutôt que la fonction de publication / abonnement de la version 7 de WebSphere MQ), définissez PROVIDERVERSION sur 6 pour vous assurer que le mode de migration du fournisseur de messagerie WebSphere MQ est utilisé.

PROXYHOSTNAME

Nom d'hôte ou adresse IP du système sur lequel le serveur proxy s'exécute lors de l'utilisation d'une connexion en temps réel à un courtier via un serveur proxy.

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: PROXYHOSTNAME

Nom abrégé de l'outil d'administration JMS: PHOST

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setProxyHostName()
- MQConnectionFactory.getProxyHostName()

Valeurs

null
Nom d'hôte du serveur proxy. Il s'agit de la valeur par défaut.

PROXYPORT

Numéro du port sur lequel le serveur proxy est à l'écoute lors de l'utilisation d'une connexion en temps réel à un courtier via un serveur proxy.

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: PROXYPORT

Nom abrégé de l'outil d'administration JMS: PPORT

Accès par programme

Configurateurs / méthodes d'accès get

MQConnectionFactory.setProxyPort ()

MQConnectionFactory.getProxyPort ()

Valeurs

443
Numéro de port du serveur proxy. Il s'agit de la valeur par défaut.

PUBACKINT

Nombre de messages publiés par un diffuseur de publications avant que WebSphere MQ classes for JMS demande un accusé de réception au courtier.

Lorsque vous réduisez la valeur de cette propriété, WebSphere MQ classes for JMS demande des accusés de réception plus souvent, par conséquent, les performances du diffuseur de publications diminuent. Lorsque vous augmentez la valeur, les classes WebSphere MQ pour JMS prennent plus de temps à émettre une exception en cas d'échec du courtier. Cette propriété est pertinente uniquement si TRANSPORT a la valeur BIND ou CLIENT.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: PROXYPORT

Nom abrégé de l'outil d'administration JMS: PPORT

Accès par programme

Configurateurs / méthodes d'accès get

MQConnectionFactory.setPubAckInterval()

MQConnectionFactory.getPubAckInterval()

Valeurs

25
Tout entier positif peut être la valeur par défaut.

PUTASYNCALLOWED

Cette propriété détermine si les expéditeurs de message sont autorisés à utiliser les insertions asynchrones pour envoyer des messages à cette destination.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: PUTASYNCALLOWED

Nom abrégé de l'outil d'administration JMS: PAALD

Accès par programme

Configurateurs / méthodes d'accès get

MQDestination.setPutAsyncAllowed()

MQDestination.getPutAsyncAllowed()

Valeurs

AS_DEST
Détermine si les insertions asynchrones sont autorisées conformément à la définition de rubrique ou de file d'attente. Il s'agit de la valeur par défaut.

AS_Q_DEF
Détermine si les insertions asynchrones sont autorisées conformément à la définition de file d'attente.

AS_TOPIC_DEF
Détermine si les insertions asynchrones sont autorisées conformément à la définition de rubrique.

NO
Les insertions asynchrones ne sont pas autorisées.

YES
Les insertions asynchrones sont autorisées.

QMANAGER

Nom du gestionnaire de files d'attente auquel établir la connexion.

Toutefois, si votre application utilise une table de définition de canal du client pour se connecter à un gestionnaire de files d'attente, voir Utilisation d'une table de définition de canal du client avec WebSphere MQ classes for JMS.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, XAConnectionFactory, XAQueueConnection Factory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: QMANAGER

Nom abrégé de l'outil d'administration JMS: QMGR

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory. GestionnairesetQueue()
- MQConnectionFactory. GestionnairegetQueue()

Valeurs

"" (chaîne vide)
Toute chaîne peut être la valeur par défaut.

QUEUE

Nom de la destination de file d'attente JMS. Correspond au nom de la file d'attente utilisée par le gestionnaire de files d'attente.

Objets applicables

File d'attente

Nom long de l'outil d'administration JMS: QUEUE

Nom abrégé de l'outil d'administration JMS: QU

Valeurs

N'importe quelle chaîne
Tout nom de file d'attente IBM WebSphere MQ valide.

Concepts associés
Règles de dénomination des objets IBM WebSphere MQ

READAHEADALLOWED

Cette propriété détermine si les consommateurs de messages et les navigateurs de file d'attente sont autorisés à utiliser la lecture anticipée pour extraire des messages non persistants de cette destination dans une mémoire tampon interne avant de les recevoir.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: READAHEADALLOWED

Nom abrégé de l'outil d'administration JMS: RAALD

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setReadAheadAllowed()
- MQDestination.getReadAheadAllowed()

Valeurs

AS_DEST
Détermine si la lecture anticipée est autorisée conformément à la définition de file d'attente ou de rubrique. Il s'agit de la valeur par défaut dans les outils d'administration.

Utilisez WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_DEST dans les programmes.

AS_Q_DEF
Détermine si la lecture anticipée est autorisée conformément à la définition de file d'attente.

Utilisez WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_Q_DEF dans les programmes.

AS_TOPIC_DEF
Détermine si la lecture anticipée est autorisée conformément à la définition de rubrique.

Utilisez WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_TOPIC_DEF dans les programmes.

NO
La lecture anticipée n'est pas autorisée.

Utilisez WMQConstants.WMQ_READ_AHEAD_ALLOWED_DISABLED dans les programmes.

YES
La lecture anticipée est autorisée.

Utilisez WMQConstants.WMQ_READ_AHEAD_ALLOWED_ENABLED dans les programmes.

READAHEADCLOSEPOLICY

Pour les messages distribués à un programme d'écoute de message asynchrone, qu'arrive-t-il aux messages de la mémoire tampon de lecture anticipée interne lorsque le consommateur de message est fermé?

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: READAHEADCLOSEPOLICY

Nom abrégé de l'outil d'administration JMS: RACP

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setReadAheadClosePolicy()
- MQDestination.getReadAheadClosePolicy()

Valeurs

DISTRIBUER
Tous les messages de la mémoire tampon de lecture anticipée interne sont distribués au programme d'écoute des messages de l'application avant d'être renvoyés. Il s'agit de la valeur par défaut dans les outils d'administration.

Utilisez WMQConstants.WMQ_READ_AHEAD_DELIVERALL dans les programmes.

EN COURS DE LIVRAISON
Seul l'appel du programme d'écoute des messages se termine avant d'être renvoyé, laissant potentiellement des messages dans la mémoire tampon interne de lecture anticipée qui sont ensuite supprimés.

Utilisez WMQConstants.WMQ_READ_AHEAD_DELIVERCURRENT dans les programmes.

RECEIVECCSID

Propriété de destination qui définit le CCSID cible pour la conversion des messages du gestionnaire de files d'attente. La valeur est ignorée sauf si RECEIVECONVERSION est défini sur `WMQ_RECEIVE_CONVERSION_QMGR`

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: RECEIVECCSID

Nom abrégé de l'outil d'administration JMS: RCCS

Accès par programme

Méthodes d'accès set / Getters

- `MQDestination.setReceiveCCSID`
- `MQDestination.getReceiveCCSID`

Valeurs

`WMQConstants.WMQ_RECEIVE_CCSID_JVM_DEFAULT`
0 -Utiliser la machine virtuelle Java `Charset.defaultCharset`

1208
UTF-8

ccsid
Identificateur de jeu de caractères codés pris en charge.

RECEIVECONVERSION

Propriété de destination qui détermine si la conversion de données sera effectuée par le gestionnaire de files d'attente.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: RECEIVECONVERSION

Nom abrégé de l'outil d'administration JMS: RCNV

Accès par programme

Méthodes d'accès set / Getters

- `MQDestination.setReceiveConversion`
- `MQDestination.getReceiveConversion`

Valeurs

WMQConstants.WMQ_RECEIVE_CONVERSION_CLIENT_MSG
1 -effectue uniquement la conversion de données sur le client JMS. La valeur par défaut est comprise entre V7.0et 7.0.1.5.

WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR
2 -Effectuer la conversion de données sur le gestionnaire de files d'attente avant d'envoyer un message au client. Valeur par défaut (et uniquement) comprise entre V7.0 et V7.0.1.4 inclus, sauf si l'APAR IC72897 est appliqué.

RECEIVEISOLATION

Cette propriété détermine si un abonné peut recevoir des messages qui n'ont pas été validés dans la file d'attente de l'abonné.

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: RECEIVEISOLATION

Nom abrégé de l'outil d'administration JMS: RCVISOL

Valeurs

VALIDE
Un abonné reçoit uniquement les messages de la file d'attente d'abonné qui ont été validés. Il s'agit de la valeur par défaut dans les outils d'administration.

Utilisez WMQConstants.WMQ_RCVISOL_COMMITTED dans les programmes.

NON VALIDE
Un abonné peut recevoir des messages qui n'ont pas été validés dans la file d'attente de l'abonné.

Utilisez WMQConstants.WMQ_RCVISOL_UNCOMMITTED dans les programmes.

RECEXIT

Identifie un exit de réception de canal ou une séquence d'exits de réception à exécuter successivement.

Une configuration supplémentaire peut être nécessaire pour que le IBM WebSphere MQ classes for JMS puisse localiser les exits de réception. Pour plus d'informations, voir Configuration des classes IBM WebSphere MQ pour JMS afin d'utiliser des exits de canal.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: RECEXIT

Nom abrégé de l'outil d'administration JMS: RCX

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setReceiveExit ()
- MQConnectionFactory.getReceiveExit ()

Valeurs

null
Chaîne comprenant un ou plusieurs éléments séparés par des virgules, où chaque élément est:

- Nom d'une classe qui implémente l'interface WMQReceiveExit (pour un exit de réception de canal écrit en Java).
- Chaîne au format *libraryName(entryPointName)* (pour un exit de réception de canal non écrit en Java).

Il s'agit de la valeur par défaut.

RECEXITINIT

Données utilisateur transmises aux exits de réception de canal lorsqu'ils sont appelés.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: RECEXITINIT

Nom abrégé de l'outil d'administration JMS: RCXI

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setReceiveExitInit()
- MQConnectionFactory.getReceiveExitInit()

Valeurs

null
Chaîne comprenant un ou plusieurs éléments de données utilisateur séparés par des virgules. Il s'agit de la valeur par défaut.

REPLYTOSTYLE

Détermine comment la zone JMSReplyTo d'un message reçu est construite.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: REPLYTOSTYLE

Nom abrégé de l'outil d'administration JMS: RTOST

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setReplyToStyle()
- MQConnectionFactory.getReplyToStyle()

Valeurs

Valeur par défaut
Equivalent à `MQMD`.

RFH2
Utilisez la valeur fournie dans l'en-tête RFH2 . Si une valeur JMSReplyTo a été définie dans l'application émettrice, utilisez cette valeur.

MQMD
Utilisez la valeur fournie par MQMD. Ce comportement est équivalent au comportement par défaut de WebSphere MQ version 6.0.2.4 et 6.0.2.5.

Si la valeur JMSReplyTo définie par l'application émettrice ne contient pas de nom de gestionnaire de files d'attente, le gestionnaire de files d'attente de réception insère son propre nom dans le MQMD. Si vous définissez ce paramètre sur `MQMD`, la file d'attente de réponse que vous utilisez se trouve sur le gestionnaire de files d'attente de réception. Si vous définissez ce paramètre sur RFH2, la file d'attente de réponse que vous utilisez se trouve sur le gestionnaire de files d'attente spécifié dans RFH2 du message envoyé, comme défini à l'origine par l'application émettrice.

Si la valeur JMSReplyTo définie par l'application émettrice contient un nom de gestionnaire de files d'attente, la valeur de ce paramètre n'est pas importante car MQMD et RFH2 contiennent la même valeur.

RESCANINT

Lorsqu'un consommateur de message du domaine point à point utilise un sélecteur de message pour sélectionner les messages qu'il souhaite recevoir, WebSphere MQ classes for JMS recherche dans la file d'attente WebSphere MQ les messages appropriés dans la séquence déterminée par l'attribut `MsgDeliverySequence` de la file d'attente.

Une fois que WebSphere MQ classes for JMS a trouvé un message approprié et l'a livré au destinataire, WebSphere MQ classes for JMS reprend la recherche du message approprié suivant à partir de sa position actuelle dans la file d'attente. WebSphere MQ classes for JMS poursuit la recherche dans la file d'attente de cette manière jusqu'à ce qu'elle atteigne la fin de la file d'attente ou jusqu'à ce que l'intervalle de temps en millisecondes, déterminé par la valeur de cette propriété, ait expiré. Dans chaque cas, les classes WebSphere MQ pour JMS reviennent au début de la file d'attente pour poursuivre la recherche et un nouvel intervalle de temps commence.

Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Nom long de l'outil d'administration JMS: RESCANINT

Nom abrégé de l'outil d'administration JMS: RINT

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setRescanIntervalle ()
- MQConnectionFactory.getRescanIntervalle ()

Valeurs

5 000
Tout entier positif peut être la valeur par défaut.

SECEXIT

Identifie un exit de sécurité de canal.

Une configuration supplémentaire peut être nécessaire pour que le IBM WebSphere MQ classes for JMS puisse localiser les exits de sécurité. Pour plus d'informations, voir Configuration des classes IBM WebSphere MQ pour JMS afin d'utiliser des exits de canal.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SECEXIT

Nom abrégé de l'outil d'administration JMS: SXC

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSecurityExit ()
- MQConnectionFactory.getSecurityExit ()

Valeurs

null
Nom d'une classe qui implémente l'interface WMQSecurityExit (pour un exit de sécurité de canal écrit en Java).

Chaîne au format *libraryName(entryPointName)* (pour un exit de sécurité de canal non écrit en Java).

SECEXITINIT

Données utilisateur transmises à un exit de sécurité de canal lors de l'appel.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SECEXITINIT

Nom abrégé de l'outil d'administration JMS: SCXI

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSecurityExitInit()
- MQConnectionFactory.getSecurityExitInit()

Valeurs

null
Toute chaîne peut être la valeur par défaut.

SENDCHECKCOUNT

Nombre d'appels d'envoi à autoriser entre deux vérifications d'erreurs d'insertion asynchrone, au sein d'une même session JMS non transactionnelle.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SENDCHECKCOUNT

Nom abrégé de l'outil d'administration JMS: SCC

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSendCheckCount()
- MQConnectionFactory.getSendCheckCount()

Valeurs

null
Toute chaîne peut être la valeur par défaut.

SENDEXIT

Identifie un exit d'émission de canal ou une séquence d'exits d'émission à exécuter successivement.

Une configuration supplémentaire peut être nécessaire pour que le IBM WebSphere MQ classes for JMS puisse localiser les exits d'émission. Pour plus d'informations, voir Configuration des classes IBM WebSphere MQ pour JMS afin d'utiliser des exits de canal.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SENDEXIT

Nom abrégé de l'outil d'administration JMS: SDX

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSendExit ()
- MQConnectionFactory.getSendExit ()

Valeurs

null
Toute chaîne comprenant un ou plusieurs éléments séparés par des virgules, où chaque élément est:

- Nom d'une classe qui implémente l'interface WMQSendExit (pour un exit d'émission de canal écrit en Java).
- Chaîne au format *libraryName(entryPointName)* (pour un exit d'émission de canal non écrit en Java).
-

Il s'agit de la valeur par défaut.

SENDEXITINIT

Données utilisateur qui sont transmises aux exits d'émission une fois ceux-ci appelés.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SENDEXITINIT

Nom abrégé de l'outil d'administration JMS: SDXI

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSendExitInit()
- MQConnectionFactory.getSendExitInit()

Valeurs

null
Toute chaîne comprenant un ou plusieurs éléments de données utilisateur séparés par des virgules peut être la valeur par défaut.

SHARECONVALLOWED

Cette propriété détermine si une connexion client peut partager son socket avec d'autres connexions JMS de niveau supérieur à partir du même processus vers le même gestionnaire de files d'attente, si les définitions de canal correspondent.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SHARECONVALLOWED

Nom abrégé de l'outil d'administration JMS: SCALD

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setShareConvAllowed()
- MQConnectionFactory.getShareConvAllowed()

Valeurs

YES
Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ_SHARE_CONV_ALLOWED_YES.

NO
Cette valeur concerne les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ_SHARE_CONV_ALLOWED_NO.

SPARSESUBS

Contrôle la stratégie d'extraction de message d'un objet TopicSubscriber .

Objets applicables

ConnectionFactory, TopicConnectionFactory

Nom long de l'outil d'administration JMS: SPARSESUBS

Nom abrégé de l'outil d'administration JMS: SSUBS

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSparseAbonnements ()
- MQConnectionFactory.getSparseAbonnements ()

Valeurs

NO
Les abonnements reçoivent des messages correspondants fréquents. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez false.

YES
Les abonnements reçoivent des messages correspondants peu fréquents. Cette valeur nécessite que la file d'attente d'abonnement puisse être ouverte pour l'exploration.

Pour les programmes, utilisez true.

SSLCIPHERSUITE

CipherSuite à utiliser pour une connexion SSL.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SSLCIPHERSUITE

Nom abrégé de l'outil d'administration JMS: SCPHS

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSSLCipherSuite ()
- MQConnectionFactory. SuitegetSSLCipher()

Valeurs

null
Il s'agit de la valeur par défaut. Pour plus d'informations, voir Propriétés SSL des objets JMS.

SSLCRL

Serveurs CRL pour vérifier la révocation de certificat SSL.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SSLCRL

Nom abrégé de l'outil d'administration JMS: SCRL

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSSLCertMagasins ()
- MQConnectionFactory.getSSLCertMagasins ()

Valeurs

null
Liste d'URL LDAP séparées par des espaces. Il s'agit de la valeur par défaut. Pour plus d'informations, voir Propriétés SSL des objets JMS.

SSLFIPSREQUIRED

Cette propriété détermine si une connexion SSL doit utiliser une CipherSuite prise en charge par le fournisseur IBM Java JSSE FIPS (IBMJSSEFIPS).

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SSLFIPSREQUIRED

Nom abrégé de l'outil d'administration JMS: SFIPS

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSSLFipsObligatoire ()
- MQConnectionFactory.getSSLFipsObligatoire ()

Valeurs

NO
Une connexion SSL peut utiliser n'importe quelle CipherSuite qui n'est pas prise en charge par le fournisseur IBM Java JSSE FIPS (IBMJSSEFIPS).

Il s'agit de la valeur par défaut. Dans les programmes, utilisez false.

YES
Une connexion SSL doit utiliser une CipherSuite prise en charge par IBMJSSEFIPS.

Dans les programmes, utilisez true.

SSLPEERNAME

Pour SSL, squelette de *nom distinctif* qui doit correspondre à celui fourni par le gestionnaire de files d'attente.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SSLPEERNAME

Nom abrégé de l'outil d'administration JMS: SPEER

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSSLPeerNom ()
- MQConnectionFactory.getSSLPeerNom ()

Valeurs

null
Il s'agit de la valeur par défaut. Pour plus d'informations, voir Propriétés SSL des objets JMS.

SSLRESETCOUNT

Pour SSL, le nombre total d'octets envoyés et reçus par une connexion avant que la clé secrète utilisée pour le chiffrement ne soit renégociée.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SSLRESETCOUNT

Nom abrégé de l'outil d'administration JMS: SRC

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSSLResetNombre ()
- MQConnectionFactory.getSSLResetNombre ()

Valeurs

0
Zéro, ou tout entier positif inférieur ou égal à 999, 999, 999. Il s'agit de la valeur par défaut. Pour plus d'informations, voir Propriétés SSL des objets JMS.

STATREFRESHINT

Intervalle, en millisecondes, entre les actualisations de la transaction à exécution longue qui détecte lorsqu'un abonné perd sa connexion au gestionnaire de files d'attente.

Cette propriété est pertinente uniquement si la valeur de la substance est QUEUE.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: STATREFRESHINT

Nom abrégé de l'outil d'administration JMS: SRI

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setStatusRefreshInterval()
- MQConnectionFactory.getStatusRefreshInterval()

Valeurs

6000
Tout entier positif peut être la valeur par défaut. Pour plus d'informations, voir Propriétés SSL des objets JMS.

SUBSTORE

Où WebSphere MQ classes for JMS stocke les données persistantes relatives aux abonnements actifs.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: CONTENU

Nom abrégé de l'outil d'administration JMS: SS

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSubscriptionStore ()
- MQConnectionFactory.getSubscriptionStore ()

Valeurs

COURTIER
Utilisez le magasin d'abonnements basé sur un courtier pour conserver les détails des abonnements. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ_SUBSTORE_BROKER.

MIGRATE
Transférez les informations d'abonnement du magasin d'abonnements basé sur la file d'attente vers le magasin d'abonnements basé sur le courtier.

Pour les programmes, utilisez WMQConstants.WMQ_SUBSTORE_MIGRATE.

QUEUE
Utilisez le magasin d'abonnements basé sur une file d'attente pour conserver les détails des abonnements.

Pour les programmes, utilisez WMQConstants.WMQ_SUBSTORE_QUEUE.

SYNCPOINTALLGETS

Cette propriété détermine si toutes les extractions doivent être effectuées sous le point de synchronisation.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: SYNCPOINTALLGETS

Nom abrégé de l'outil d'administration JMS: SPAG

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setSyncpointAllGets()
- MQConnectionFactory.getSyncpointAllGets()

Valeurs

Non
Il s'agit de la valeur par défaut.

Oui

TARGCLIENT

Cette propriété détermine si le format WebSphere MQ RFH2 est utilisé pour échanger des informations avec les applications cible.

Objets applicables

File d'attente, Rubrique

Nom long de l'outil d'administration JMS: TARGCLIENT

Nom abrégé de l'outil d'administration JMS: TC

Accès par programme

Configurateurs / méthodes d'accès get

- MQDestination.setTargetClient()
- MQDestination.getTargetClient()

Valeurs

JMS
La cible du message est une application JMS. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ_CLIENT_JMS_COMPLIANT.

MQ
La cible du message est une application WebSphere MQ non JMS.

Pour les programmes, utilisez WMQConstants.WMQ_CLIENT_NONJMS_MQ.

TARGCLIENTMATCHING

Cette propriété détermine si un message de réponse, envoyé à la file d'attente identifiée par la zone d'en-tête JMSReplyTo d'un message entrant, a un en-tête MQRFH2 uniquement si le message entrant a un en-tête MQRFH2 .

Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Nom long de l'outil d'administration JMS: TARGCLIENTMATCHING

Nom abrégé de l'outil d'administration JMS: TCM

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setTargetClientMatching()
- MQConnectionFactory.getTargetClientMatching()

Valeurs

YES
Si un message entrant ne possède pas d'en-tête MQRFH2 , la propriété TARGCLIENT de l'objet Queue dérivée de la zone d'en-tête JMSReplyTo du message est envoyée à MQ. Si le message comporte un en-tête MQRFH2 , la propriété TARGCLIENT est définie sur JMS à la place. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez true.

NO
La propriété TARGCLIENT de l'objet Queue dérivée de la zone d'en-tête JMSReplyTo d'un message entrant est toujours définie sur JMS.

Pour les programmes, utilisez false.

TEMPMODEL

Nom de la file d'attente modèle à partir de laquelle les files d'attente temporaires JMS sont créées.

Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Nom long de l'outil d'administration JMS: TEMPMODEL

Nom abrégé de l'outil d'administration JMS: TM

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setTemporaryModèle ()
- MQConnectionFactory.getTemporaryModèle ()

Valeurs

SYSTEM.DEFAULT.MODEL.QUEUE
Toute chaîne peut être la valeur par défaut.

TEMPQPREFIX

Préfixe utilisé pour former le nom d'une file d'attente dynamique WebSphere MQ .

Objets applicables

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Nom long de l'outil d'administration JMS: TEMPQPREFIX

Nom abrégé de l'outil d'administration JMS: TQP

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setTempQPrefix ()
- MQConnectionFactory.getTempQPrefix ()

Valeurs

" " (chaîne vide)
Le préfixe utilisé est `CSQ.*` sur z/OS et `AMQ.*` sur toutes les autres plateformes. Il s'agit des valeurs par défaut.

Préfixe de la file d'attente
Le préfixe de file d'attente correspond à toute chaîne conforme aux règles de formation du contenu de la zone *DynamicQName* dans un descripteur d'objet WebSphere MQ (structure MQOD), mais le dernier caractère non vide doit être un astérisque.

TEMPTOPICPREFIX

Lors de la création de rubriques temporaires, JMS génère une chaîne de rubrique au format '' TEMP/*TEMPTOPICPREFIX/unique_id*'' ou, si cette propriété est laissée avec la valeur par défaut, uniquement '' TEMP/*unique_id*''. La spécification d'un TEMPTOPICPREFIX non vide permet de définir des files d'attente modèles spécifiques pour la création de files d'attente gérées pour les abonnés à des rubriques temporaires créées sous cette connexion.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: TEMPTOPICPREFIX

Nom abrégé de l'outil d'administration JMS: TTP

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setTempTopicPrefix()
- MQConnectionFactory.getTempTopicPrefix()

Valeurs

Toute chaîne non nulle composée uniquement de caractères valides pour une chaîne de rubrique WebSphere MQ . La valeur par défaut est '' '' (chaîne vide).

TOPIC

Nom de la destination de rubrique JMS. Cette valeur est utilisée par le gestionnaire de files d'attente en tant que chaîne de rubrique d'une publication ou d'un abonnement.

Objets applicables

Topic

Nom long de l'outil d'administration JMS: TOPIC

Nom abrégé de l'outil d'administration JMS: TOP

Valeurs

N'importe quelle chaîne

Chaîne qui forme une chaîne de rubrique IBM WebSphere MQ valide. Lorsque vous utilisez IBM WebSphere MQ comme fournisseur de messagerie avec WebSphere Application Server, indiquez une valeur correspondant au nom sous lequel la rubrique est connue dans WebSphere Application Serverà des fins d'administration.

Concepts associés

Chaînes de rubrique

TRANSPORT

Nature d'une connexion à un gestionnaire de files d'attente ou à un courtier.

Objets applicables

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: TRANSPORT

Nom abrégé de l'outil d'administration JMS: TRAN

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setTransportType ()
- MQConnectionFactory.getTransportType ()

Valeurs

BIND
Pour une connexion à un gestionnaire de files d'attente en mode liaisons. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ_CM_BINDINGS.

client
Pour une connexion à un gestionnaire de files d'attente en mode client.

Pour les programmes, utilisez WMQConstants.WMQ_CM_CLIENT.

Directe
Pour une connexion en temps réel à un courtier qui n'utilise pas de tunnellisation HTTP.

Pour les programmes, utilisez WMQConstants.WMQ_CM_DIRECT_TCPIP.

DIRECTIVE THTTP
Pour une connexion en temps réel à un courtier à l'aide de la tunnellisation HTTP. Seul HTTP 1.0 est pris en charge.

Pour les programmes, utilisez WMQConstants.WMQ_CM_DIRECT_HTTP.

WILDCARDFORMAT

Cette propriété détermine la version de la syntaxe générique à utiliser.

Objets applicables

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Nom long de l'outil d'administration JMS: WILDCARDFORMAT

Nom abrégé de l'outil d'administration JMS: WCFMT

Accès par programme

Configurateurs / méthodes d'accès get

- MQConnectionFactory.setWildCardFormat()
- MQConnectionFactory.getWildCardFormat()

Valeurs

TOPIC_ONLY
Reconnaît uniquement les caractères génériques de niveau rubrique, tels qu'ils sont utilisés dans la version 2 du courtier. Il s'agit de la valeur par défaut pour les outils d'administration.

Pour les programmes, utilisez WMQConstants.WMQ_WILDCARD_TOPIC_ONLY.

CHAR_ONLY (UNIQUEMENT)
Reconnaît uniquement les caractères génériques, tels qu'ils sont utilisés dans la version 1 du courtier.

Pour les programmes, utilisez WMQConstants.WMQ_WILDCARD_CHAR_ONLY.

Dépendances entre les propriétés des objets WebSphere MQ classes for JMS

La validité de certaines propriétés dépend des valeurs particulières d'autres propriétés.

Cette dépendance peut se produire dans les groupes de propriétés suivants:

- Propriétés du client
- Propriétés d'une connexion en temps réel à un courtier
- Chaînes d'initialisation d'exit

Propriétés du client

Pour une connexion à un gestionnaire de files d'attente, les propriétés suivantes ne sont pertinentes que si TRANSPORT a la valeur CLIENT:

- HOSTNAME
- PORT
- Canal
- LOCALADDRESS
- CCDTURL
- CCSID
- COMPHDR
- COMPMSG
- RECEXIT
- RECEXITINIT
- SECEXIT
- SECEXITINIT
- SENDEXIT
- SENDEXITINIT
- SHARECONVALLOWED
- SSLCIPHERSUITE
- SSLCRL
- SSLFIPSREQUIRED
- SSLPEERNAME
- SSLRESETCOUNT
- APPLICATIONNAME

Vous ne pouvez pas définir de valeurs pour ces propriétés à l'aide de l'outil d'administration si TRANSPORT a la valeur BIND.

Si TRANSPORT a la valeur CLIENT, la valeur par défaut de la propriété BROKERVER est V1 et la valeur par défaut de la propriété PORT est 1414. Si vous définissez explicitement la valeur de BROKERVER ou PORT, une modification ultérieure de la valeur de TRANSPORT ne remplace pas vos choix.

Propriétés d'une connexion en temps réel à un courtier

Seules les propriétés suivantes sont pertinentes si TRANSPORT a la valeur DIRECT ou DIRECTION THTTP:

- BROKERVER
- IDCLIENT
- DESCRIPTION
- DIRECTAUTH
- HOSTNAME

- LOCALADDRESS
- MAXBUFFSIZE
- MULTICAST (pris en charge uniquement pour DIRECT)
- PORT
- PROXYHOSTNAME (pris en charge uniquement pour DIRECT)
- PROXYPORT (pris en charge uniquement pour DIRECT)

Si TRANSPORT a la valeur DIRECT ou DIRECTIVE THTTP, la valeur par défaut de la propriété BROKERVER est V2et la valeur par défaut de la propriété PORT est 1506. Si vous définissez explicitement la valeur de BROKERVER ou PORT, une modification ultérieure de la valeur de TRANSPORT ne remplace pas vos choix.

Chaînes d'initialisation d'exit
Ne définissez aucune des chaînes d'initialisation d'exit sans indiquer le nom d'exit correspondant. Les propriétés d'initialisation de l'exit sont les suivantes:

- RECEXITINIT
- SECEXITINIT
- SENDEXITINIT

Par exemple, si vous spécifiez `RECEXITINIT(myString)` sans spécifier `RECEXIT(some.exit.classname)` , une erreur se produit.

Propriété ENCODING

La propriété ENCODING comprend trois sous-propriétés, en douze combinaisons possibles.

Les valeurs valides que la propriété `ENCODING` peut prendre sont construites à partir des trois sous-propriétés:

Codage d'entier
Normale ou inversée

Codage décimal
Normale ou inversée

codage en virgule flottante
IEEE normal, IEEE inversé ou z/OS

La propriété `ENCODING` est exprimée sous la forme d'une chaîne de trois caractères avec la syntaxe suivante:

```
{N|R}{N|R}{N|R|3}
```

Dans cette chaîne:

- N indique normal
- R indique que la valeur est inversée
- 3 indique z/OS
- Le premier caractère représente le *codage d'entier*
- Le second caractère représente le *codage décimal*
- Le troisième caractère représente le *codage en virgule flottante*

Fournit un ensemble de douze valeurs possibles pour la propriété `ENCODING` .

Il existe une valeur supplémentaire, la chaîne `NATIVE`, qui définit les valeurs de codage appropriées pour la plateforme Java.

Les exemples suivants montrent des combinaisons valides pour ENCODING:

```
ENCODING(NNR)
ENCODING(NATIVE)
ENCODING(RR3)
```

Propriétés SSL des objets JMS

Activez le chiffrement SSL (Secure Sockets Layer) à l'aide de la propriété SSLCIPHERSUITE. Vous pouvez ensuite modifier les caractéristiques du chiffrement SSL à l'aide de plusieurs autres propriétés.

Lorsque vous spécifiez TRANSPORT (CLIENT), vous pouvez activer la communication chiffrée SSL (Secure Sockets Layer) à l'aide de la propriété SSLCIPHERSUITE. Définissez cette propriété sur une CipherSuite valide fournie par votre fournisseur JSSE ; elle doit correspondre au CipherSpec nommé sur le canal SVRCONN nommé par la propriété CHANNEL.

Toutefois, les CipherSpecs (comme indiqué sur le canal SVRCONN) et les CipherSuites (comme indiqué sur les objets ConnectionFactory) utilisent des schémas de dénomination différents pour représenter les mêmes algorithmes de chiffrement SSL. Si un nom CipherSpec reconnu est spécifié dans la propriété SSLCIPHERSUITE, JMSAdmin émet un avertissement et mappe le CipherSpec à son équivalent CipherSuite. Voir SSL CipherSpecs et CipherSuites dans JMS pour obtenir la liste des CipherSpecs reconnus par WebSphere MQ et JMSAdmin.

Si vous avez besoin d'une connexion pour utiliser une CipherSuite prise en charge par le fournisseur Java JSSE FIPS IBM (IBMJSSEFIPS), définissez la propriété SSLFIPSREQUIRED de la fabrique de connexions sur YES. La valeur par défaut de cette propriété est NO, ce qui signifie qu'une connexion peut utiliser n'importe quelle CipherSuiteprise en charge. La propriété est ignorée si SSLCIPHERSUITE n'est pas défini.

Le paramètre SSLPEERNAME correspond au format du paramètre SSLPEER, qui peut être défini sur les définitions de canal. Il s'agit d'une liste de paires de nom et de valeur d'attribut séparées par des virgules ou des points-virgules. Exemple :

```
SSLPEERNAME(CN=QMGR.*, OU=IBM, OU=WEBSPHERE)
```

L'ensemble de noms et de valeurs constitue un *nom distinctif*. Pour plus de détails sur les noms distinctifs et leur utilisation avec WebSphere MQ, voir Sécurité.

L'exemple donné vérifie le certificat d'identification présenté par le serveur lors de la connexion. Pour que la connexion aboutisse, le certificat doit avoir un nom usuel commençant par QMGR., et doit avoir au moins deux noms d'unité organisationnelle, le premier étant IBM et le second étant WEBSPHERE. La vérification n'est pas sensible à la casse.

Si SSLPEERNAME n'est pas défini, aucune vérification de ce type n'est effectuée. SSLPEERNAME est ignoré si SSLCIPHERSUITE n'est pas défini.

La propriété SSLCRL spécifie zéro ou plusieurs serveurs CRL (Certificate Revocation List). L'utilisation de cette propriété nécessite une machine virtuelle Java sur Java 2 v1.4. Il s'agit d'une liste délimitée par des espaces d'entrées au format suivant:

```
ldap://hostname:[port]
```

éventuellement suivi d'un / unique. Si *port* est omis, le port LDAP par défaut 389 est utilisé. Lors de la connexion, le certificat SSL présenté par le serveur est vérifié par rapport aux serveurs CRL spécifiés. Pour plus d'informations sur la sécurité CRL, voir Sécurité .

Si SSLCRL n'est pas défini, aucune vérification de ce type n'est effectuée. SSLCRL est ignoré si SSLCIPHERSUITE n'est pas défini.

La propriété SSLRESETCOUNT représente le nombre total d'octets envoyés et reçus par une connexion avant que la clé secrète utilisée pour le chiffrement ne soit renégociée. Le nombre d'octets envoyés est le nombre avant chiffrement et le nombre d'octets reçus est le nombre après déchiffrement. Le nombre

d'octets inclut également les informations de contrôle envoyées et reçues par WebSphere MQ classes for JMS.

Par exemple, pour configurer un objet ConnectionFactory qui peut être utilisé pour créer une connexion via un canal MQI activé par SSL avec une clé secrète qui est renégociée après 4 Mo de données transmises, exécutez la commande suivante à JMSAdmin:

```
ALTER CF(my.cf) SSLRESETCOUNT(4194304)
```

Si la valeur de SSLRESETCOUNT est zéro, qui est la valeur par défaut, la clé secrète n'est jamais renégociée. La propriété SSLRESETCOUNT est ignorée si SSLCIPHERSUITE n'est pas défini.

Remarques

:NONE.

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service IBM puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM EMEA Director of Licensing
IBM Corporation
Tour Descartes
Armonk, NY 10504-1785
U.S.A.

Pour toute demande d'informations relatives au jeu de caractères codé sur deux octets, contactez le service de propriété intellectuelle IBM ou envoyez vos questions par courrier à l'adresse suivante :

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japon

Le paragraphe suivant ne s’applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales. LE PRESENT DOCUMENT EST LIVRE "EN L'ETAT" SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Corporation
Coordinateur d'interopérabilité logicielle, département 49XA
3605 Autoroute 52 N

Rochester, MN 55901
U.S.A.

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans le présent document et tous les éléments sous disponibles s'y rapportant sont fournis par IBM conformément aux dispositions du Contrat sur les produits et services IBM, aux Conditions Internationales d'Utilisation de Logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

Licence sur les droits d'auteur :

Le présent logiciel contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

Documentation sur l'interface de programmation

Les informations d'interface de programmation, si elles sont fournies, sont destinées à vous aider à créer un logiciel d'application à utiliser avec ce programme.

Ce manuel contient des informations sur les interfaces de programmation prévues qui permettent au client d'écrire des programmes pour obtenir les services de IBM WebSphere MQ.

Toutefois, lesdites informations peuvent également contenir des données de diagnostic, de modification et d'optimisation. Ces données vous permettent de déboguer votre application.

Important : N'utilisez pas ces informations de diagnostic, de modification et d'optimisation en tant qu'interface de programmation car elles sont susceptibles d'être modifiées.

Marques

IBM, le logo IBM , ibm.com, sont des marques d' IBM Corporation dans de nombreux pays. La liste actualisée de toutes les marques d' IBM est disponible sur la page Web "Copyright and trademark

information"www.ibm.com/legal/copytrade.shtml. Les autres noms de produits et de services peuvent être des marques d'IBM ou d’autres sociétés.

Microsoft et Windows sont des marques de Microsoft Corporation aux Etats-Unis et/ou dans d'autres pays.

UNIX est une marque de The Open Group aux Etats-Unis et dans certains autres pays.

Linux est une marque de Linus Torvalds aux Etats-Unis et/ou dans certains autres pays.

Ce produit inclut des logiciels développés par le projet Eclipse (http://www.eclipse.org/).

Java ainsi que tous les logos et toutes les marques incluant Java sont des marques d'Oracle et/ou de ses sociétés affiliées.

Référence :

(1P) P/N: