9.3

IBM MQ Vývoj odkazů na aplikace

Poznámka

Než začnete používat tyto informace a produkt, který podporují, přečtěte si informace, které uvádí “Poznámky” na stránce 2191.

Toto vydání se vztahuje na verzi 9 vydání 3 produktu IBM® MQ a na všechna následná vydání a úpravy, není-li v nových vydáních uvedeno jinak.

Když odešlete informace na adresu IBM, udělujete IBM nevýhradní právo používat nebo distribuovat informace libovolným způsobem, který považuje za odpovídající, aniž by vám tím vznikl jakýkoliv závazek.

Obsah

Odkaz na vývoj aplikací

Odkazy uvedené v této části vám pomohou s vývojem aplikací IBM MQ .

- “Odkaz na aplikace rozhraní MQI” na stránce 7
- IBM i “IBM i Application Programming Reference (ILE/RPG) (Referenční příručka pro programování aplikací)” na stránce 991
- IBM i “Převod dat na IBM i” na stránce 1428
- “Uživatelské procedury, uživatelské procedury rozhraní API a odkaz na instalovatelné služby” na stránce 1448
- “Třídy a rozhraní produktu IBM MQ .NET” na stránce 1716
- “IBM MQ třídy C++” na stránce 1778
- “Vlastnosti objektů IBM MQ classes for JMS” na stránce 1887
- “Odkaz na systém zpráv REST API” na stránce 2158

Související úlohy

Vývoj aplikací

Související odkazy

Třídy IBM MQ pro knihovny Java

IBM MQ třídy pro JMS

Odkaz na aplikace rozhraní MQI

Odkazy uvedené v této části vám pomohou při vývoji aplikací rozhraní MQI (Message Queue Interface).

- “Příklady kódu” na stránce 8
- “Konstanty” na stránce 61
- “Datové typy použité v rozhraní MQI” na stránce 235
- “Volání funkcí” na stránce 628
- “Atributy objektů” na stránce 799
- “Návratové kódy” na stránce 874
- “Pravidla pro ověřování platnosti voleb rozhraní MQI” na stránce 875
- “Strojové kódování” na stránce 898
- “Volby sestavy a příznaky zpráv” na stránce 901
- “Uživatelská procedura převodu dat” na stránce 905
- “Vlastnosti určené jako prvky MQRFH2” na stránce 928
- “Převod kódové stránky” na stránce 936

Související pojmy

“Uživatelské procedury, uživatelské procedury rozhraní API a odkaz na instalovatelné služby” na stránce 1448

Informace v této části vám pomohou při vývoji uživatelských procedur, uživatelských procedur rozhraní API a instalovatelných aplikací služeb:

Související úlohy

Vývoj aplikací

Související odkazy

“Třídy a rozhraní produktu IBM MQ .NET” na stránce 1716

Třídy a rozhraní produktu IBM MQ .NET jsou uvedeny abecedně. Jsou popsány vlastnosti, metody a konstruktory.

"IBM MQ třídy C++" na stránce 1778
Třídy IBM MQ C++ zapouzdřují rozhraní MQI (IBM MQ Message Queue Interface). Existuje jeden soubor záhlaví C++, **imqi.hpp**, který pokrývá všechny tyto třídy.

IBM MQ Třídy pro knihovny Java
IBM MQ Třídy pro JMS

Příklady kódu

Použijte referenční informace v této sekci k provedení úloh, které odpovídají vašim obchodním potřebám.

Příklady jazyka C

Tato kolekce témat je většinou převzata z ukázkových aplikací IBM MQ for z/OS . Vztahují se na všechny platformy, s výjimkou případů, kdy je to uvedeno.

Připojení ke správci front

Tento příklad demonstruje, jak použít volání MQCONN pro připojení programu ke správci front v dávce z/OS .

Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BCA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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;
      }
      .
      .
      .
}
```

Odpojení od správce front

Tento příklad demonstruje, jak použít volání MQDISC k odpojení programu od správce front v dávce z/OS .

Proměnné použité v tomto extraktu kódu jsou ty, které byly nastaveny v “Připojení ke správci front” na stránce 8. Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BCA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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;
      }
.
```

Vytvoření dynamické fronty

Tento příklad demonstruje, jak použít volání MQOPEN k vytvoření dynamické fronty.

Tato extrakce je převzata z ukázkové aplikace správce pošty (program CSQ4TCD1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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;
}
.
```

Otevření existující fronty

Tento příklad demonstruje, jak použít volání MQOPEN k otevření fronty, která již byla definována.

Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BCA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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 */
```

Zavření fronty

Tento příklad demonstruje, jak použít volání MQCLOSE k zavření fronty.

Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BCA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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;
      }
.
```

Vložení zprávy pomocí příkazu MQPUT

Tento příklad demonstruje, jak použít volání MQPUT k vložení zprávy do fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ. Názvy a umístění ukázkových aplikací viz Ukázkové procedurální programy (platformy s výjimkou z/OS) z/OS a Ukázkové programy pro IBM MQ for 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 */
        }
}
```

Vložení zprávy pomocí MQPUT1

Tento příklad ukazuje, jak použít volání MQPUT1 k otevření fronty, vložení jedné zprávy do fronty a zavření fronty.

Tato extrakce je převzata z ukázkové aplikace Credit Check (program CSQ4CCB5) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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;
   }
   .
   .
   .
```

získávání zprávy

Tento příklad demonstruje, jak použít volání MQGET k odebrání zprávy z fronty.

Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BCA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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 */
```

Získání zprávy pomocí volby čekání

Tento příklad demonstruje, jak použít volbu wait volání MQGET.

Tento kód přijímá oříznuté zprávy. Tato extrakce je převzata z ukázkové aplikace Credit Check (program CSQ4CCB5) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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();
         }
         .
```

Získání zprávy pomocí signalizace

Signalizace je k dispozici pouze s produktem IBM MQ for z/OS .

Tento příklad ukazuje, jak použít volání MQGET k nastavení signálu, abyste byli upozorněni, když do fronty dorazí vhodná zpráva. Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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;
}
```

Zjišťování atributů objektu

Tento příklad demonstruje, jak použít volání MQINQ k dotazování na atributy fronty.

Tato extrakce je převzata z ukázkové aplikace Atributy fronty (program CSQ4CCC1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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 */
```

Nastavení atributů fronty

Tento příklad demonstruje, jak použít volání MQSET ke změně atributů fronty.

Tato extrakce je převzata z ukázkové aplikace Atributy fronty (program CSQ4CCC1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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 */
```

Načítání informací o stavu pomocí MQSTAT

Tento příklad demonstruje, jak vydat asynchronní příkaz MQPUT a načíst informace o stavu pomocí příkazu MQSTAT.

Tato extrakce je převzata z ukázkové aplikace volání MQSTAT (program amqsapt0). dodávané se systémy IBM MQ for Windows . Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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);
  }
```

Příklady v jazyce COBOL

Tato kolekce témat je převzata z ukázkových aplikací IBM MQ for z/OS . Vztahují se na všechny platformy, s výjimkou případů, kdy je to uvedeno.

Připojení ke správci front

Tento příklad demonstruje, jak použít volání MQCONN pro připojení programu ke správci front v dávce z/OS .

Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BVA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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.
     ⋮
```

Odpojení od správce front

Tento příklad demonstruje, jak použít volání MQDISC k odpojení programu od správce front v dávce z/OS .

Proměnné použité v tomto extraktu kódu jsou ty, které byly nastaveny v “Připojení ke správci front” na stránce 23. Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BVA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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.
⋮
```

Vytvoření dynamické fronty

Tento příklad demonstruje, jak použít volání MQOPEN k vytvoření dynamické fronty.

Tato extrakce je převzata z ukázkové aplikace Credit Check (program CSQ4CVB1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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
*
```

Otevření existující fronty

Tento příklad demonstruje, jak použít volání MQOPEN k otevření existující fronty.

Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BVA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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
```

Zavření fronty

Tento příklad ukazuje, jak použít volání MQCLOSE.

Proměnné použité v tomto extraktu kódu jsou ty, které byly nastaveny v "Připojení ke správci front" na stránce 23. Tato extrakce je převzata z ukázkové aplikace Procházet (program CSQ4BVA1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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.
*
```

Vložení zprávy pomocí příkazu MQPUT

Tento příklad demonstruje použití volání MQPUT s použitím kontextu.

Tato extrakce je převzata z ukázkové aplikace Credit Check (program CSQ4CVB1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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.
```

Vložení zprávy pomocí MQPUT1

Tento příklad ukazuje, jak použít volání MQPUT1 .

Tato extrakce je převzata z ukázkové aplikace Credit Check (program CSQ4CVB5) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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.
*
```

získávání zprávy

Tento příklad demonstruje, jak použít volání MQGET k odebrání zprávy z fronty.

Tato extrakce je převzata z ukázkové aplikace Credit Check (program CSQ4CVB1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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.
```

Získání zprávy pomocí volby čekání

Tento příklad demonstruje, jak použít volání MQGET s volbou wait a jak přijmout oříznuté zprávy.

Tato extrakce je převzata z ukázkové aplikace Credit Check (program CSQ4CVB5) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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.
⋮
```

Získání zprávy pomocí signalizace

Tento příklad demonstruje, jak používat volání MQGET se signalizací. Tato extrakce je převzata z ukázkové aplikace Credit Check (program CSQ4CVB2) dodané s produktem IBM MQ for z/OS.

Signalizace je k dispozici pouze s produktem IBM 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
*
     ⋮
```

Zjišťování atributů objektu

Tento příklad demonstruje, jak použít volání MQINQ k dotazování na atributy fronty.

Tato extrakce je převzata z ukázkové aplikace Atributy fronty (program CSQ4CVC1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách viz Ukázkové procedurální programy (platformy s výjimkou 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.
        ⋮
```

Nastavení atributů fronty

Tento příklad demonstruje, jak použít volání MQSET ke změně atributů fronty.

Tato extrakce je převzata z ukázkové aplikace Atributy fronty (program CSQ4CVC1) dodané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách naleznete v tématu Ukázkové procedurální programy (platformy s výjimkou 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.
```

System/390 assembler-language examples

Tato kolekce témat je většinou převzata z ukázkových aplikací IBM MQ for z/OS .

Připojení ke správci front

Tento příklad demonstruje, jak použít volání MQCONN pro připojení programu ke správci front v dávce z/OS .

Tato extrakce je převzata z ukázkového programu Procházet (CSQ4BAA1) dodaného s produktem IBM 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
*
```

Odpojení od správce front

Tento příklad demonstruje, jak použít volání MQDISC k odpojení programu od správce front v dávce z/OS .

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

Vytvoření dynamické fronty

Tento příklad demonstruje, jak použít volání MQOPEN k vytvoření dynamické fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

Otevření existující fronty

Tento příklad demonstruje, jak použít volání MQOPEN k otevření fronty, která již byla definována.

Ukazuje, jak zadat dvě volby. Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

Zavření fronty

Tento příklad demonstruje, jak použít volání MQCLOSE k zavření fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

Vložení zprávy pomocí příkazu MQPUT

Tento příklad demonstruje, jak použít volání MQPUT k vložení zprávy do fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

Vložení zprávy pomocí MQPUT1

Tento příklad ukazuje, jak použít volání MQPUT1 k otevření fronty, vložení jedné zprávy do fronty a zavření fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

získávání zprávy

Tento příklad demonstruje, jak použít volání MQGET k odebrání zprávy z fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

Získání zprávy pomocí volby čekání

Tento příklad demonstruje, jak použít volbu wait volání MQGET.

Tento kód přijímá oříznuté zprávy. Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

Získání zprávy pomocí signalizace

Tento příklad ukazuje, jak použít volání MQGET k nastavení signálu, abyste byli upozorněni, když do fronty dorazí vhodná zpráva.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM 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
```

Zjišťování a nastavení atributů fronty

Tento příklad demonstruje použití volání MQINQ k dotazování na atributy fronty a použití volání MQSET ke změně atributů fronty.

Tato extrakce je převzata z ukázkové aplikace Atributy fronty (program CSQ4CAC1) dodané s produktem IBM 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 ,(0,0,0,0,0,0,0,0,0,0,0),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?
         ⋮
```

Příklady PL/I

Použití PL/I je podporováno pouze produktem z/OS . Tato kolekce témat demonstruje techniky využívající příklady PL/I.

Připojení ke správci front

Tento příklad demonstruje, jak použít volání MQCONN pro připojení programu ke správci front v dávce z/OS .

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****************************************************/
/* STRUCTURE BASED ON PARAMETER INPUT AREA (PARAM)  */
/*****************************************************/
DCL 1 INPUT_PARAM      BASED(ADDR(PARAM)),
      2 PARAM_LENGTH   FIXED BIN(15),
      2 PARAM_MQMNAME  CHAR(48);
      ⋮
/*****************************************************/
/* WORKING STORAGE DECLARATIONS                      */
/*****************************************************/
DCL MQMNAME                  CHAR(48);
DCL COMPCODE                 BINARY FIXED (31);
DCL REASON                   BINARY FIXED (31);
DCL HCONN                    BINARY FIXED (31);
⋮
/*****************************************************/
/* COPY QUEUE MANAGER NAME PARAMETER                 */
/* TO LOCAL STORAGE                                  */
/*****************************************************/
MQMNAME = ' ';
MQMNAME = SUBSTR(PARAM_MQMNAME,1,PARAM_LENGTH);
⋮
/*****************************************************/
/* CONNECT FROM THE QUEUE MANAGER                    */
/*****************************************************/
CALL MQCONN (MQMNAME,     /* MQM SYSTEM NAME        */
             HCONN,       /* CONNECTION HANDLE      */
             COMPCODE,    /* COMPLETION CODE        */
             REASON);     /* REASON CODE            */

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

Odpojení od správce front

Tento příklad demonstruje, jak použít volání MQDISC k odpojení programu od správce front v dávce z/OS .

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****************************************************/
/* WORKING STORAGE DECLARATIONS                      */
/*****************************************************/
DCL COMPCODE                 BINARY FIXED (31);
DCL REASON                   BINARY FIXED (31);
DCL HCONN                    BINARY FIXED (31);
⋮
/*****************************************************/
/* DISCONNECT FROM THE QUEUE MANAGER                 */
/*****************************************************/
```

```
     CALL MQDISC (HCONN,       /* CONNECTION HANDLE      */
                  COMPCODE,    /* COMPLETION CODE        */
                  REASON);     /* REASON CODE            */

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

Vytvoření dynamické fronty

Tento příklad demonstruje, jak použít volání MQOPEN k vytvoření dynamické fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

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

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

Otevření existující fronty

Tento příklad demonstruje, jak použít volání MQOPEN k otevření existující fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

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

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

Zavření fronty

Tento příklad ukazuje, jak použít volání MQCLOSE.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

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

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

Vložení zprávy pomocí příkazu MQPUT

Tento příklad demonstruje použití volání MQPUT s použitím kontextu.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

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

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

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

Vložení zprávy pomocí MQPUT1

Tento příklad ukazuje, jak použít volání MQPUT1 .

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

/*************************************************************/
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED.        */
/*************************************************************/
LMQMD.MSGTYPE = MQMT_REQUEST;
LMQMD.PRIORITY = 5;
LMQMD.PERSISTENCE = MQPER_PERSISTENT;
LMQMD.REPLYTOQ = REPLY_TO_QUEUE;
LMQMD.REPLYTOQMGR = ' ';
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;
```

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

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

CALL MQPUT1 (HCONN,
LMQOD,
LMQMD,
LMQPMO,
```

```
BUFFLEN,
BUFFER,
COMPCODE,
REASON);

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

získávání zprávy

Tento příklad demonstruje, jak použít volání MQGET k odebrání zprávy z fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

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

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

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

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

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

Získání zprávy pomocí volby čekání

Tento příklad demonstruje, jak použít volání MQGET s volbou wait a jak přijmout oříznuté zprávy.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

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

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

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

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

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

     SELECT(COMPCODE);
       WHEN (MQCC_OK) DO;   /* GET WAS SUCCESSFUL */
       ⋮
       END;
       WHEN (MQCC_WARNING) DO;
         IF REASON = MQRC_TRUNCATED_MSG_ACCEPTED
           THEN DO;         /* GET WAS SUCCESSFUL */
           ⋮
           END;
           ELSE DO;
             ⋮
             CALL ERROR_ROUTINE;
           END;
       END;
       WHEN (MQCC_FAILED) DO;
           ⋮
           CALL ERROR_ROUTINE;
         END;
       END;
       OTHERWISE;
     END;
```

Získání zprávy pomocí signalizace

Extrakce kódu, která demonstruje, jak používat volání MQGET se signalizací.

Signalizace je k dispozici pouze s produktem IBM MQ for z/OS .

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

```
/* WAIT INTERVAL SET TO ONE MINUTE.                       */
/*********************************************************/
     LMQGMO.OPTIONS = MQGMO_SET_SIGNAL +
                      MQGMO_NO_SYNCPOINT;
     LMQGMO.WAITINTERVAL=60000;
     LMQGMO.SIGNAL1 = ADDR(ECB_FIXED);
```

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

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

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

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

          END;
```

```
      END;
    :
END DO_WORK;

GET_MSG: PROC;
```

```
/*******************************************************/
/*                                                     */
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST.           */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.            */
/* MD AND GMO SET UP AS REQUIRED.                      */
/*                                                     */
/*******************************************************/

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

END GET_MSG;

NO_MSG: PROC;
:
END NO_MSG;
```

Zjišťování atributů objektu

Tento příklad demonstruje, jak použít volání MQINQ k dotazování na atributy fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

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

     SELECTORS(01) = MQIA_CURRENT_Q_DEPTH;

     SELECTORCOUNT  = 1;
     INTATTRCOUNT   = 1;

     CHARATTRLENGTH = 0;
/*******************************************************/
/*                                                     */
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST.           */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.            */
/*                                                     */
/*******************************************************/
     CALL MQINQ (HCONN,
                 HOBJ,
```

```
                 SELECTORCOUNT,
                 SELECTORS,
                 INTATTRCOUNT,
                 INTATTRS,
                 CHARATTRLENGTH,
                 CHARATTRS,
                 COMPCODE,
                 REASON);


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

Nastavení atributů fronty

Tento příklad demonstruje, jak použít volání MQSET ke změně atributů fronty.

Tato extrakce není převzata z ukázkových aplikací dodávaných s produktem IBM MQ.

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

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

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

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

     SELECTORCOUNT  = 2;
     INTATTRCOUNT   = 2;

     CHARATTRLENGTH = 0;
```

```
/*************************************************************/
/*                                                           */
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST.                 */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.                  */
/*                                                           */
/*************************************************************/
     CALL MQSET (HCONN,
                 HOBJ,
                 SELECTORCOUNT,
                 SELECTORS,
                 INTATTRCOUNT,
                 INTATTRS,
```

```
                  CHARATTRLENGTH,
                  CHARATTRS,
                  COMPCODE,
                  REASON);

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

Konstanty

Použijte referenční informace v této sekci k provedení úloh, které odpovídají vašim obchodním potřebám.

IBM MQ COPY, záhlaví, zahrnutí a soubory modulu

Tyto informace jsou obecné informace o programovacím rozhraní.

Tento oddíl obsahuje informace, které vám pomohou používat rozhraní MQI pro různé programovací jazyky, jak je uvedeno níže.

Soubory záhlaví C

K dispozici jsou soubory záhlaví, které vám pomohou při psaní aplikačních programů v jazyce C, které používají rozhraní MQI.

Soubory záhlaví C jsou shrnuty v následující tabulce:

Tabulka 1. Hlavičkové soubory C-volání prototypů, datových typů, návratových kódů, konstant a struktur

Název souboru	Popis	IBM i	Systémy AIX and Linux®	Windows	z/OS
Prototypy volání, datové typy, návratové kódy, konstanty a struktury					
CMQC	Definice rozhraní MQI	C	C	C	C
CMQBC	Definice MQAI	C	C	C	
CMQEC	Definice vstupních bodů rozhraní (zahrnuje CMQC, CMQXC a CMQZC)		C	C	
CMQCFC	Definice PCF	C	C	C	C
CMQPSC	Definice publikování/odběru	C	C	C	C
CMQXC	Definice kanálů a ukončení	C	C	C	C
CMQZC	Definice instalovatelných služeb	C	C	C	
Klíč: C= poskytnuté soubory					

Soubory COPY v jazyce COBOL

K dispozici jsou různé soubory COPY, které vám pomohou při psaní aplikačních programů v jazyce COBOL, které používají rozhraní MQI.

Tabulka 2. Kopírované soubory COBOL-návratové kódy, konstanty a struktury					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
Návratové kódy a konstanty					
CMQx	Definice rozhraní MQI	V	V	V	V
CMQCFx	Definice PCF	V	V	V	V
CMQPSx	Definice publikování/odběru	V	V	V	V
CMQXx	Definice kanálů a ukončení	V	V	V	V
Struktury					
CMQAIRx	MQAIR-záznam ověřovacích informací		V L	V L	
CMQBOx	MQBO-volby zahájení	V L	V L	V L	
CMQCDx	MQCD-Definice kanálu	V L	V L	V L	V L
CMQCFBFx	Parametr filtru bajtového řetězce MQCFBF-PCF	V L	V L	V L	V L
CMQCFBSx	Parametr bajtového řetězce MQCFBS-PCF	V L	V L	V L	V L
CMQCFGRx	parametr skupiny MQCFGR-PCF	V L	V L	V L	V L
CMQCFHx	Záhlaví MQCFH-PCF	V L	V L	V L	V L
CMQCFIFx	Parametr celočíselného filtru MQCFIF-PCF	V L	V L	V L	V L
CMQCFILx	MQCFIL-parametr seznamu celých čísel PCF	V L	V L	V L	V L
CMQCFINx	Celočíselný parametr MQCFIN-PCF	V L	V L	V L	V L
CMQCFSFx	MQCFSF-parametr filtru řetězců PCF	V L	V L	V L	V L
CMQCFSLx	MQCFSL-Parametr seznamu řetězců PCF	V L	V L	V L	V L
CMQCFSTx	Parametr řetězce MQCFST-PCF	V L	V L	V L	V L
CMQCFXLx	MQCFIL64 -Parametr 64bitového seznamu celých čísel PCF	V L	V L	V L	V L
CMQCFXNx	MQCFIN64 -64bitový celočíselný parametr PCF	V L	V L	V L	V L
CMQCHRVx	MQCHARV-Řetězec s proměnnou délkou	V L	V L	V L	V L
CMQCIHx	Záhlaví MQCIH- CICS bridge	V L	V L	V L	V L
CMQCNOx	MQCNO-Volby připojení	V L	V L	V L	V L
CMQCSPx	MQCSP-parametry zabezpečení	V L	V L	V L	V L
CMQCXPx	MQCXP-Parametry uživatelské procedury kanálu	V L			V L
CMQDHx	Záhlaví MQDH-Distribution	V L	V L	V L	V L

Tabulka 2. Kopírované soubory COBOL-návratové kódy, konstanty a struktury (pokračování)

Název souboru	Popis	IBM i	AIX and Linux	Windows	z/OS
CMQDLHx	MQDLH-Záhlaví nedoručených zpráv	V L	V L	V L	V L
CMQDXPx	MQDXP-Parametry uživatelské procedury převodu dat	V L		V L	
CMQEPHx	MQEPH-Vložené záhlaví PCF	V L	V L	V L	V L
CMQGMOx	MQGMO-Volby získání zprávy	V L	V L	V L	V L
CMQIIHx	Záhlaví informací MQIIH- IMS	V L	V L	V L	V L
CMQMDx	MQMD-Deskriptor zpráv	V L	V L	V L	V L
CMQMD1x	MQMD1 -Popisovač zprávy verze 1.	V L	V L	V L	V L
CMQMD2x	MQMD2 -Popisovač zprávy verze 2.	V L	V L	V L	V L
CMQMDEx	MQMDE-Rozšířený deskriptor zprávy	V L	V L	V L	V L
CMQODx	MQOD-Popisovač objektu	V L	V L	V L	V L
CMQORx	MQOR-záznam objektu	V L	V L	V L	V L
CMQPMOx	MQPMO-Volby vložení zprávy	V L	V L	V L	V L
CMQRFHx	MQRFH-Pravidla a formátování záhlaví	V L	V L	V L	V L
CMQRFH2x	MQRFH2 -Pravidla a formátování záhlaví 2	V L	V L	V L	V L
CMQRMHx	MQRMH-Záhlaví referenční zprávy	V L	V L	V L	V L
CMQRRx	MQRR-záznam odezvy	V L	V L	V L	
CMQSCOx	Volby konfigurace MQSCO-TLS		V L	V L	
CMQTMx	MQTM-zpráva spouštěče	V L		V L	V L
CMQTMCx	MQTMC-Znak zprávy spouštěče	V L	V L		
CMQTMC2x	MQTMC2 -zpráva spouštěče 2 znaky	V L	V L	V L	V L
CMQWIHx	MQWIH-záhlaví informací o práci	V L	V L	V L	V L
CMQXQHx	MQXQH-Záhlaví přenosové fronty	V L	V L	V L	V L

Klíč:

- Soubory s poskytnutými počátečními hodnotami, x = V
- Soubory bez poskytnutých počátečních hodnot, x = L

z/OS *Soubory začlenění PL/I*

Pro programovací jazyk PL/I je k dispozici řada souborů INCLUDE. Tyto soubory jsou k dispozici pouze v systému z/OS .

Tabulka 3. PL/I zahrnuje soubory-datové typy, návratové kódy, konstanty a struktury					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
Datové typy, návratové kódy, konstanty a struktury					
CMQP	Definice rozhraní MQI				P
CMQCFP	Definice PCF				P
CMQEPP	Definice vstupních bodů				P
CMQPSP	Definice publikování/odběru				P
CMQXP	Definice kanálů a ukončení				P
Klíč: P= poskytnutý soubor					

IBM i *Kopírované soubory RPG*

Soubory RPG COPY jsou poskytovány pro programovací jazyk RPG. Tyto soubory jsou k dispozici pouze na webu IBM i.

Tabulka 4. Kopírované soubory RPG-návratové kódy, konstanty a struktury					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
Návratové kódy a konstanty					
CMQx	Definice rozhraní MQI	G R			
CMQCFx	Definice PCF	G			
CMQPSx	Definice publikování/odběru	G			
CMQXx	Definice kanálů a ukončení	G R			
Struktury					
CMQBOx	MQBO-volby zahájení	G H			
CMQCDx	MQCD-Definice kanálu	G H R			
CMQCFBFx	Parametr filtru bajtového řetězce MQCFBF-PCF	G H			
CMQCFBSx	Parametr bajtového řetězce MQCFBS-PCF	G H			
CMQCFGRx	parametr skupiny MQCFGR-PCF	G H			
CMQCFHx	Záhlaví MQCFH-PCF	G H			
CMQCFIFx	Parametr celočíselného filtru MQCFIF-PCF	G H			
CMQCFILx	MQCFIL-parametr seznamu celých čísel PCF	G H			
CMQCFINx	Celočíselný parametr MQCFIN-PCF	G H			
CMQCFSFx	MQCFSF-parametr filtru řetězců PCF	G H			
CMQCFSLx	MQCFSL-Parametr seznamu řetězců PCF	G H			

Tabulka 4. Kopírované soubory RPG-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQCFSTx	Parametr řetězce MQCFST-PCF	G H			
CMQCFXLx	MQCFIL64 -Parametr 64bitového seznamu celých čísel PCF	G H			
CMQCFXNx	MQCFIN64 -64bitový celočíselný parametr PCF	G H			
CMQCHARVx	MQCHARV-Řetězec s proměnnou délkou	G H			
CMQCIHx	Záhlaví MQCIH- CICS bridge	G H			
CMQCNOx	MQCNO-Volby připojení	G H			
CMQCSPx	MQCSP-parametry zabezpečení	G H			
CMQCXPx	MQCXP-Parametry uživatelské procedury kanálu	G H R			
CMQDHx	Záhlaví MQDH-Distribution	G H R			
CMQDLHx	MQDLH-Záhlaví nedoručených zpráv	G H R			
CMQDXPx	MQDXP-Parametry uživatelské procedury převodu dat	G H R			
CMQEPHx	MQEPH-Vložené záhlaví PCF	G H			
CMQGMOx	MQGMO-Volby získání zprávy	G H R			
CMQIIHx	Záhlaví informací MQIIH- IMS	G H R			
CMQMDx	MQMD-Deskriptor zpráv	G H R			
CMQMD1x	MQMD1 -Popisovač zprávy verze 1.	G H R			
CMQMD2x	MQMD2 -Popisovač zprávy verze 2.	G H			
CMQMDEx	MQMDE-Rozšířený deskriptor zprávy	G H R			
CMQODx	MQOD-Popisovač objektu	G H R			
CMQORx	MQOR-záznam objektu	G H R			
CMQPMOx	MQPMO-Volby vložení zprávy	G H R			
CMQPXPx	MQPXP-Parametry ukončení směrování publikování/odběru	G H			
CMQRFHx	MQRFH-Pravidla a formátování záhlaví	G H			
CMQRFH2x	MQRFH2 -Pravidla a formátování záhlaví 2	G H			
CMQRMHx	MQRMH-Záhlaví referenční zprávy	G H R			
CMQRRx	MQRR-záznam odezvy	G H R			
CMQTMx	MQTM-zpráva spouštěče	G H R			
CMQTMCx	MQTMC-Znak zprávy spouštěče	G H R			

Tabulka 4. Kopírované soubory RPG-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQTMC2x	MQTMC2 -zpráva spouštěče 2 znaky	G H R			
CMQWIHx	MQWIH-záhlaví informací o práci	G H			
CMQXQHx	MQXQH-Záhlaví přenosové fronty	G H R			
Klíč: • Soubor pro statické sestavení, inicializováno, poskytováno x = G • Soubor pro statické sestavení, neinicializováno, poskytováno x = H • Soubor pro dynamické sestavení, inicializováno, poskytnuto, x = R					

Windows *Soubory modulu Visual Basic*

Soubory záhlaví (nebo formuláře) jsou poskytovány jako pomůcka při psaní aplikačních programů jazyka Visual Basic, které používají rozhraní MQI. Tyto hlavičkové soubory jsou dodávány pouze ve 32bitových verzích.

Tabulka 5. Soubory modulu Visual Basic-deklarace volání, datové typy, návratové kódy, konstanty a struktury					
Název souboru	**Popis**	**IBM i**	**Systémy AIX and Linux**	**Windows**	**z/OS**
Deklarace volání, datové typy, návratové kódy, konstanty a struktury					
CMQB	Definice rozhraní MQI			B	
CMQBB	Definice MQAI			B	
CMQCFB	Definice PCF			B	
CMQXB	Definice kanálů a ukončení			B	
Klíč: B= Soubor poskytnut					

z/OS *z/OS Assembler COPY soubory (kopie souborů)*

K dispozici jsou různé soubory COPY, které vám pomohou při psaní aplikačních programů z/OS Assembler, které používají rozhraní MQI.

Tabulka 6. z/OS Assembler copy files-datové typy, návratové kódy, konstanty a struktury					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
Datové typy, návratové kódy a konstanty					
CMQA	Definice rozhraní MQI				A
CMQCFA	Definice PCF				A
CMQPSA	Definice publikování/odběru				A
CMQVERA	Řízení verze struktury				A
CMQXA	Definice kanálů a ukončení				A
Struktury					
CMQCDA	MQCD-Definice kanálu				

Tabulka 6. z/OS Assembler copy files-datové typy, návratové kódy, konstanty a struktury (pokračování)

Název souboru	Popis	IBM i	AIX and Linux	Windows	z/OS
CMQCFBFA	Parametr filtru bajtového řetězce MQCFBF-PCF				
CMQCFBSA	Parametr bajtového řetězce MQCFBS-PCF				A
CMQCFGRA	parametr skupiny MQCFGR-PCF				A
CMQCFHA	Záhlaví MQCFH-PCF				A
CMQCFIFA	Parametr celočíselného filtru MQCFIF-PCF				A
CMQCFILA	MQCFIL-parametr seznamu celých čísel PCF				A
CMQCFINA	Celočíselný parametr MQCFIN-PCF				A
CMQCFSFA	MQCFSF-parametr filtru řetězců PCF				A
CMQCFSLA	MQCFSL-Parametr seznamu řetězců PCF				A
CMQCFSTA	Parametr řetězce MQCFST-PCF				A
CMQCFXLA	MQCFIL64 -Parametr 64bitového seznamu celých čísel PCF				A
CMQCFXNA	MQCFIN64 -64bitový celočíselný parametr PCF				A
CMQCHARVA	MQCHARV-Řetězec s proměnnou délkou				A
CMQCIHA	Záhlaví MQCIH- CICS bridge				A
CMQCNOA	MQCNO-Volby připojení				A
CMQCSPA	MQCSP-parametry zabezpečení				A
CMQCXPA	MQCXP-Parametry uživatelské procedury kanálu				A
CMQDHA	Záhlaví MQDH-Distribution				A
CMQDLHA	MQDLH-Záhlaví nedoručených zpráv				A
CMQDXPA	MQDXP-Parametry uživatelské procedury převodu dat				A
CMQEPHA	MQEPH-Vložené záhlaví PCF				A
CMQGMOA	MQGMO-Volby získání zprávy				A
CMQIIHA	Záhlaví informací MQIIH- IMS				A
CMQMDA	MQMD-Deskriptor zpráv				A
CMQMD1A	MQMD1 -Popisovač zprávy verze 1.				A
CMQMD2A	MQMD2 -Popisovač zprávy verze 2.				A

Tabulka 6. z/OS Assembler copy files-datové typy, návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQMDEA	MQMDE-Rozšířený deskriptor zprávy				A
CMQODA	MQOD-Popisovač objektu				A
CMQORA	MQOR-záznam objektu				A
CMQPMOA	MQPMO-Volby vložení zprávy				A
CMQRFHA	MQRFH-Pravidla a formátování záhlaví				A
CMQRFH2A	MQRFH2 -Pravidla a formátování záhlaví 2				A
CMQRMHA	MQRMH-Záhlaví referenční zprávy				A
CMQTMA	MQTM-zpráva spouštěče				A
CMQTMC2A	MQTMC2 -zpráva spouštěče 2 znaky				A
CMQWCRA	MQWCR-Záznam klastru pracovní zátěže klastru				A
CMQWDRA	MQWDR-Záznam místa určení pracovní zátěže klastru				A
CMQWDR1A	MQWDR1 -Záznam cíle pracovní zátěže klastru verze 1				A
CMQWDR2A	MQWDR2 -Záznam cíle pracovní zátěže klastru verze 2				A
CMQWIHA	MQWIH-záhlaví informací o práci				A
CMQWQRA	MQWQR-Záznam fronty pracovní zátěže klastru				A
CMQWQR1A	MQWQR1 -Záznam fronty pracovní zátěže klastru verze 1.				A
CMQWQR2A	MQWQR2 -Záznam fronty pracovní zátěže klastru verze 2				A
CMQWXPA	MQWXP-Parametry ukončení pracovní zátěže klastru				A
CMQWXP1A	MQWXP1 -Parametry uživatelské procedury pracovní zátěže klastru verze 1				A
CMQWXP2A	MQWXP2 -Parametry ukončení pracovní zátěže klastru verze 2				A
CMQWXP3A	MQWXP3 -Parametry ukončení pracovní zátěže klastru verze 3				A
CMQXPA	Parametry uživatelské procedury rozhraní API MQXP- CICS				A
CMQXQHA	MQXQH-Záhlaví přenosové fronty				A

Tabulka 6. z/OS Assembler copy files-datové typy, návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQXWDA	MQXWD-deskriptor čekání na ukončení				A
Klíč: A= poskytnutý soubor					

MQ_ * (délky řetězců)

Tabulka 7. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQ_ABEND_CODE_LENGTH	4	X'00000004'
MQ_ACCOUNTING_TOKEN_LENGTH	32	X'00000020'
MQ_APPL_FUNCTION_NAME_LENGTH	10	X'0000000A'
MQ_APPL_IDENTITY_DATA_LENGTH	32	X'00000020'
MQ_NÁZEV_APLIKACE_DÉLKA	28	X'0000001C'
MQ_APPL_ORIGIN_DATA_LENGTH	4	X'00000004'
MQ_APPL_TAG_LENGTH	28	X'0000001C'
MQ_ARM_SUFFIX_LENGTH	2	X'00000002'
MQ_ATTENTION_ID_LENGTH	4	X'00000004'
MQ_AUTH_INFO_CONN_NAME_LENGTH	264	X'00000108'
MQ_AUTH_INFO_DESC_LENGTH	64	X'00000040'
MQ_AUTH_INFO_NAME_LENGTH	48	X'00000030'
MQ_AUTH_INFO_OCSP_URL_LENGTH	256	X'00000100'
MQ_AUTHENTICATOR_LENGTH	8	X'00000008'
MQ_AUTO_REORG_CATALOG_LENGTH	44	X'0000002C'
MQ_AUTO_REORG_TIME_LENGTH	4	X'00000004'
MQ_BATCH_INTERFACE_ID_LENGTH	8	X'00000008'
MQ_BRIDGE_NAME_LENGTH	24	X'00000018'
MQ_CANCEL_CODE_LENGTH	4	X'00000004'
MQ_CF_STRUC_DESC_LENGTH	64	X'00000040'
MQ_CF_STRUC_NAME_LENGTH	12	X'0000000C'
MQ_CHANNEL_DATE_LENGTH	12	X'0000000C'
MQ_CHANNEL_DESC_LENGTH	64	X'00000040'
MQ_CHANNEL_NAME_LENGTH	20	X'00000014'
MQ_CHANNEL_TIME_LENGTH	8	X'00000008'
MQ_CHINIT_SERVICE_PARM_LENGTH	32	X'00000020'
MQ_CICS_NÁZEV_SOUBORU_LENGTH	8	X'00000008'
MQ_CLIENT_ID_LENGTH	23	X'00000017'
MQ_CLUSTER_NAME_LENGTH	48	X'00000030'
MQ_CONN_NAME_LENGTH	264	X'00000108'
MQ_CONN_TAG_LENGTH	128	X'00000080'

Tabulka 7. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQ_CONNECTION_ID_LENGTH	24	X'00000018'
MQ_CORREL_ID_LENGTH	24	X'00000018'
MQ_CREATION_DATE_LENGTH	12	X'0000000C'
MQ_CREATION_TIME_LENGTH	8	X'00000008'
MQ_DATE_LENGTH	12	X'0000000C'
MQ_ROZLIŠENÉ_DÉLKA_NÁZVU	1024	X'00000400'
MQ_DNS_GROUP_NAME_LENGTH	18	X'00000012'
MQ_EXIT_DATA_LENGTH	32	X'00000020'
MQ_EXIT_INFO_NAME_LENGTH	48	X'00000030'
MQ_EXIT_NAME_LENGTH	(value differs by platform or version)	
MQ_EXIT_PD_AREA_LENGTH	48	X'00000030'
MQ_EXIT_USER_AREA_LENGTH	16	X'00000010'
DÉLKA_ZAŘÍZENÍ_MQ_LENGTH	8	X'00000008'
MQ_FACILITY_LIKE_LENGTH	4	X'00000004'
MQ_FORMAT_LENGTH	8	X'00000008'
DÉLKA funkce MQ_FUNCTION_LENGTH	4	X'00000004'
MQ_GROUP_ID_LENGTH	24	X'00000018'
MQ_LDAP_PASSWORD_LENGTH	32	X'00000020'
MQ_LISTENER_NAME_LENGTH	48	X'00000030'
MQ_LISTENER_DESC_LENGTH	64	X'00000040'
MQ_LOCAL_ADDRESS_LENGTH	48	X'00000030'
MQ_LTERM_OVERRIDE_LENGTH	8	X'00000008'
MQ_LU_NAME_LENGTH	8	X'00000008'
MQ_LUWID_LENGTH	16	X'00000010'
MQ_MAX_EXIT_NAME_LENGTH	128	X'00000080'
MQ_MAX_MCA_USER_ID_LENGTH	64	X'00000040'
MQ_MAX_NÁZEV_VLASTNOSTI_DÉLKA	4095	X'00000FFF'
MQ_MAX_ID_UŽIVATELE_DÉLKA	64	X'00000040'
MQ_MCA_JOB_NAME_LENGTH	28	X'0000001C'
MQ_MCA_NAME_LENGTH	20	X'00000014'
MQ_MCA_USER_DATA_LENGTH	32	X'00000020'
DÉLKA_ID_UŽIVATELE_MQ_MCA_ID_LENGTH	(value differs by platform or version)	(value differs by platform or version)
MQ_MFS_MAP_NAME_LENGTH	8	X'00000008'
MQ_MODE_NAME_LENGTH	8	X'00000008'
MQ_MSG_HEADER_LENGTH	4000	X'00000FA0'
MQ_MSG_ID_LENGTH	24	X'00000018'
MQ_MSG_TOKEN_LENGTH	16	X'00000010'
MQ_NAMELIST_DESC_LENGTH	64	X'00000040'
MQ_NAMELIST_NAME_LENGTH	48	X'00000030'

Tabulka 7. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQ_NHA_INSTANCE_NAME_LENGTH	48	X'00000030'
MQ_OBJECT_INSTANCE_ID_LENGTH	24	X'00000018'
MQ_OBJECT_NAME_LENGTH	48	X'00000030'
MQ_PASS_TICKET_APPL_LENGTH	8	X'00000008'
MQ_PASSWORD_LENGTH	12	X'0000000C'
MQ_PROCESS_APPL_ID_LENGTH	256	X'00000100'
MQ_PROCESS_DESC_LENGTH	64	X'00000040'
MQ_PROCESS_ENV_DATA_LENGTH	128	X'00000080'
MQ_PROCESS_NAME_LENGTH	48	X'00000030'
MQ_PROCESS_USER_DATA_LENGTH	128	X'00000080'
NÁZEV_PROGRAMU_MQ_LENGTH	20	X'00000014'
MQ_PUT_NÁZEV_APLIKACE_LENGTH	28	X'0000001C'
MQ_PUT_DATE_LENGTH	8	X'00000008'
MQ_PUT_TIME_LENGTH	8	X'00000008'
MQ_Q_DESC_LENGTH	64	X'00000040'
MQ_Q_MGR_DESC_LENGTH	64	X'00000040'
MQ_Q_MGR_IDENTIFIER_LENGTH	48	X'00000030'
MQ_Q_NÁZEV_MGR_LENGTH	48	X'00000030'
MQ_Q_DÉLKA_NÁZVU	48	X'00000030'
MQ_QSG_NAME_LENGTH	4	X'00000004'
MQ_REMOTE_SYS_ID_LENGTH	4	X'00000004'
MQ_SECURITY_ID_LENGTH	40	X'00000028'
MQ_SELECTOR_LENGTH	10240	X'00002800'
MQ_SERVICE_ARGS_LENGTH	255	X'000000FF'
Délka příkazu MQ_SERVICE_COMMAND_LENGTH	255	X'000000FF'
MQ_SERVICE_DESC_LENGTH	64	X'00000040'
MQ_SERVICE_NAME_LENGTH	32	X'00000020'
MQ_SERVICE_PATH_LENGTH	255	X'000000FF'
MQ_SERVICE_STEP_LENGTH	8	X'00000008'
MQ_SHORT_CONN_NAME_LENGTH	20	X'00000014'
MQ_SHORT_DNAME_LENGTH	256	X'00000100'
MQ_SSL_CIPHER_SPEC_LENGTH	32	X'00000020'
MQ_SSL_CRYPTO_HARDWARE_LENGTH	256	X'00000100'
MQ_SSL_HANDSHAKE_STAGE_LENGTH	32	X'00000020'
MQ_SSL_KEY_LIBRARY_LENGTH	44	X'0000002C'
MQ_SSL_KEY_MEMBER_LENGTH	8	X'00000008'
MQ_SSL_KEY_REPOSITORY_LENGTH	256	X'00000100'
MQ_SSL_PEER_NAME_LENGTH	1024	X'00000400'
MQ_SSL_SHORT_PEER_NAME_LENGTH	256	X'00000100'

Tabulka 7. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQ_START_CODE_LENGTH	4	X'00000004'
MQ_STORAGE_CLASS_DESC_LENGTH	64	X'00000040'
MQ_STORAGE_CLASS_LENGTH	8	X'00000008'
MQ_SUB_IDENTITY_LENGTH	128	X'00000080'
MQ_SUB_POINT_LENGTH	128	X'00000080'
MQ_SUITE_B_128_BIT	2	X'00000002'
MQ_SUITE_B_192_BIT	4	X'00000004'
MQ_SUITE_B_NONE	1	X'00000001'
MQ_SUITE_B_NOT_AVAILABLE	0	X'00000000'
MQ_TCP_NAME_LENGTH	8	X'00000008'
MQ_TIME_LENGTH	8	X'00000008'
MQ_TOPIC_DESC_LENGTH	64	X'00000040'
MQ_TOPIC_NAME_LENGTH	48	X'00000030'
MQ_TOPIC_STR_LENGTH	10240	X'00002800'
MQ_TOTAL_EXIT_DATA_LENGTH	999	X'000003E7'
MQ_TOTAL_EXIT_NAME_LENGTH	999	X'000003E7'
MQ_TP_NAME_LENGTH	64	X'00000040'
MQ_TPIPE_NAME_LENGTH	8	X'00000008'
MQ_TRAN_INSTANCE_ID_LENGTH	16	X'00000010'
MQ_TRANSACTION_ID_LENGTH	4	X'00000004'
MQ_TRIGGER_DATA_LENGTH	64	X'00000040'
NÁZEV_PROGRAMU_XX_ENCODE_CASE_ONE mq_trigger_trigger_length	8	X'00000008'
MQ_TRIGGER_TERM_ID_LENGTH	4	X'00000004'
MQ_TRIGGER_TRANS_ID_LENGTH	4	X'00000004'
DÉLKA_ID_UŽIVATELE_MQ_LENGTH	12	X'0000000C'
DÉLKA_VERZE_MQ_VERZE	8	X'00000008'
MQ_XCF_NÁZEV_SKUPINY_DÉLKA	8	X'00000008'
MQ_XCF_NÁZEV_ČLENU_MQ_XCF_LENGTH	16	X'00000010'

*MQ_ * (Formát příkazu Délky řetězce)*

Tabulka 8. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQ_ARCHIVE_PFX_LENGTH	36	X'00000024'
MQ_ARCHIVE_UNIT_LENGTH	8	X'00000008'
MQ_ASID_LENGTH	4	X'00000004'
MQ_AUTH_PROFILE_NAME_LENGTH	48	X'00000030'
MQ_CF_LEID_LENGTH	12	X'0000000C'
MQ_COMMAND_MQSC_LENGTH	32768	X'00008000'
MQ_DATA_SET_NAME_LENGTH	44	X'0000002C'

Tabulka 8. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQ_DB2_NAME_LENGTH	4	X'00000004'
MQ_DSG_NAME_LENGTH	8	X'00000008'
MQ_ENTITY_NAME_LENGTH	1024	X'00000400'
MQ_ENV_INFO_LENGTH	96	X'00000060'
MQ_IP_ADDRESS_LENGTH	48	X'00000030'
MQ_LOG_CORREL_ID_LENGTH	8	X'00000008'
MQ_LOG_EXTENT_NAME_LENGTH	24	X'00000018'
MQ_LOG_PATH_LENGTH	1024	X'00000400'
MQ_LRSN_LENGTH	12	X'0000000C'
MQ_ORIGIN_NAME_LENGTH	8	X'00000008'
MQ_PSB_NAME_LENGTH	8	X'00000008'
MQ_PST_ID_LENGTH	8	X'00000008'
MQ_Q_MGR_CPF_LENGTH	4	X'00000004'
MQ_RESPONSE_ID_LENGTH	24	X'00000018'
MQ_RBA_LENGTH	16	X'00000010'
MQ_SECURITY_PROFILE_LENGTH	40	X'00000028'
MQ_SERVICE_COMPONENT_LENGTH	48	X'00000030'
MQ_SUB_NAME_LENGTH	10240	X'00002800'
MQ_SYSP_SERVICE_LENGTH	32	X'00000020'
MQ_NÁZEV_SYSTÉMU_DÉLKA	8	X'00000008'
MQ_TASK_NUMBER_LENGTH	8	X'00000008'
MQ_TPIPE_PFX_LENGTH	4	X'00000004'
MQ_UOW_ID_LENGTH	256	X'00000100'
MQ_USER_DATA_LENGTH	10240	X'00002800'
MQ_VOLSER_LENGTH	6	X'00000006'

MQACH_ * (struktura záhlaví oblasti řetězce uživatelské procedury rozhraní API)

Tabulka 9. Struktury konstant	
Název	**Struktura**
ID_STRUC_MQACH_STRUC_ID	"ACH¬"
MQACH_STRUC_ID_ARRAY	'A','C','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 10. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQACH_VERSION_1	1	X'00000001'
MQACH_CURRENT_VERSION	1	X'00000001'
MQACH_LENGTH_1	(value differs by platform or version)	(value differs by platform or version)

Tabulka 10. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQACH_CURRENT_LENGTH	(value differs by platform or version)	(value differs by platform or version)

MQACT_ * (token evidence)

Tabulka 11. Názvy konstant a hodnoty	
Název	**Hodnota**
MQACT_NONE	X'00...00' (32 nulových hodnot)
MQACT_NONE_ARRAY	'\0','\0',... (32 nulových hodnot)

MQACT_ * (Volby akce formátu příkazu)

Tabulka 12. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQACT_FORCE_REMOVE	1	X'00000001'
MQACT_ADVANCE_LOG	2	X'00000002'
MQACT_COLLECT_STATISTICS	3	X'00000003'
MQACT_PUBSUB	4	X'00000004'

MQACTP_ * (Akce)

Tabulka 13. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQACTP_NEW	0	X'00000000'
MQACTP_FORWARD	1	X'00000001'
MQACTP_REPLY	2	X'00000002'
MQACTP_REPORT	3	X'00000003'

MQACTT_ * (typy tokenů evidence)

Tabulka 14. Hodnoty konstant	
Název	**Hexadecimální hodnota**
MQACTT_UNKNOWN (neznámý)	X'00'
MQACTT_CICS_LUOW_ID	X'01'
MQACTT_OS2_DEFAULT	X'04'
VÝCHOZÍ-MQACTT_DOS_DEFAULT	X'05'
MQACTT_UNIX_NUMERIC_ID	X'06'
MQACTT_OS400_ACCOUNT_TOKEN	X'08'
MQACTT_WINDOWS_DEFAULT	X'09'
MQACTT_NT_SECURITY_ID	X'0B'
MQACTT_USER	X'19'

MQADOPT_ * (Přijmout nové kontroly MCA a přijmout nové typy MCA)

Převzít nové kontroly MCA

Tabulka 15. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'

Převzít nové typy MCA

Tabulka 16. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQADOPT_TYPE_NO	0	X'00000000'
MQADOPT_TYPE_ALL	1	X'00000001'
MQADOPT_TYPE_SVR	2	X'00000002'
MQADOPT_TYPE_SDR	4	X'00000004'
MQADOPT_TYPE_RCVR	8	X'00000008'
MQADOPT_TYPE_CLUSRCVR	16	X'00000010'

MQAIR_ * (struktura záznamu ověřovacích informací)

Tabulka 17. Struktury konstant	
Název	**Struktura**
MQAIR_STRUC_ID	"AIR¬"
MQAIR_STRUC_ID_POLE	'A','I','R','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 18. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQAIR_VERSION_1	1	X'00000001'
MQAIR_VERSION_2	2	X'00000002'
MQAIR_CURRENT_VERSION	2	X'00000002'

MQAIT_ * (Typ ověřovacích informací)

Tabulka 19. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQAIT_ALL	0	X'00000000'
MQAIT_CRL_LDAP	1	X'00000001'
MQAIT_OCSP	2	X'00000002'
MQAIT_IDPW_OS	3	X'00000003'
MQAIT_IDPW_LDAP	4	X'00000004'

MQAS_ * (Formát příkazu Asynchronní hodnoty stavu)

Tabulka 20. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQAS_NONE	0	X'00000000'
MQAS_STARTED	1	X'00000001'
MQAS_START_WAIT	2	X'00000002'
MQAS_ZASTAVENO	3	X'00000003'
MQAS_POZASTAVENO	4	X'00000004'
MQAS_SUSPENDED_TEMPORARY	5	X'00000005'
MQAS_ACTIVE	6	X'00000006'
MQAS_INACTIVE	7	X'00000007'

MQAT_ * (Vložit typy aplikací)

Tabulka 21. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQAT_UNKNOWN (neznámý)	-1	X'FFFFFFFF'
MQAT_NO_CONTEXT	0	X'00000000'
MQAT_CICS	1	X'00000001'
MQAT_MVS	2	X'00000002'
MQAT_OS390	2	X'00000002'
MQAT_ZOS	2	X'00000002'
MQAT_IMS	3	X'00000003'
MQAT_OS2	4	X'00000004'
MQAT_DOS	5	X'00000005'
MQAT_AIX-operační systém	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_STRÁŽCE	13	X'0000000D'
MQAT_NSK	13	X'0000000D'
MQAT_VOS	14	X'0000000E'
MQAT_OPEN_TP1	15	X'0000000F'
MQAT_VM	18	X'00000012'
MQAT_IMS_BRIDGE	19	X'00000013'
MQAT_XCF	20	X'00000014'
MQAT_CICS_BRIDGE	21	X'00000015'
MQAT_NOTES_AGENT	22	X'00000016'

Tabulka 21. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQAT_TPF	23	X'00000017'
MQAT_USER	25	X'00000019'
MQAT_BROKER	26	X'0000001A'
MQAT_QMGR_PUBLISH	26	X'0000001A'
MQAT_JAVA	28	X'0000001C'
MQAT_DQM	29	X'0000001D'
Iniciátor 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'
VÝCHOZÍ	(value differs by platform or version)	(value differs by platform or version)
MQAT_USER_FIRST	65536	X'00010000'
MQAT_USER_LAST	999999999	X'3B9AC9FF'

MQAUTH_ * (Hodnoty oprávnění formátu příkazu)

Tabulka 22. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQAUTH_NONE	0	X'00000000'
MQAUTH_ALT_USER_AUTHORITY	1	X'00000001'
MQAUTH_BROWSE	2	X'00000002'
MQAUTH_CHANGE-změna	3	X'00000003'
MQAUTH_CLEAR	4	X'00000004'
MQAUTH_CONNECT-připojení	5	X'00000005'
MQAUTH_CREATE	6	X'00000006'
MQAUTH_DELETE	7	X'00000007'
MQAUTH_DISPLAY-zobrazení	8	X'00000008'
MQAUTH_INPUT	9	X'00000009'
DOTAZ_MQAUTH_INQUIRE	10	X'0000000A'
MQAUTH_OUTPUT	11	X'0000000B'
MQAUTH_PASS_ALL_CONTEXT	12	X'0000000C'
MQAUTH_PASS_IDENTITY_CONTEXT	13	X'0000000D'
MQAUTH_SET	14	X'0000000E'
MQAUTH_SET_ALL_CONTEXT	15	X'0000000F'
MQAUTH_SET_IDENTITY_CONTEXT	16	X'00000010'
MQAUTH_CONTROL	17	X'00000011'
MQAUTH_CONTROL_EXTENDED	18	X'00000012'
MQAUTH_PUBLISH	19	X'00000013'
MQAUTH_SUBSCRIBE	20	X'00000014'

Tabulka 22. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQAUTH_RESUME	21	X'00000015'
MQAUTH_SYSTEM	22	X'00000016'

MQAUTHOPT_ * (Volby oprávnění ve formátu příkazu)

Tabulka 23. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQAUTHOPT_KUMULATIVNÍ	256	X'00000100'
MQAUTHOPT_ENTITY_EXPLICIT	1	X'00000001'
MQAUTHOPT_ENTITY_SET	2	X'00000002'
MQAUTHOPT_NAME_ALL_MATCHING	32	X'00000020'
MQAUTHOPT_NAME_AS_ZÁSTUPNÝ znak	64	X'00000040'
MQAUTHOPT_NAME_EXPLICIT	16	X'00000010'

MQAXC_ * (struktura kontextu uživatelské procedury rozhraní API)

Tabulka 24. Struktury konstant	
Název	**Struktura**
MQAXC_STRUC_ID	"AXC¬"
MQAXC_STRUC_ID_ARRAY	'A','X','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 25. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQAXC_VERSION_1	1	X'00000001'
MQAXC_CURRENT_VERSION	1	X'00000001'

MQAXP_ * (struktura parametrů uživatelské procedury rozhraní API)

Tabulka 26. Struktury konstant	
Název	**Struktura**
MQAXP_STRUC_ID	"AXP¬"
MQAXP_STRUC_ID_ARRAY	'A','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 27. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQAXP_VERSION_1	1	X'00000001'
MQAXP_VERSION_2	2	X'00000002'
MQAXP_CURRENT_VERSION	2	X'00000002'

MQBA_ * (Bajtové selektory atributů)

Tabulka 28. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQBA_FIRST	6001	X'00001771'
MQBA_LAST	8000	X'00001F40'

MQBACF_ * (Typy parametrů bajtového formátu příkazu)

Tabulka 29. Hodnoty konstant

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

MQBL_ * (Délka vyrovnávací paměti pro řetězec mqAdda řetězec mqSet)

Tabulka 30. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQBL_NULL_UKONČENO	-1	X'FFFFFFFF'

MQBMHO_ * (Volby a struktura popisovače vyrovnávací paměti pro zprávu)

Struktura voleb z vyrovnávací paměti do popisovače zprávy

Tabulka 31. Struktury konstant

Název	Struktura
MQBMHO_STRUC_ID	"BMHO"
MQBMHO_STRUC_ID_ARRAY	'B','M','H','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 32. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBMHO_VERSION_1	1	X'00000001'
MQBMHO_CURRENT_VERSION	1	X'00000001'

Volby popisovače vyrovnávací paměti pro zprávu

Tabulka 33. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBMHO_NONE	0	X'00000000'
MQBMHO_DELETE_PROPERTIES	1	X'00000001'

MQBND_ * (výchozí vazby)

Tabulka 34. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBND_BIND_ON_OPEN	0	X'00000000'
MQBND_BIND_NOT_FIXED	1	X'00000001'
MQBND_BIND_ON_GROUP	2	X'00000002'

MQBO_ * (začátek voleb a struktura)

Struktura voleb začátku

Tabulka 35. Struktury konstant	
Název	**Struktura**
ID_STRUC_MQBO_STRUC_ID	"BO¬¬"
MQBO_STRUC_ID_ARRAY	'B','O','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 36. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBO_VERSION_1	1	X'00000001'
MQBO_CURRENT_VERSION	1	X'00000001'

Volby začátku

Tabulka 37. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBO_NONE	0	X'00000000'

MQBT_ * (Typy mostu ve formátu příkazů)

Tabulka 38. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBT_OTMA	1	X'00000001'

MQCA_ * (selektory znakových atributů)

Tabulka 39. Hodnoty konstant

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

Tabulka 39. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCA_INITIATION_Q_NAME	2008	X'000007D8'
MQCA_LAST	4000	X'00000FA0'
MQCA_LAST_USED	2109	X'0000083D'
MQCA_LDAP_PASSWORD	2048	X'00000800'
MQCA_LDAP_USER_NAME	2047	X'000007FF'
MQCA_LU_GROUP_NAME	2072	X'00000818'
MQCA_LU_NAME	2073	X'00000819'
MQCA_LU62_ARM_SUFFIX	2074	X'0000081A'
MQCA_MODEL_DURABLE_Q	2096	X'00000830'
MQCA_MODEL_NON_DURABLE_Q	2097	X'00000831'
MQCA_MONITOR_Q_NAME	2066	X'00000812'
MQCA_NAMELIST_DESC	2009	X'000007D9'
MQCA_NAMELIST_NAME	2010	X'000007DA'
MQCA_NAMES	2020	X'000007E4'
MQCA_PARENT	2102	X'00000836'
MQCA_PASS_TICKET_APPL	2086	X'00000826'
MQCA_PROCESS_DESC	2011	X'000007DB'
MQCA_PROCESS_NAME	2012	X'000007DC'
MQCA_Q_DESC	2013	X'000007DD'
MQCA_Q_MGR_DESC	2014	X'000007DE'
MQCA_Q_MGR_IDENTIFIER	2032	X'000007F0'
MQCA_Q_MGR_NAME	2015	X'000007DF'
MQCA_Q_NAME	2016	X'000007E0'
MQCA_QSG_NAME	2040	X'000007F8'
MQCA_REMOTE_Q_MGR_NAME	2017	X'000007E1'
MQCA_REMOTE_Q_NAME	2018	X'000007E2'
MQCA_REPOSITORY_NAME	2035	X'000007F3'
MQCA_REPOSITORY_NAMELIST	2036	X'000007F4'
MQCA_RESUME_DATE	2098	X'00000832'
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'

Tabulka 39. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCA_SSL_CRYPTO_HARDWARE	2051	X'00000803'
MQCA_SSL_KEY_LIBRARY	2069	X'00000815'
MQCA_SSL_KEY_MEMBER	2070	X'00000816'
MQCA_SSL_KEY_REPOSITORY	2049	X'00000801'
MQCA_STORAGE_CLASS	2022	X'000007E6'
MQCA_STORAGE_CLASS_DESC	2042	X'000007FA'
MQCA_SYSTEM_LOG_Q_NAME	2065	X'00000811'
MQCA_TCP_NAME	2075	X'0000081B'
MQCA_TOPIC_DESC	2093	X'0000082D'
MQCA_TOPIC_NAME	2092	X'0000082C'
MQCA_TOPIC_STRING_FILTER	2108	X'0000083C'
MQCA_TOPIC_STRING	2094	X'0000082E'
MQCA_TPIPE_NAME	2085	X'00000825'
MQCA_TRIGGER_CHANNEL_NAME	2064	X'00000810'
MQCA_TRIGGER_DATA	2023	X'000007E7'
MQCA_TRIGGER_PROGRAM_NAME	2062	X'0000080E'
MQCA_TRIGGER_TERM_ID	2063	X'0000080F'
MQCA_TRIGGER_TRANS_ID	2061	X'0000080D'
MQCA_USER_DATA	2021	X'000007E5'
MQCA_USER_LIST	4000	X'00000FA0'
MQCA_VERSION	2120	X'00000848'
MQCA_XCF_GROUP_NAME	2043	X'000007FB'
MQCA_XCF_MEMBER_NAME	2044	X'000007FC'
MQCA_XMIT_Q_NAME	2024	X'000007E8'

MQCACF_ * (Formát příkazu Typy znakových parametrů)

Tabulka 40. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_FIRST	3001	X'00000BB9'
MQCACF_FROM_Q_NAME	3001	X'00000BB9'
MQCACF_TO_Q_NAME	3002	X'00000BBA'
MQCACF_FROM_PROCESS_NAME	3003	X'00000BBB'
MQCACF_TO_PROCESS_NAME	3004	X'00000BBC'
MQCACF_FROM_NAMELIST_NAME	3005	X'00000BBD'
MQCACF_TO_NAMELIST_NAME	3006	X'00000BBE'
MQCACF_FROM_CHANNEL_NAME	3007	X'00000BBF'
MQCACF_TO_CHANNEL_NAME	3008	X'00000BC0'
MQCACF_FROM_AUTH_INFO_NAME	3009	X'00000BC1'
MQCACF_TO_AUTH_INFO_NAME	3010	X'00000BC2'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_Q_NAMES	3011	X'00000BC3'
MQCACF_PROCESS_NAMES	3012	X'00000BC4'
MQCACF_NAMELIST_NAMES	3013	X'00000BC5'
MQCACF_ESCAPE_TEXT	3014	X'00000BC6'
MQCACF_LOCAL_Q_NAMES	3015	X'00000BC7'
MQCACF_MODEL_Q_NAMES	3016	X'00000BC8'
MQCACF_ALIAS_Q_NAMES	3017	X'00000BC9'
MQCACF_REMOTE_Q_NAMES	3018	X'00000BCA'
MQCACF_SENDER_CHANNEL_NAMES	3019	X'00000BCB'
MQCACF_SERVER_CHANNEL_NAMES	3020	X'00000BCC'
MQCACF_REQUESTER_CHANNEL_NAMES	3021	X'00000BCD'
MQCACF_RECEIVER_CHANNEL_NAMES	3022	X'00000BCE'
MQCACF_OBJECT_Q_MGR_NAME	3023	X'00000BCF'
MQCACF_APPL_NAME	3024	X'00000BD0'
Identifikátor uživatele MQCACF_USER_IDENTIFIER	3025	X'00000BD1'
MQCACF_AUX_ERROR_DATA_STR_1	3026	X'00000BD2'
MQCACF_AUX_ERROR_DATA_STR_2	3027	X'00000BD3'
MQCACF_AUX_ERROR_DATA_STR_3	3028	X'00000BD4'
MQCACF_BRIDGE_NAME	3029	X'00000BD5'
MQCACF_STREAM_NAME	3030	X'00000BD6'
MQCACF_TOPIC	3031	X'00000BD7'
MQCACF_PARENT_Q_MGR_NAME	3032	X'00000BD8'
MQCACF_CORREL_ID	3033	X'00000BD9'
MQCACF_PUBLISH_TIMESTAMP	3034	X'00000BDA'
MQCACF_STRING_DATA	3035	X'00000BDB'
MQCACF_SUPPORTED_STREAM_NAME	3036	X'00000BDC'
MQCACF_REG_TOPIC	3037	X'00000BDD'
MQCACF_REG_TIME	3038	X'00000BDE'
MQCACF_REG_USER_ID	3039	X'00000BDF'
MQCACF_přelomu_Q_MGR_NAME	3040	X'00000BE0'
MQCACF_REG_STREAM_NAME	3041	X'00000BE1'
MQCACF_REG_Q_MGR_NAME	3042	X'00000BE2'
MQCACF_REG_Q_NAME	3043	X'00000BE3'
MQCACF_REG_CORREL_ID	3044	X'00000BE4'
MQCACF_EVENT_USER_ID	3045	X'00000BE5'
MQCACF_OBJECT_NAME	3046	X'00000BE6'
MQCACF_EVENT_Q_MGR	3047	X'00000BE7'
MQCACF_AUTH_INFO_NAMES	3048	X'00000BE8'
MQCACF_EVENT_APPL_IDENTITY	3049	X'00000BE9'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_EVENT_APPL_NAME	3050	X'00000BEA'
MQCACF_EVENT_APPL_ORIGIN	3051	X'00000BEB'
MQCACF_SUBSCRIPTION_NAME	3052	X'00000BEC'
MQCACF_REG_SUB_NAME	3053	X'00000BED'
MQCACF_SUBSCRIPTION_IDENTITY	3054	X'00000BEE'
MQCACF_REG_SUB_IDENTITY	3055	X'00000BEF'
MQCACF_SUBSCRIPTION_USER_DATA	3056	X'00000BF0'
MQCACF_REG_SUB_USER_DATA	3057	X'00000BF1'
MQCACF_APPL_TAG	3058	X'00000BF2'
MQCACF_DATA_SET_NAME	3059	X'00000BF3'
MQCACF_UOW_START_DATE	3060	X'00000BF4'
MQCACF_UOW_START_TIME	3061	X'00000BF5'
MQCACF_UOW_LOG_START_DATE	3062	X'00000BF6'
MQCACF_UOW_LOG_START_TIME	3063	X'00000BF7'
MQCACF_UOW_LOG_EXTENT_NAME	3064	X'00000BF8'
MQCACF_PRINCIPAL_ENTITY_NAMES	3065	X'00000BF9'
MQCACF_GROUP_ENTITY_NAMES	3066	X'00000BFA'
MQCACF_AUTH_PROFILE_NAME	3067	X'00000BFB'
MQCACF_ENTITY_NAME	3068	X'00000BFC'
MQCACF_SERVICE_COMPONENT	3069	X'00000BFD'
MQCACF_RESPONSE_Q_MGR_NAME	3070	X'00000BFE'
MQCACF_CURRENT_LOG_EXTENT_NAME	3071	X'00000BFF'
MQCACF_RESTART_LOG_EXTENT_NAME	3072	X'00000C00'
MQCACF_MEDIA_LOG_EXTENT_NAME	3073	X'00000C01'
MQCACF_LOG_PATH	3074	X'00000C02'
MQCACF_COMMAND_MQSC	3075	X'00000C03'
MQCACF_Q_MGR_CPF	3076	X'00000C04'
MQCACF_USAGE_LOG_RBA	3078	X'00000C06'
MQCACF_USAGE_LOG_LRSN	3079	X'00000C07'
MQCACF_COMMAND_SCOPE	3080	X'00000C08'
MQCACF_ASID	3081	X'00000C09'
MQCACF_PSB_NAME	3082	X'00000C0A'
MQCACF_PST_ID	3083	X'00000C0B'
MQCACF_TASK_NUMBER	3084	X'00000C0C'
MQCACF_TRANSACTION_ID	3085	X'00000C0D'
MQCACF_Q_MGR_UOW_ID	3086	X'00000C0E'
MQCACF_ORIGIN_NAME	3088	X'00000C10'
MQCACF_ENV_INFO	3089	X'00000C11'
MQCACF_SECURITY_PROFILE	3090	X'00000C12'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_CONFIGURATION_DATE	3091	X'00000C13'
MQCACF_CONFIGURATION_TIME	3092	X'00000C14'
MQCACF_FROM_CF_STRUC_NAME	3093	X'00000C15'
MQCACF_TO_CF_STRUC_NAME	3094	X'00000C16'
MQCACF_CF_STRUC_NAMES	3095	X'00000C17'
MQCACF_FAIL_DATE	3096	X'00000C18'
MQCACF_FAIL_TIME	3097	X'00000C19'
MQCACF_BACKUP_DATE	3098	X'00000C1A'
MQCACF_BACKUP_TIME	3099	X'00000C1B'
MQCACF_SYSTEM_NAME	3100	X'00000C1C'
MQCACF_CF_STRUC_BACKUP_START	3101	X'00000C1D'
MQCACF_CF_STRUC_BACKUP_END	3102	X'00000C1E'
MQCACF_CF_STRUC_LOG_Q_MGRS	3103	X'00000C1F'
MQCACF_FROM_STORAGE_CLASS	3104	X'00000C20'
MQCACF_TO_STORAGE_CLASS	3105	X'00000C21'
MQCACF_STORAGE_CLASS_NAMES	3106	X'00000C22'
MQCACF_DSG_NAME	3108	X'00000C24'
MQCACF_DB2_NAME	3109	X'00000C25'
MQCACF_SYSP_CMD_USER_ID	3110	X'00000C26'
MQCACF_SYSP_OTMA_GROUP	3111	X'00000C27'
MQCACF_SYSP_OTMA_MEMBER	3112	X'00000C28'
MQCACF_SYSP_OTMA_DRU_EXIT	3113	X'00000C29'
MQCACF_SYSP_OTMA_TPIPE_PFX	3114	X'00000C2A'
MQCACF_SYSP_ARCHIVE_PFX1	3115	X'00000C2B'
MQCACF_SYSP_ARCHIVE_UNIT1	3116	X'00000C2C'
MQCACF_SYSP_LOG_CORREL_ID	3117	X'00000C2D'
MQCACF_SYSP_UNIT_VOLSER	3118	X'00000C2E'
MQCACF_SYSP_Q_MGR_TIME	3119	X'00000C2F'
MQCACF_SYSP_Q_MGR_DATE	3120	X'00000C30'
MQCACF_SYSP_Q_MGR_RBA	3121	X'00000C31'
MQCACF_SYSP_LOG_RBA	3122	X'00000C32'
MQCACF_SYSP_SERVICE	3123	X'00000C33'
MQCACF_FROM_LISTENER_NAME	3124	X'00000C34'
MQCACF_TO_LISTENER_NAME	3125	X'00000C35'
MQCACF_FROM_SERVICE_NAME	3126	X'00000C36'
MQCACF_TO_SERVICE_NAME	3127	X'00000C37'
MQCACF_LAST_PUT_DATE	3128	X'00000C38'
MQCACF_LAST_PUT_TIME	3129	X'00000C39'
MQCACF_LAST_GET_DATE	3130	X'00000C3A'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_LAST_GET_TIME	3131	X'00000C3B'
MQCACF_OPERATION_DATE	3132	X'00000C3C'
MQCACF_OPERATION_TIME	3133	X'00000C3D'
MQCACF_ACTIVITY_DESC	3134	X'00000C3E'
MQCACF_APPL_IDENTITY_DATA	3135	X'00000C3F'
MQCACF_APPL_ORIGIN_DATA	3136	X'00000C40'
MQCACF_PUT_DATE	3137	X'00000C41'
MQCACF_PUT_TIME	3138	X'00000C42'
MQCACF_REPLY_TO_Q	3139	X'00000C43'
MQCACF_REPLY_TO_Q_MGR	3140	X'00000C44'
MQCACF_RESOLVED_Q_NAME	3141	X'00000C45'
MQCACF_STRUC_ID	3142	X'00000C46'
MQCACF_VALUE_NAME	3143	X'00000C47'
MQCACF_SERVICE_START_DATE	3144	X'00000C48'
MQCACF_SERVICE_START_TIME	3145	X'00000C49'
MQCACF_SYSP_OFFLINE_RBA	3146	X'00000C4A'
MQCACF_SYSP_ARCHIVE_PFX2	3147	X'00000C4B'
MQCACF_SYSP_ARCHIVE_UNIT2	3148	X'00000C4C'
MQCACF_TO_TOPIC_NAME	3149	X'00000C4D'
MQCACF_FROM_TOPIC_NAME	3150	X'00000C4E'
MQCACF_TOPIC_NAMES	3151	X'00000C4F'
MQCACF_SUB_NAME	3152	X'00000C50'
MQCACF_DESTINATION_Q_MGR	3153	X'00000C51'
MQCACF_DESTINATION	3154	X'00000C52'
MQCACF_SUB_USER_ID	3156	X'00000C54'
MQCACF_SUB_USER_DATA	3159	X'00000C57'
MQCACF_SUB_SELECTOR	3160	X'00000C58'
MQCACF_LAST_PUB_DATE	3161	X'00000C59'
MQCACF_LAST_PUB_TIME	3162	X'00000C5A'
MQCACF_FROM_SUB_NAME	3163	X'00000C5B'
MQCACF_TO_SUB_NAME	3164	X'00000C5C'
MQCACF_LAST_MSG_TIME	3167	X'00000C5F'
MQCACF_LAST_MSG_DATE	3168	X'00000C60'
MQCACF_SUBSCRIPTION_POINT	3169	X'00000C61'
MQCACF_FILTER	3170	X'00000C62'
MQCACF_NONE	3171	X'00000C63'
MQCACF_ADMIN_TOPIC_NAMES	3172	X'00000C64'
MQCACF_ROUTING_FINGER_PRINT	3173	X'00000C65'
MQCACF_APPL_DESC	3174	X'00000C66'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_Q_MGR_START_DATE	3175	X'00000C67'
MQCACF_Q_MGR_START_TIME	3176	X'00000C68'
MQCACF_FROM_COMM_INFO_NAME	3177	X'00000C69'
MQCACF_TO_COMM_INFO_NAME	3178	X'00000C6A'
MQCACF_CF_OFFLOAD_SIZE1	3179	X'00000C6B'
MQCACF_CF_OFFLOAD_SIZE2	3180	X'00000C6C'
MQCACF_CF_OFFLOAD_SIZE3	3181	X'00000C6D'
MQCACF_CF_SMDS_GENERIC_NAME	3182	X'00000C6E'
MQCACF_CF_SMDS	3183	X'00000C6F'
MQCACF_RECOVERY_DATE	3184	X'00000C70'
MQCACF_RECOVERY_TIME	3185	X'00000C71'
MQCACF_CF_SMDSCONN	3186	X'00000C72'
MQCACF_CF_STRUC_NAME	3187	X'00000C73'
MQCACF_ALTERNATE_USERID	3188	X'00000C74'
MQCACF_CHAR_ATTRS	3189	X'00000C75'
MQCACF_DYNAMIC_Q_NAME	3190	X'00000C76'
MQCACF_HOST_NAME	3191	X'00000C77'
MQCACF_MQCB_NAME	3192	X'00000C78'
MQCACF_OBJECT_STRING	3193	X'00000C79'
MQCACF_RESOLVED_LOCAL_Q_MGR	3194	X'00000C7A'
MQCACF_RESOLVED_LOCAL_Q_NAME	3195	X'00000C7B'
MQCACF_RESOLVED_OBJECT_STRING	3196	X'00000C7C'
MQCACF_RESOLVED_Q_MGR	3197	X'00000C7D'
MQCACF_SELECTION_STRING	3198	X'00000C7E'
MQCACF_XA_INFO	3199	X'00000C7F'
MQCACF_APPL_FUNCTION	3200	X'00000C80'
MQCACF_XQH_REMOTE_Q_NAME	3201	X'00000C81'
MQCACF_XQH_REMOTE_Q_MGR	3202	X'00000C82'
MQCACF_XQH_PUT_TIME	3203	X'00000C83'
MQCACF_XQH_PUT_DATE	3204	X'00000C84'
MQCACF_EXCL_OPERATOR_MESSAGES	3205	X'00000C85'
MQCACF_CSP_USER_IDENTIFIER	3206	X'00000C86'
MQCACF_AMQP_CLIENT_ID	3207	X'00000C87'
MQCACF_ARCHIVE_LOG_EXTENT_NAME	3208	X'00000C88'
MQCACF_APPL_IMMOVABLE_DATE	3209	X'00000C89'
MQCACF_APPL_IMMOVABLE_TIME	3210	X'00000C8A'
MQCACF_NHA_INSTANCE_NAME	3211	X'00000C8B'
MQCACF_LAST_USED	3211	X'00000C8B'

MQCACH_ * (Formát příkazu Typy parametrů znakového kanálu)

Tabulka 41. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACH_FIRST	3501	X'00000DAD'
MQCACH_CHANNEL_NAME	3501	X'00000DAD'
MQCACH_DESC	3502	X'00000DAE'
MQCACH_MODE_NAME	3503	X'00000DAF'
MQCACH_TP_NAME	3504	X'00000DB0'
MQCACH_XMIT_Q_NAME	3505	X'00000DB1'
MQCACH_CONNECTION_NAME	3506	X'00000DB2'
MQCACH_MCA_NAME	3507	X'00000DB3'
MQCACH_SEC_EXIT_NAME	3508	X'00000DB4'
MQCACH_MSG_EXIT_NAME	3509	X'00000DB5'
MQCACH_SEND_EXIT_NAME	3510	X'00000DB6'
MQCACH_RCV_EXIT_NAME	3511	X'00000DB7'
MQCACH_CHANNEL_NAMES	3512	X'00000DB8'
MQCACH_SEC_EXIT_USER_DATA	3513	X'00000DB9'
MQCACH_MSG_EXIT_USER_DATA	3514	X'00000DBA'
MQCACH_SEND_EXIT_USER_DATA	3515	X'00000DBB'
MQCACH_RCV_EXIT_USER_DATA	3516	X'00000DBC'
MQCACH_USER_ID	3517	X'00000DBD'
MQCACH_PASSWORD	3518	X'00000DBE'
MQCACH_LOCAL_ADDRESS	3520	X'00000DC0'
MQCACH_LOCAL_NAME	3521	X'00000DC1'
MQCACH_LAST_MSG_TIME	3524	X'00000DC4'
MQCACH_LAST_MSG_DATE	3525	X'00000DC5'
MQCACH_MCA_USER_ID	3527	X'00000DC7'
MQCACH_CHANNEL_START_TIME	3528	X'00000DC8'
MQCACH_CHANNEL_START_DATE	3529	X'00000DC9'
MQCACH_MCA_JOB_NAME	3530	X'00000DCA'
MQCACH_LAST_LUWID	3531	X'00000DCB'
MQCACH_CURRENT_LUWID	3532	X'00000DCC'
MQCACH_FORMAT_NAME	3533	X'00000DCD'
MQCACH_MR_EXIT_NAME	3534	X'00000DCE'
MQCACH_MR_EXIT_USER_DATA	3535	X'00000DCF'
MQCACH_SSL_CIPHER_SPEC	3544	X'00000DD8'
MQCACH_SSL_PEER_NAME	3545	X'00000DD9'
MQCACH_SSL_HANDSHAKE_STAGE	3546	X'00000DDA'
MQCACH_SSL_SHORT_PEER_NAME	3547	X'00000DDB'
MQCACH_REMOTE_APPL_TAG	3548	X'00000DDC'
MQCACH_SSL_CERT_USER_ID	3549	X'00000DDD'

Tabulka 41. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCACH_SSL_CERT_ISSUER_NAME	3550	X'00000DDE'
MQCACH_LU_NAME	3551	X'00000DDF'
MQCACH_IP_ADDRESS	3552	X'00000DE0'
MQCACH_TCP_NAME	3553	X'00000DE1'
MQCACH_LISTENER_NAME	3554	X'00000DE2'
MQCACH_LISTENER_DESC	3555	X'00000DE3'
MQCACH_LISTENER_START_DATE	3556	X'00000DE4'
MQCACH_LISTENER_START_TIME	3557	X'00000DE5'
MQCACH_SSL_KEY_RESET_DATE	3558	X'00000DE6'
MQCACH_SSL_KEY_RESET_TIME	3559	X'00000DE7'
MQCACH_LAST_USED	3559	X'00000DE7'

MQCADSD_ * (deskriptory ADS záhlaví informací CICS)

Tabulka 42. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCADSD_NONE	0	X'00000000'
MQCADSD_SEND	1	X'00000001'
MQCADSD_RECV	16	X'00000010'
MQCADSD_MSGFORMAT	256	X'00000100'

MQCAFTY_ * (Hodnoty afinity připojení)

Tabulka 43. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCAFTY_NONE	0	X'00000000'
MQCAFTY_PREFERRED	1	X'00000001'

MQCAMO_ * (Formát příkazu Typy parametrů monitorování znaků)

Tabulka 44. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCAMO_FIRST	2701	X'00000A8D'
MQCAMO_CLOSE_DATE	2701	X'00000A8D'
MQCAMO_CLOSE_TIME	2702	X'00000A8E'
MQCAMO_CONN_DATE	2703	X'00000A8F'
MQCAMO_CONN_TIME	2704	X'00000A90'
MQCAMO_DISC_DATE	2705	X'00000A91'
MQCAMO_DISC_TIME	2706	X'00000A92'
MQCAMO_END_DATE	2707	X'00000A93'
MQCAMO_END_TIME	2708	X'00000A94'
MQCAMO_OPEN_DATE	2709	X'00000A95'
MQCAMO_OPEN_TIME	2710	X'00000A96'

Tabulka 44. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCAMO_START_DATE	2711	X'00000A97'
MQCAMO_START_TIME	2712	X'00000A98'
MQCAMO_LAST_USED	2712	X'00000A98'

MQCBC_ * (struktura konstant MQCBC)

Tabulka 45. Struktury konstant	
Název	**Struktura**
MQCBC_STRUC_ID	"CBC¬"
MQCBC_STRUC_ID_ARRAY	'C','B','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 46. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBC_VERSION_1	1	X'00000001'
MQCBC_CURRENT_VERSION	1	X'00000001'

MQCBCF_ * (příznaky konstant MQCBC)

Tabulka 47. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBCF_NONE	0	X'00000000'
MQCBCF_READA_BUFFER_EMPTY	1	X'00000001'

MQCBCT_ * (typ zpětného volání konstant MQCBC)

Tabulka 48. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBCT_START_CALL	1	X'00000001'
MQCBCT_STOP_CALL	2	X'00000002'
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_ * (struktura konstant MQCBD)

Tabulka 49. Struktury konstant	
Název	**Struktura**
MQCBD_STRUC_ID	"CBD¬"
MQCBD_STRUC_ID_ARRAY	'C','B','D','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 50. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBD_VERSION_1	1	X'00000001'
MQCBD_CURRENT_VERSION	1	X'00000001'

MQCBDO_ * (Volby zpětného volání konstant MQCBD)

Tabulka 51. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBDO_NONE	0	X'00000000'
MQCBDO_START_CALL	1	X'00000001'
MQCBDO_STOP_CALL	4	X'00000004'
MQCBDO_REGISTER_CALL	256	X'00000100'
MQCBDO_DEREGISTER_CALL	512	X'00000200'
MQCBDO_FAIL_IF_QUIESCING	8192	X'00002000'

MQCBO_ * (Volby vytvoření-Bag pro mqCreateBag)

Tabulka 52. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBO_NONE	0	X'00000000'
MQCBO_USER_BAG	0	X'00000000'
MQCBO_ADMIN_BAG	1	X'00000001'
MQCBO_COMMAND_BAG	16	X'00000010'
MQCBO_SYSTEM_BAG	32	X'00000020'
MQCBO_GROUP_BAG	64	X'00000040'
MQCBO_LIST_FORM_ALLOWED	2	X'00000002'
MQCBO_LIST_FORM_INHIBITED	0	X'00000000'
MQCBO_REORDER_AS_REQUIRED	4	X'00000004'
MQCBO_DO_NOT_REORDER	0	X'00000000'
MQCBO_CHECK_SELECTORS	8	X'00000008'
MQCBO_DO_NOT_CHECK_SELECTORS	0	X'00000000'

MQCBT_ * (konstanty MQCBD Toto je typ funkce zpětného volání)

Tabulka 53. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBT_MESSAGE_CONSUMER	1	X'00000001'
Obslužná rutina MQCBT_EVENT_HANDLER	2	X'00000002'

MQCC_ * (kódy dokončení)

Tabulka 54. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCC_OK	0	X'00000000'
MQCC_VAROVÁNÍ	1	X'00000001'

Tabulka 54. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCC_FAILED	2	X'00000002'
MQCC_NEZNÁMÝ	-1	X'FFFFFFFF'

MQCCSI_ * (identifikátory kódované znakové sady)

Tabulka 55. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCCSI_UNDEFINED	0	X'00000000'
VÝCHOZÍ-MQCCSI_DEFAULT	0	X'00000000'
MQCCSI_Q_MGR	0	X'00000000'
MQCCSI_INHERIT	-2	X'FFFFFFFE'
MQCCSI_EMBEDDED	-1	X'FFFFFFFF'
MQCCSI_APPL	-3	X'FFFFFFFD'

MQCCT_ * (Volby konverzační úlohy záhlaví informací CICS)

Tabulka 56. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCCT_YES	1	X'00000001'
MQCCT_NO	0	X'00000000'

MQCD_ * (Struktura definice kanálu)

Tabulka 57. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_VERSION_10	10	X'0000000A'
Multi MQCD_VERSION_11	11	X'0000000B'
Multi MQCD_CURRENT_VERSION	11	X'0000000B'
z/OS MQCD_VERSION_12	12	X'0000000C'
z/OS MQCD_CURRENT_VERSION	12	X'0000000C'
MQCD_LENGTH_4	(value differs by platform or version)	(value differs by platform or version)

Tabulka 57. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCD_LENGTH_5	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_6	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_7	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_8	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_9	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_10	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_11	(value differs by platform or version)	(value differs by platform or version)
z/OS MQCD_LENGTH_12	(value differs by platform or version)	(value differs by platform or version)
MQCD_CURRENT_LENGTH	(value differs by platform or version)	(value differs by platform or version)

MQCDC_ * (Převod dat kanálu)

Tabulka 58. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCDC_SENDER_CONVERSION	1	X'00000001'
MQCDC_NO_SENDER_CONVERSION	0	X'00000000'

MQCERT_ * (Typ zásady ověření certifikátu)

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_ * (příznaky schopnosti)

Tabulka 59. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCF_NONE	0	X'00000000'
MQCF_DIST_LISTS	1	X'00000001'

MQCFAC_ * (prostředek záhlaví informací CICS)

Tabulka 60. Názvy konstant a hodnoty	
Název	**Hexadecimální hodnota**
MQCFAC_NONE	X'00...00' (8 nulových hodnot)
MQCFAC_NONE_ARRAY	'\0','\0',... (8 nulových hodnot)

MQCFBF_ * (Struktura parametru filtru bajtového řetězce formátu příkazu)

Tabulka 61. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFBF_STRUC_LENGTH_FIXED	20	X'00000014'

MQCFBS_ * (struktura parametru bajtového řetězce formátu příkazu)

Tabulka 62. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFBS_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFC_ * (Volby řízení záhlaví formátu příkazu)

Tabulka 63. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFC_LAST	1	X'00000001'
MQCFC_NOT_LAST	0	X'00000000'

MQCFGR_ * (Struktura parametrů skupiny formátů příkazů)

Tabulka 64. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFGR_STRUC_LENGTH	16	X'00000010'

MQCFH_ * (Struktura záhlaví formátu příkazu)

Tabulka 65. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFH_STRUC_LENGTH	36	X'00000024'
MQCFH_VERSION_1	1	X'00000001'
MQCFH_VERSION_2	2	X'00000002'
MQCFH_VERSION_3	3	X'00000003'
MQCFH_CURRENT_VERSION	3	X'00000003'

MQCFIF_ * (Struktura parametrů filtru celého čísla formátu příkazu)

Tabulka 66. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIF_STRUC_LENGTH	20	X'00000014'

MQCFIL_ * (Struktura parametru seznamu celých čísel formátu příkazu)

Tabulka 67. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIL_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFIL64_* (Struktura parametrů 64bitového seznamu celých čísel ve formátu příkazu)

Tabulka 68. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIL64_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFIN_ * (Struktura celočíselného parametru formátu příkazu)

Tabulka 69. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIN_STRUC_LENGTH	16	X'00000010'

MQCFIN64_* (Struktura parametrů 64bitového celého čísla ve formátu příkazu)

Tabulka 70. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIN64_STRUC_LENGTH	24	X'00000018'

MQCFO_ * (Formát příkazů Obnovit volby úložiště a Formát příkazů Odebrat volby front)

Formát příkazu Aktualizovat volby úložiště

Tabulka 71. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFO_REFRESH_REPOSITORY_YES	1	X'00000001'
MQCFO_REFRESH_REPOSITORY_NO	0	X'00000000'

Formát příkazu Odebrat volby front

Tabulka 72. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFO_REMOVE_QUEUES_YES	1	X'00000001'
MQCFO_REMOVE_QUEUES_NO	0	X'00000000'

MQCFOP_ * (Operátory filtru formátu příkazu)

Tabulka 73. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFOP_LESS	1	X'00000001'
MQCFOP_EQUAL	2	X'00000002'
MQCFOP_VĚTŠÍ	4	X'00000004'
MQCFOP_NOT_LESS	6	X'00000006'
MQCFOP_NOT_EQUAL	5	X'00000005'
MQCFOP_NOT_VĚTŠÍ	3	X'00000003'
MQCFOP_LIKE	18	X'00000012'

Tabulka 73. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFOP_NOT_LIKE	21	X'00000015'
MQCFOP_CONTAINS	10	X'0000000A'
MQCFOP_VYLUČUJE	13	X'0000000D'
MQCFOP_CONTAINS_GEN	26	X'0000001A'
MQCFOP_EXCLUDES_GEN	29	X'0000001D'

MQCFR_ * (opravitelnost prostředku CF)

Tabulka 74. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFR_YES	1	X'00000001'
MQCFR_NO	0	X'00000000'

MQCFSF_ * (Struktura parametrů filtru řetězce formátu příkazu)

Tabulka 75. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFSF_STRUC_LENGTH_FIXED	24	X'00000018'

MQCFSL_ * (Struktura parametrů seznamu řetězců formátu příkazu)

Tabulka 76. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFSL_STRUC_LENGTH_FIXED	24	X'00000018'

MQCFST_ * (Struktura parametrů řetězce formátu příkazu)

Tabulka 77. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFST_STRUC_LENGTH_FIXED	20	X'00000014'

MQCFSTATUS_ * (Formát příkazu Stav prostředku CF)

Tabulka 78. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFSTATUS_NOT_FOUND	0	X'00000000'
MQCFSTATUS_ACTIVE	1	X'00000001'
MQCFSTATUS_IN_RECOVER	2	X'00000002'
MQCFSTATUS_IN_BACKUP	3	X'00000003'
MQCFSTATUS_FAILED	4	X'00000004'
MQCFSTATUS_NONE	5	X'00000005'
MQCFSTATUS_UNKNOWN	6	X'00000006'
MQCFSTATUS_ADMIN_NEKOMPLETNÍ	20	X'00000014'
MQCFSTATUS_NEVER_USED	21	X'00000015'
MQCFSTATUS_NO_BACKUP	22	X'00000016'

Tabulka 78. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFSTATUS_NOT_FAILED	23	X'00000017'
MQCFSTATUS_NEZOTAVITELNÉ	24	X'00000018'
MQCFSTATUS_XES_ERROR	25	X'00000019'

MQCFT_ * (Formát příkazu Typy struktury)

Tabulka 79. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFT_NONE	0	X'00000000'
MQCFT_COMMAND	1	X'00000001'
MQCFT_RESPONSE-odpověď	2	X'00000002'
MQCFT_INTEGER	3	X'00000003'
MQCFT_STRING	4	X'00000004'
MQCFT_INTEGER_LIST	5	X'00000005'
MQCFT_STRING_LIST	6	X'00000006'
MQCFT_EVENT	7	X'00000007'
MQCFT_USER	8	X'00000008'
MQCFT_BYTE_STRING	9	X'00000009'
MQCFT_TRACE_ROUTE	10	X'0000000A'
MQCFT_REPORT	12	X'0000000C'
MQCFT_INTEGER_FILTER	13	X'0000000D'
MQCFT_STRING_FILTER	14	X'0000000E'
MQCFT_BYTE_STRING_FILTER	15	X'0000000F'
MQCFT_COMMAND_XR	16	X'00000010'
MQCFT_XR_MSG	17	X'00000011'
MQCFT_XR_ITEM	18	X'00000012'
MQCFT_XR_SUMMARY	19	X'00000013'
MQCFT_GROUP	20	X'00000014'
MQCFT_STATISTICS	21	X'00000015'
MQCFT_ACCOUNTING	22	X'00000016'
MQCFT_INTEGER64	23	X'00000017'
MQCFT_INTEGER64_LIST	25	X'00000019'

MQCFTYPE_ * (Typy prostředku CF ve formátu příkazu)

Tabulka 80. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFTYPE_APPL	0	X'00000000'
MQCFTYPE_ADMIN	1	X'00000001'

MQCFUNC_ * (funkce záhlaví informací CICS)

Tabulka 81. Struktury konstant	
Název	**Struktura**
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_ŽÁDNÉ pole	'¬','¬','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQCGWI_ * (informační záhlaví CICS -interval čekání na získání)

Tabulka 82. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝCHOZÍ-MQCGWI_DEFAULT	-2	X'FFFFFFFE'

MQCHAD_ * (Automatická definice kanálu)

Tabulka 83. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHAD_DISABLED	0	X'00000000'
MQCHAD_ENABLED	1	X'00000001'

MQCHIDS_ * (formát příkazu Pochybný stav)

Tabulka 84. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHIDS_NOT_INDOUBT	0	X'00000000'
MQCHIDS_INDOUBT	1	X'00000001'

MQCHLD_ * (Dispozice kanálu ve formátu příkazu)

Tabulka 85. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHLD_ALL	-1	X'FFFFFFFF'
VÝCHOZÍ-MQCHLD_DEFAULT	1	X'00000001'

Tabulka 85. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHLD_SHARED	2	X'00000002'
MQCHLD_PRIVATE	4	X'00000004'
MQCHLD_FIXSHARED	5	X'00000005'

MQCHS_ * (Formát příkazu Stav kanálu)

Tabulka 86. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (Volby sdíleného restartování kanálu ve formátu příkazu)

Tabulka 87. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHSH_RESTART_NO	0	X'00000000'
MQCHSH_RESTART_YES	1	X'00000001'

MQCHSR_ * (Volby zastavení kanálu ve formátu příkazu)

Tabulka 88. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHSR_STOP_NOT_VÁMI požadováno	0	X'00000000'
MQCHSR_STOP_VÁMI požadováno	1	X'00000001'

MQCHSSTATE_ * (Podstavy kanálu ve formátu příkazu)

Tabulka 89. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHSSTATE_OTHER	0	X'00000000'
MQCHSSTATE_END_OF_BATCH	100	X'00000064'
MQCHSSTATE_SENDING	200	X'000000C8'
MQCHSSTATE_PŘIJÍMAJÍCÍ	300	X'0000012C'
MQCHSSTATE_SERIALIZING	400	X'00000190'
MQCHSSTATE_RESYNCHING	500	X'000001F4'

Tabulka 89. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHSSTATE_HEARTBEATING	600	X'00000258'
MQCHSSTATE_IN_SCYEXIT	700	X'000002BC'
MQCHSSTATE_IN_RCVEXIT	800	X'00000320'
MQCHSSTATE_IN_SENDEXIT	900	X'00000384'
MQCHSSTATE_IN_MSGEXIT	1000	X'000003E8'
MQCHSSTATE_IN_MREXIT	1100	X'0000044C'
MQCHSSTATE_IN_CHADEXIT	1200	X'000004B0'
MQCHSSTATE_NET_CONNECTING	1250	X'000004E2'
MQCHSSTATE_SSL_HANDSHAKING	1300	X'00000514'
MQCHSSTATE_NAME_SERVER	1400	X'00000578'
MQCHSSTATE_IN_MQPUT	1500	X'000005DC'
MQCHSSTATE_IN_MQGET	1600	X'00000640'
MQCHSSTATE_IN_MQI_CALL	1700	X'000006A4'
MQCHSSTATE_COMPRESSING	1800	X'00000708'

MQCHT_ * (typy kanálů)

Tabulka 90. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHT_SENDER-odesilatel	1	X'00000001'
MQCHT_SERVER	2	X'00000002'
MQCHT_RECEIVER	3	X'00000003'
MQCHT_REQUESTER	4	X'00000004'
MQCHT_ALL	5	X'00000005'
MQCHT_CLNTCONN	6	X'00000006'
MQCHT_SVRCONN	7	X'00000007'
MQCHT_CLUSRCVR	8	X'00000008'
MQCHT_CLUSSDR	9	X'00000009'

MQCHTAB_ * (Typy tabulek kanálů ve formátu příkazů)

Tabulka 91. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHTAB_Q_MGR	1	X'00000001'
MQCHTAB_CLNTCONN	2	X'00000002'

MQCI_ * (Identifikátor korelace)

Tabulka 92. Názvy konstant a hodnoty	
Název	**Hodnota**
MQCI_NONE	X'00...00' (24 nulových hodnot)
MQCI_NONE_ARRAY	'\0','\0',... (24 nulových hodnot)
MQCI_NEW_SESSION	X'414D5121...'

Tabulka 92. Názvy konstant a hodnoty (pokračování)	
Název	**Hodnota**
MQCI_NEW_SESSION_ARRAY	`'\x41','\x4D','\51','\x21',...`

MQCIH_ * (struktura a příznaky záhlaví informací CICS)

Struktura záhlaví informací CICS

Tabulka 93. Struktury konstant	
Název	**Struktura**
MQCIH_STRUC_ID	`"CIH¬"`
MQCIH_STRUC_ID_ARRAY	`'C','I','H','¬'`

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 94. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCIH_VERSION_1	1	X'00000001'
MQCIH_VERSION_2	2	X'00000002'
MQCIH_CURRENT_VERSION	2	X'00000002'
MQCIH_LENGTH_1	164	X'000000A4'
MQCIH_LENGTH_2	180	X'000000B4'
MQCIH_CURRENT_LENGTH	180	X'000000B4'

Příznaky záhlaví informací CICS

Tabulka 95. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCIH_NONE	0	X'00000000'
MQCIH_PASS_EXPIRATION	1	X'00000001'
MQCIH_UNLIMITED_EXPIRATION	0	X'00000000'
MQCIH_REPLY_WITHOUT_NULLS	2	X'00000002'
MQCIH_REPLY_WITH_NULLS	0	X'00000000'
MQCIH_SYNC_ON_RETURN	4	X'00000004'
MQCIH_NO_SYNC_ON_RETURN	0	X'00000000'

MQCLCT_ * (Typy mezipaměti klastru)

Tabulka 96. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLCT_STATIC	0	X'00000000'
MQCLCT_DYNAMIC	1	X'00000001'

MQCLRS_ * (Formát příkazu Vymazat rozsah řetězce tématu)

Tabulka 97. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLRS_LOCAL	1	X'00000001'
MQCLRS_GLOBAL	2	X'00000002'

MQCLRT_ * (Formát příkazu Vymazat typ řetězce tématu)

Tabulka 98. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLRT_ZACHOVÁNO	1	X'00000001'

MQCLT_ * (typy odkazů záhlaví informací CICS)

Tabulka 99. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLT_PROGRAM	1	X'00000001'
MQCLT_TRANSACTION	2	X'00000002'

MQCLWL_ * (pracovní zátěž klastru)

Tabulka 100. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLWL_USEQ_LOCAL	0	X'00000000'
MQCLWL_USEQ_ANY	1	X'00000001'
MQCLWL_USEQ_AS_Q_MGR	-3	X'FFFFFFFD'

MQCLXQ_ * (Typ přenosové fronty klastru)

MQCLXQ_* jsou hodnoty, které lze nastavit v atributu správce front DEFCLXQ . Atribut **DEFCLXQ** řídí, která přenosová fronta je standardně vybrána odesílacími kanály klastru pro získání zpráv, pro odeslání zpráv přijímacím kanálům klastru.

Tabulka 101. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLXQ_SCTQ	0	X'00000000'
MQCLXQ_CHANNEL	1	X'00000001'

Související odkazy

“DefClusterXmitQueue(MQLONG)” na stránce 817
Atribut DefClusterXmitQueue řídí, která přenosová fronta je standardně vybrána odesílacími kanály klastru pro získání zpráv, pro odeslání zpráv přijímacím kanálům klastru.

Změnit správce front

Zjistit správce front

Dotaz na správce front (odezva)

“MQINQ-Atributy dotazovaného objektu” na stránce 706
Volání MQINQ vrátí pole celých čísel a sadu znakových řetězců obsahujících atributy objektu.

MQCMD_ * (Kódy příkazů)

Tabulka 102. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCMD_NONE	0	X'00000000'
MQCMD_CHANGE_Q_MGR	1	X'00000001'
MQCMD_INQUIRE_Q_MGR	2	X'00000002'
MQCMD_CHANGE_PROCESS	3	X'00000003'
MQCMD_COPY_PROCESS	4	X'00000004'
MQCMD_CREATE_PROCESS	5	X'00000005'
MQCMD_DELETE_PROCESS	6	X'00000006'
MQCMD_INQUIRE_PROCESS	7	X'00000007'
MQCMD_CHANGE_Q	8	X'00000008'
MQCMD_CLEAR_Q	9	X'00000009'
MQCMD_COPY_Q	10	X'0000000A'
MQCMD_CREATE_Q	11	X'0000000B'
MQCMD_DELETE_Q	12	X'0000000C'
MQCMD_INQUIRE_Q	13	X'0000000D'
MQCMD_REFRESH_Q_MGR	16	X'00000010'
MQCMD_RESET_Q_STATS	17	X'00000011'
MQCMD_INQUIRE_Q_NAMES	18	X'00000012'
MQCMD_INQUIRE_PROCESS_NAMES	19	X'00000013'
MQCMD_INQUIRE_CHANNEL_NAMES	20	X'00000014'
MQCMD_CHANGE_CHANNEL	21	X'00000015'
MQCMD_COPY_CHANNEL	22	X'00000016'
MQCMD_CREATE_CHANNEL	23	X'00000017'
MQCMD_DELETE_CHANNEL	24	X'00000018'
MQCMD_INQUIRE_CHANNEL	25	X'00000019'
MQCMD_PING_CHANNEL	26	X'0000001A'
MQCMD_RESET_CHANNEL	27	X'0000001B'
MQCMD_START_CHANNEL	28	X'0000001C'
MQCMD_STOP_CHANNEL	29	X'0000001D'
MQCMD_START_CHANNEL_INIT	30	X'0000001E'
MQCMD_START_CHANNEL_LISTENER	31	X'0000001F'
MQCMD_CHANGE_NAMELIST	32	X'00000020'
MQCMD_COPY_NAMELIST	33	X'00000021'
MQCMD_CREATE_NAMELIST	34	X'00000022'
MQCMD_DELETE_NAMELIST	35	X'00000023'
MQCMD_INQUIRE_NAMELIST	36	X'00000024'
MQCMD_INQUIRE_NAMELIST_NAMES	37	X'00000025'
MQCMD_ESCAPE	38	X'00000026'
MQCMD_RESOLVE_CHANNEL	39	X'00000027'

Tabulka 102. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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_PUBLIKACE	60	X'0000003C'
MQCMD_DEREGISTER_PUBLISHER	61	X'0000003D'
MQCMD_DEREGISTER_ODBĚRATEL	62	X'0000003E'
MQCMD_PUBLISH	63	X'0000003F'
MQCMD_REGISTER_PUBLISHER.	64	X'00000040'
MQCMD_REGISTER_ODBĚRATEL	65	X'00000041'
MQCMD_REQUEST_UPDATE	66	X'00000042'
MQCMD_BROKER_INTERNAL	67	X'00000043'
MQCMD_ACTIVITY_MSG	69	X'00000045'
MQCMD_INQUIRE_CLUSTER_Q_MGR	70	X'00000046'
MQCMD_RESUME_Q_MGR_CLUSTER	71	X'00000047'
MQCMD_SUSPEND_Q_MGR_CLUSTER	72	X'00000048'
MQCMD_REFRESH_CLUSTER	73	X'00000049'
MQCMD_RESET_CLUSTER	74	X'0000004A'
MQCMD_TRACE_ROUTE	75	X'0000004B'
MQCMD_REFRESH_SECURITY	78	X'0000004E'
MQCMD_CHANGE_AUTH_INFO	79	X'0000004F'
MQCMD_COPY_AUTH_INFO	80	X'00000050'
MQCMD_CREATE_AUTH_INFO	81	X'00000051'
MQCMD_DELETE_AUTH_INFO	82	X'00000052'
MQCMD_INQUIRE_AUTH_INFO	83	X'00000053'
MQCMD_INQUIRE_AUTH_INFO_NAMES	84	X'00000054'
MQCMD_INQUIRE_CONNECTION	85	X'00000055'
MQCMD_STOP_CONNECTION	86	X'00000056'
MQCMD_INQUIRE_AUTH_RECS	87	X'00000057'
MQCMD_INQUIRE_ENTITY_AUTH	88	X'00000058'
MQCMD_DELETE_AUTH_REC	89	X'00000059'
MQCMD_SET_AUTH_REC-nastavení	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'

Tabulka 102. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
Událost MQCMD_COMMAND_EVENT	99	X'00000063'
MQCMD_CHANGE_SECURITY	100	X'00000064'
MQCMD_CHANGE_CF_STRUC	101	X'00000065'
MQCMD_CHANGE_STG_CLASS	102	X'00000066'
Trasování MQCMD_CHANGE_TRACE	103	X'00000067'
MQCMD_ARCHIVE_LOG	104	X'00000068'
MQCMD_BACKUP_CF_STRUC	105	X'00000069'
MQCMD_CREATE_BUFFER_POOL	106	X'0000006A'
MQCMD_CREATE_PAGE_SET	107	X'0000006B'
MQCMD_CREATE_CF_STRUC	108	X'0000006C'
MQCMD_CREATE_STG_CLASS	109	X'0000006D'
MQCMD_COPY_CF_STRUC	110	X'0000006E'
MQCMD_COPY_STG_CLASS	111	X'0000006F'
MQCMD_DELETE_CF_STRUC	112	X'00000070'
MQCMD_DELETE_STG_CLASS	113	X'00000071'
MQCMD_INQUIRE_ARCHIVE	114	X'00000072'
MQCMD_INQUIRE_CF_STRUC	115	X'00000073'
MQCMD_INQUIRE_CF_STRUC_STATUS	116	X'00000074'
MQCMD_INQUIRE_CMD_SERVER	117	X'00000075'
MQCMD_INQUIRE_CHANNEL_INIT	118	X'00000076'
MQCMD_INQUIRE_QSG	119	X'00000077'
MQCMD_INQUIRE_LOG	120	X'00000078'
MQCMD_INQUIRE_SECURITY	121	X'00000079'
MQCMD_INQUIRE_STG_CLASS	122	X'0000007A'
MQCMD_INQUIRE_SYSTEM	123	X'0000007B'
MQCMD_INQUIRE_THREAD	124	X'0000007C'
MQCMD_INQUIRE_TRACE	125	X'0000007D'
MQCMD_INQUIRE_USAGE	126	X'0000007E'
MQCMD_MOVE_Q	127	X'0000007F'
MQCMD_RECOVER_BSDS	128	X'00000080'
MQCMD_RECOVER_CF_STRUC	129	X'00000081'
MQCMD_RESET_TPIPE	130	X'00000082'
MQCMD_RESOLVE_INDOUBT	131	X'00000083'
MQCMD_RESUME_Q_MGR	132	X'00000084'
MQCMD_REVERIFY_SECURITY	133	X'00000085'

Tabulka 102. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCMD_SET_ARCHIVE	134	X'00000086'
MQCMD_SET_LOG	136	X'00000088'
MQCMD_SET_SYSTEM	137	X'00000089'
MQCMD_START_CMD_SERVER	138	X'0000008A'
MQCMD_START_Q_MGR	139	X'0000008B'
MQCMD_START_TRACE	140	X'0000008C'
MQCMD_STOP_CHANNEL_INIT	141	X'0000008D'
MQCMD_STOP_CHANNEL_LISTENER	142	X'0000008E'
MQCMD_STOP_CMD_SERVER	143	X'0000008F'
MQCMD_STOP_Q_MGR	144	X'00000090'
MQCMD_STOP_TRACE	145	X'00000091'
MQCMD_SUSPEND_Q_MGR	146	X'00000092'
MQCMD_INQUIRE_CF_STRUC_NAMES	147	X'00000093'
MQCMD_INQUIRE_STG_CLASS_NAMES	148	X'00000094'
MQCMD_CHANGE_SERVICE	149	X'00000095'
MQCMD_COPY_SERVICE	150	X'00000096'
MQCMD_CREATE_SERVICE	151	X'00000097'
MQCMD_DELETE_SERVICE	152	X'00000098'
MQCMD_INQUIRE_SERVICE	153	X'00000099'
MQCMD_INQUIRE_SERVICE_STATUS	154	X'0000009A'
MQCMD_START_SERVICE	155	X'0000009B'
MQCMD_STOP_SERVICE	156	X'0000009C'
MQCMD_DELETE_BUFFER_POOL	157	X'0000009D'
MQCMD_DELETE_PAGE_SET	158	X'0000009E'
FOND vyrovnávacích pamětí 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'

Tabulka 102. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMD_INQUIRE_TOPIC_NAMES	175	X'000000AF'
MQCMD_INQUIRE_SUBSCRIPTION	176	X'000000B0'
MQCMD_CREATE_SUBSCRIPTION	177	X'000000B1'
Odběr MQCMD_CHANGE_SUBSCRIPTION	178	X'000000B2'
MQCMD_DELETE_SUBSCRIPTION	179	X'000000B3'
MQCMD_COPY_SUBSCRIPTION	181	X'000000B5'
MQCMD_INQUIRE_SUB_STATUS	182	X'000000B6'
MQCMD_INQUIRE_TOPIC_STATUS	183	X'000000B7'
MQCMD_CLEAR_TOPIC_STRING	184	X'000000B8'
MQCMD_INQUIRE_PUBSUB_STATUS	185	X'000000B9'
MQCMD_PURGE_CHANNEL	195	X'000000C3'

MQCMDI_ * (Hodnoty informací o příkazu ve formátu příkazu)

Tabulka 103. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
Příkaz MQCMDI_COMMAND_ACCEPTED	5	X'00000005'
MQCMDI_CLUSTER_REQUEST_QUEUED	6	X'00000006'
MQCMDI_CHANNEL_INIT_STARTED	7	X'00000007'
MQCMDI_RECOVER_STARTED	11	X'0000000B'
MQCMDI_BACKUP_STARTED	12	X'0000000C'
MQCMDI_RECOVER_COMPLETED	13	X'0000000D'
MQCMDI_SEC_TIMER_ZERO	14	X'0000000E'
MQCMDI_REFRESH_CONFIGURATION	16	X'00000010'
MQCMDI_SEC_SIGNOFF_ERROR	17	X'00000011'
MQCMDI_IMS_BRIDGE_POZASTAVENO	18	X'00000012'
MQCMDI_DB2_SUSPENDED	19	X'00000013'
MQCMDI_DB2_OBSOLETE_MSGS	20	X'00000014'
MQCMDI_SEC_UPPERCASE, velká písmena	21	X'00000015'
MQCMDI_SEC_MIXEDCASE	22	X'00000016'

MQCMDL_ * (Úrovně příkazů)

Tabulka 104. Názvy konstant a hodnoty	
Název	**Hodnota**
MQCMDL_LEVEL_800	800
MQCMDL_LEVEL_801	801
MQCMDL_LEVEL_802	802

Tabulka 104. Názvy konstant a hodnoty (pokračování)	
Název	**Hodnota**
MQCMDL_LEVEL_900	900
MQCMDL_LEVEL_901	901
MQCMDL_LEVEL_902	902
MQCMDL_LEVEL_903	903
MQCMDL_LEVEL_904	904
MQCMDL_LEVEL_905	905
MQCMDL_LEVEL_910	910
MQCMDL_LEVEL_912	912
MQCMDL_LEVEL_913	913
MQCMDL_LEVEL_914	914
MQCMDL_LEVEL_915	915
MQCMDL_LEVEL_920	920
MQCMDL_LEVEL_921	921
MQCMDL_LEVEL_922	922
MQCMDL_LEVEL_923	923
MQCMDL_LEVEL_924	924
MQCMDL_LEVEL_925	925
MQCMDL_LEVEL_930	930
MQCMDL_LEVEL_931	931
MQCMDL_LEVEL_932	932

MQCMHO_ * (Vytvořit volby a strukturu popisovače zprávy)

Vytvořit strukturu voleb popisovače zprávy

Tabulka 105. Struktury konstant	
Název	**Struktura**
MQCMHO_STRUC_ID	"CMHO"
MQCMHO_STRUC_ID_ARRAY	'C','M','H','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 106. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMHO_VERSION_1	1	X'00000001'
MQCMHO_CURRENT_VERSION	1	X'00000001'

Volby pro vytvoření popisovače zprávy

Tabulka 107. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMHO_DEFAULT_VALIDATION	0	X'00000000'
MQCMHO_NO_VALIDATION	1	X'00000001'

Tabulka 107. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMHO_VALIDATE	2	X'00000002'
MQCMHO_NONE	0	X'00000000'

MQCNO_ * (Volby a struktura připojení)

Struktura voleb připojení

Tabulka 108. Struktury konstant	
Název	**Struktura**
MQCNO_STRUC_ID	"CNO¬"
MQCNO_STRUC_ID_ARRAY	'C','N','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 109. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCNO_VERSION_1	1	X'00000001'
MQCNO_VERSION_2	2	X'00000002'
MQCNO_VERSION_3	3	X'00000003'
MQCNO_VERSION_4	4	X'00000004'
MQCNO_VERSION_5	5	X'00000005'
MQCNO_CURRENT_VERSION	5	X'00000005'

Volby připojení

Tabulka 110. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCNO_STANDARDNÍ_VAZBA	0	X'00000000'
MQCNO_FASTPATH_BINDING	1	X'00000001'
MQCNO_SERIALIZE_CONN_TAG_Q_MGR	2	X'00000002'
MQCNO_SERIALIZE_CONN_TAG_QSG	4	X'00000004'
MQCNO_RESTRICT_CONN_TAG_Q_MGR	8	X'00000008'
MQCNO_RESTRICT_CONN_TAG_QSG	16	X'00000010'
MQCNO_HANDLE_SHARE_NONE	32	X'00000020'
MQCNO_HANDLE_SHARE_BLOCK	64	X'00000040'
MQCNO_HANDLE_SHARE_NO_BLOCK	128	X'00000080'
MQCNO_SHARED_BINDING	256	X'00000100'
MQCNO_ISOLATED_BINDING	512	X'00000200'
MQCNO_LOCAL_BINDING	1024	X'00000400'
MQCNO_CLIENT_BINDING	2048	X'00000800'
MQCNO_ACCOUNTING_MQI_ENABLED	4096	X'00001000'
MQCNO_ACCOUNTING_MQI_DISABLED	8192	X'00002000'
MQCNO_ACCOUNTING_Q_ENABLED	16384	X'00004000'

Tabulka 110. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCNO_ACCOUNTING_Q_DISABLED	32768	X'00008000'
MQCNO_NO_CONV_SHARING	65536	X'00010000'
MQCNO_ALL_CONVS_SHARE	262144	X'00040000'
MQCNO_CD_FOR_OUTPUT_ONLY	524288	X'00080000'
MQCNO_USE_CD_SELECTION	1048576	X'00100000'
MQCNO_RECONNECT	16777216	X'01000000'
MQCNO_RECONNECT_AS_DEF	0	X'00000000'
MQCNO_RECONNECT_DISABLED	33554432	X'02000000'
MQCNO_RECONNECT_Q_MGR	67108864	X'04000000'
MQCNO_ACTIVITY_TRACE_ENABLED	134217728	X'08000000'
MQCNO_ACTIVITY_TRACE_DISABLED	268435456	X'10000000'
MQCNO_NONE	0	X'00000000'

MQCO_ * (Volby zavření)

Tabulka 111. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCO_IMMEDIATE	0	X'00000000'
MQCO_NONE	0	X'00000000'
MQCO_DELETE	1	X'00000001'
MQCO_DELETE_PURGE	2	X'00000002'
MQCO_KEEP_SUB	4	X'00000004'
MQCO_REMOVE_SUB	8	X'00000008'
MQCO_QUIESCE	32	X'00000020'

MQCODL_ * (délka výstupních dat záhlaví informací CICS)

Tabulka 112. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCODL_AS_INPUT	-1	X'FFFFFFFF'

MQCOMPRESS_ * (komprese kanálů)

Tabulka 113. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCOMPRESS_NOT_AVAILABLE	-1	X'FFFFFFFF'
MQCOMPRESS_NONE	0	X'00000000'
MQCOMPRESS_RLE	1	X'00000001'
MQCOMPRESS_ZLIBFAST	2	X'00000002'
MQCOMPRESS_ZLIBHIGH	4	X'00000004'
MQCOMPRESS_SYSTEM	8	X'00000008'
MQCOMPRESS_ANY	268435455	X'0FFFFFFF'

MQCONNID_ * (Identifikátor připojení)

Tabulka 114. Názvy konstant a hodnoty

Název	Hodnota
MQCONNID_NONE	`X'00...00'` (24 nulových hodnot)
MQCONNID_NONE_ARRAY	`'\0','\0',...` (24 nulových hodnot)

MQCOPY_ * (Volby kopírování vlastností)

Tabulka 115. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCOPY_NONE	0	X'00000000'
MQCOPY_ALL	1	X'00000001'
MQCOPY_FORWARD	2	X'00000002'
MQCOPY_PUBLISH	4	X'00000004'
MQCOPY_REPLY	8	X'00000008'
MQCOPY_REPORT	16	X'00000010'
MQCOPY_DEFAULT	22	X'00000016'

MQCQT_ * (Typy front klastru)

Tabulka 116. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
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_ * (návratové kódy záhlaví informací CICS)

Tabulka 117. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCRC_OK	0	X'00000000'
MQCRC_CICS_EXEC_ERROR	1	X'00000001'
MQCRC_MQ_API_ERROR	2	X'00000002'
MQCRC_BRIDGE_ERROR	3	X'00000003'
MQCRC_BRIDGE_ABEND	4	X'00000004'
MQCRC_APPLICATION_ABEND	5	X'00000005'
MQCRC_SECURITY_ERROR	6	X'00000006'
MQCRC_PROGRAM_NENÍ k dispozici	7	X'00000007'
MQCRC_BRIDGE_TIMEOUT	8	X'00000008'
MQCRC_TRANSID_NOT_AVAILABLE	9	X'00000009'

MQCS_ * (stav spotřebitele konstant MQCBC)

Tabulka 118. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCS_NONE	0	X'00000000'
MQCS_SUSPENDED_TEMPORARY	1	X'00000001'
MQCS_SUSPENDED_USER_ACTION	2	X'00000002'
MQCS_POZASTAVENO	3	X'00000003'
MQCS_STOPPED	4	X'00000004'

MQCSC_ * (počáteční kódy záhlaví informací CICS)

Tabulka 119. Struktury konstant	
Název	**Struktura**
MQCSC_START	"S¬¬¬"
MQCSC_STARTDATA	"SD¬¬"
MQCSC_TERMINPUT	"TD¬¬"
MQCSC_NONE	"¬¬¬¬"
MQCSC_START_ARRAY	'S','¬','¬','¬'
MQCSC_STARTDATA_ARRAY	'S','D','¬','¬'
MQCSC_TERMINPUT_ARRAY	'T','D','¬','¬'
MQCSC_NONE_ARRAY	'¬','¬','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQCSP_ * (struktura parametrů zabezpečení připojení a typy ověřování)

Struktura parametrů zabezpečení připojení

Tabulka 120. Struktury konstant	
Název	**Struktura**
MQCSP_STRUC_ID	"CSP¬"
MQCSP_STRUC_ID_ARRAY	'C','S','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 121. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCSP_VERSION_1	1	X'00000001'
MQCSP_VERSION_2	2	X'00000002'
MQCSP_VERSION_3	3	X'00000003'
MQCSP_CURRENT_VERSION	3	X'00000003'

Parametry zabezpečení připojení Typy ověřování

Tabulka 122. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCSP_AUTH_NONE	0	X'00000000'
MQCSP_AUTH_USER_ID_AND_PWD	1	X'00000001'
MQCSP_AUTH_ID_TOKEN	2	X'00000002'

MQCSRV_ * (Volby příkazového serveru)

Tabulka 123. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (Značka připojení správce front)

Tabulka 124. Názvy konstant a hodnoty	
Název	**Hodnota**
MQCT_NONE	X'00...00' (128 nulových hodnot)
MQCT_NONE_ARRAY	'\0','\0',... (128 nulových hodnot)

MQCTES_ * (Stav ukončení úlohy záhlaví informací CICS)

Tabulka 125. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCTES_NOSYNC	0	X'00000000'
MQCTES_COMMIT	256	X'00000100'
MQCTES_BACKOUT	4352	X'00001100'
MQCTES_ENDTASK	65536	X'00010000'

MQCTLO_ * (struktura voleb MQCTL a volby řízení spotřebitelů)

Struktura voleb MQCTL

Tabulka 126. Struktury konstant	
Název	**Struktura**
MQCTLO_STRUC_ID	"CTLO"
MQCTLO_STRUC_ID_ARRAY	'C','T','L','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 127. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCTLO_VERSION_1	1	X'00000001'
MQCTLO_CURRENT_VERSION	1	X'00000001'

Volby řízení spotřebitele MQCTL

Tabulka 128. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCTLO_NONE	0	X'00000000'
MQCTLO_THREAD_AFFINITY	1	X'00000001'
MQCTLO_FAIL_IF_QUIESCING	8192	X'00002000'

MQCUOWC_ * (záhlaví informací CICS Řízení jednotky práce)

Tabulka 129. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCUOWC_ONLY	273	X'00000111'
MQCUOWC_CONTINUE	65536	X'00010000'
MQCUOWC_FIRST	17	X'00000011'
MQCUOWC_MIDDLE	16	X'00000010'
MQCUOWC_LAST	272	X'00000110'
MQCUOWC_COMMIT	256	X'00000100'
MQCUOWC_BACKOUT	4352	X'00001100'

MQCXP_ * (struktura parametrů uživatelské procedury kanálu)

Tabulka 130. Struktury konstant

Název	Struktura
MQCXP_STRUC_ID	"CXP¬"
MQCXP_STRUC_ID_ARRAY	'C','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 131. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCXP_VERSION_1	1	X'00000001'
MQCXP_VERSION_2	2	X'00000002'
MQCXP_VERSION_3	3	X'00000003'
MQCXP_VERSION_4	4	X'00000004'
MQCXP_VERSION_5	5	X'00000005'
MQCXP_VERSION_6	6	X'00000006'
MQCXP_VERSION_7	7	X'00000007'
MQCXP_VERSION_8	8	X'00000008'
MQCXP_VERSION_9	9	X'00000009'
MQCXP_CURRENT_VERSION	9	X'00000009'

MQDC_ * (Cílová třída)

Tabulka 132. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQDC_MANAGED	1	X'00000001'

Tabulka 132. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDC_PROVIDED	2	X'00000002'

MQDCC_ * (Volby převodu a masky a faktory)

Volby převodu

Tabulka 133. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDCC_DEFAULT_CONVERSION	1	X'00000001'
MQDCC_FILL_TARGET_BUFFER	2	X'00000002'
MQDCC_INT_DEFAULT_CONVERSION	4	X'00000004'
MQDCC_SOURCE_ENC_NATIVE	(value differs by platform or version)	(value differs by platform or version)
MQDCC_SOURCE_ENC_NORMAL	16	X'00000010'
MQDCC_SOURCE_ENC_REVERSED	32	X'00000020'
MQDCC_SOURCE_ENC_UNDEFINED	0	X'00000000'
MQDCC_TARGET_ENC_NATIVE	(value differs by platform or version)	(value differs by platform or version)
MQDCC_TARGET_ENC_NORMAL	256	X'00000100'
MQDCC_TARGET_ENC_REVERSED	512	X'00000200'
MQDCC_TARGET_ENC_UNDEFINED	0	X'00000000'
MQDCC_NONE	0	X'00000000'

Masky a faktory voleb převodu

Tabulka 134. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDCC_SOURCE_ENC_MASK	240	X'000000F0'
MQDCC_TARGET_ENC_MASK	3840	X'00000F00'
MQDCC_SOURCE_ENC_FACTOR	16	X'00000010'
MQDCC_TARGET_ENC_FACTOR	256	X'00000100'

MQDELO_ * (Volby odstranění publikování/odběru)

Tabulka 135. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDELO_NONE	0	X'00000000'
MQDELO_LOCAL	4	X'00000004'

MQDH_ * (Struktura záhlaví distribuce)

Tabulka 136. Struktury konstant	
Název	**Struktura**
MQDH_STRUC_ID	"DH¬¬"

Tabulka 136. Struktury konstant (pokračování)	
Název	**Struktura**
MQDH_STRUC_ID_POLE	'D','H','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 137. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDH_VERSION_1	1	X'00000001'
MQDH_CURRENT_VERSION	1	X'00000001'

MQDHF_ * (příznaky záhlaví distribuce)

Tabulka 138. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDHF_NEW_MSG_IDS	1	X'00000001'
MQDHF_NONE	0	X'00000000'

MQDISCONNECT_ * (Formát příkazu Typy odpojení)

Tabulka 139. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDISCONNECT_NORMAL	0	X'00000000'
MQDISCONNECT_IMPLICIT	1	X'00000001'
MQDISCONNECT_Q_MGR	2	X'00000002'

MQDL_ * (distribuční seznamy)

Tabulka 140. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDL_PODPOROVÁNO	1	X'00000001'
MQDL_NOT_SUPPORTED	0	X'00000000'

MQDLH_ * (struktura záhlaví nedoručených zpráv)

Tabulka 141. Struktury konstant	
Název	**Struktura**
ID_STRUC_MQDLH_STRUC_ID	"DLH¬"
MQDLH_STRUC_ID_ARRAY	'D','L','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

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

MQDLV_ * (trvalé/dočasné doručení zprávy)

Tabulka 142. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDLV_AS_PARENT	0	X'00000000'
MQDLV_ALL	1	X'00000001'
MQDLV_ALL_DUR	2	X'00000002'
MQDLV_ALL_AVAIL	3	X'00000003'

MQDMHO_ * (Odstranit volby a strukturu popisovače zprávy)

Odstranit strukturu voleb popisovače zprávy

Tabulka 143. Struktury konstant	
Název	**Struktura**
MQDMHO_STRUC_ID	"DMHO"
MQDMHO_STRUC_ID_ARRAY	'D','M','H','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 144. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDMHO_VERSION_1	1	X'00000001'
MQDMHO_CURRENT_VERSION	1	X'00000001'

Volby odstranění popisovače zprávy

Tabulka 145. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDMHO_NONE	0	X'00000000'

MQDMPO_ * (Odstranění voleb a struktury vlastností zprávy)

Odstranit strukturu voleb vlastností zprávy

Tabulka 146. Struktury konstant	
Název	**Struktura**
ID_STRUC_MQDMPO_ID	"DMPO"
MQDMPO_STRUC_ID_ARRAY	'D','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 147. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDMPO_VERSION_1	1	X'00000001'
MQDMPO_CURRENT_VERSION	1	X'00000001'

Volby odstranění vlastnosti zprávy

Tabulka 148. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDMPO_DEL_FIRST	0	X'00000000'
MQDMPO_DEL_PROP_UNDER_CURSOR	1	X'00000001'
MQDMPO_NONE	0	X'00000000'

MQDNSWLM_ * (WLM DNS)

Tabulka 149. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDNSWLM_NO	0	X'00000000'
MQDNSWLM_YES	1	X'00000001'

MQDT_ * (Typy cílů)

Tabulka 150. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDT_APPL	1	X'00000001'
MQDT_BROKER	2	X'00000002'

MQDXP_ * (struktura parametrů uživatelské procedury převodu)

Tabulka 151. Struktury konstant	
Název	**Struktura**
MQDXP_STRUC_ID	"DXP¬"
MQDXP_STRUC_ID_ARRAY	'D','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 152. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDXP_VERSION_1	1	X'00000001'
MQDXP_VERSION_2	2	X'00000002'
MQDXP_CURRENT_VERSION	2	X'00000002'

MQEC_ * (Hodnoty signálu)

Tabulka 153. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEC_MSG_ARRXX_ENCODE_CASE_ONE tabulka	2	X'00000002'
MQEC_WAIT_INTERVAL_VYPRŠELA	3	X'00000003'
MQEC_WAIT_ZRUŠENO	4	X'00000004'
MQEC_Q_MGR_QUIESCING	5	X'00000005'
MQEC_CONNECTION_QUIESCING	6	X'00000006'

MQEI_ * (vypršení platnosti)

Tabulka 154. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQEI_UNLIMITED	-1	X'FFFFFFFF'

MQENC_ * (kódování)

MQENC_ * (kódování)

Tabulka 155. Hodnoty konstant podle platformy

Název	Platforma	Desetinná hodnota	Hexadecimální hodnota
MQENC_NATIVE	IBM i	273	X'00000111'
	Linux	546	X'00000222'
	Linux na serveru SPARC	273	X'00000111'
	Linux na platformě x86	546	X'00000222'
	AIX and Linux	273	X'00000111'
	Windows	546	X'00000222'
	Mikro fokus v jazyku COBOL na systému Windows	17	X'00000011'
	z/OS	785	X'00000311'

MQENC_ * (Kódovací masky)

Tabulka 156. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MASKA MQENC_INTEGER_MASK	15	X'0000000F'
MQENC_DECIMAL_MASK	240	X'000000F0'
MQENC_FLOAT_MASK	3840	X'00000F00'
MQENC_RESERVED_MASK	-4096	X'FFFFF000'

MQENC_ * (kódování pro binární celá čísla)

Tabulka 157. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQENC_INTEGER_UNDEFINED	0	X'00000000'
MQENC_INTEGER_NORMAL	1	X'00000001'
MQENC_INTEGER_REVERSED	2	X'00000002'

MQENC_ * (kódování pro pakovaná desetinná celá čísla)

Tabulka 158. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQENC_DECIMAL_UNDEFINED	0	X'00000000'
MQENC_DECIMAL_NORMAL	16	X'00000010'
MQENC_DECIMAL_REVERSED	32	X'00000020'

MQENC_ * (kódování pro čísla s pohyblivou řádovou čárkou)

Tabulka 159. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQENC_FLOAT_UNDEFINED	0	X'00000000'
MQENC_FLOAT_IEEE_NORMAL	256	X'00000100'
MQENC_FLOAT_IEEE_REVERSED	512	X'00000200'
MQENC_FLOAT_S390	768	X'00000300'
MQENC_FLOAT_TNS	1024	X'00000400'

MQEPH_ * (Struktura záhlaví a příznaky vloženého formátu příkazu)

Struktura záhlaví vloženého formátu příkazu

Tabulka 160. Struktury konstant	
Název	**Struktura**
MQEPH_STRUC_ID	"EPH¬"
MQEPH_STRUC_ID_POLE	'E','P','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 161. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEPH_STRUC_LENGTH_FIXED	68	X'00000044'
MQEPH_VERSION_1	1	X'00000001'
MQEPH_CURRENT_VERSION	1	X'00000001'

Příznaky záhlaví vloženého formátu příkazu

Tabulka 162. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEPH_NONE	0	X'00000000'
MQEPH_CCSID_EMBEDDED	1	X'00000001'

MQET_ * (Typ změny významu ve formátu příkazu)

Tabulka 163. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQET_MQSC	1	X'00000001'

MQEVO_ * (výchozí body událostí ve formátu příkazu)

Tabulka 164. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEVO_OTHER	0	X'00000000'
MQEVO_CONSOLE	1	X'00000001'
MQEVO_INIT	2	X'00000002'
MQEVO_MSG	3	X'00000003'

Tabulka 164. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEVO_MQSET	4	X'00000004'
MQEVO_INTERNÍ	5	X'00000005'
MQEVO_MQSUB	6	X'00000006'
MQEVO_CTLMSG	7	X'00000007'
MQEVO_REST	8	X'00000008'

MQEVR_ * (Záznam událostí ve formátu příkazu)

Tabulka 165. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEVR_DISABLED	0	X'00000000'
MQEVR_ENABLED	1	X'00000001'
VÝJIMKA-MQEVR_EXCEPTION	2	X'00000002'
MQEVR_NO_DISPLAY	3	X'00000003'

MQEXPI_ * (Interval skenování vypršení platnosti)

Tabulka 166. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEXPI_OFF	0	X'00000000'

MQFB_ * (Hodnoty zpětné vazby)

Tabulka 167. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQFB_NONE	0	X'00000000'
MQFB_SYSTEM_FIRST	1	X'00000001'
MQFB_QUIT	256	X'00000100'
MQFB_EXPIRATION	258	X'00000102'
MQFB_COA	259	X'00000103'
MQFB_COD	260	X'00000104'
MQFB_CHANNEL_DOKONČENO	262	X'00000106'
MQFB_CHANNEL_FAIL_RETRY	263	X'00000107'
MQFB_CHANNEL_FAIL-selhání	264	X'00000108'
MQFB_APPL_CANNOT_BE_STARTED	265	X'00000109'
MQFB_TM_ERROR	266	X'0000010A'
MQFB_APPL_TYPE_ERROR	267	X'0000010B'
MQFB_STOPPED_BY_MSG_EXIT	268	X'0000010C'
MQFB_ACTIVITY	269	X'0000010D'
MQFB_XMIT_Q_MSG_ERROR	271	X'0000010F'
MQFB_PAN	275	X'00000113'
MQFB_NAN	276	X'00000114'
MQFB_STOPPED_BY_CHAD_EXIT	277	X'00000115'

Tabulka 167. Hodnoty konstant (pokračování)

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

Tabulka 167. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQFB_APPL_LAST	999999999	X'3B9AC9FF'

MQFC_ * (Volby vynucení formátu příkazu)

Tabulka 168. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQFC_YES	1	X'00000001'
MQFC_NO	0	X'00000000'

MQFMT_ * (Formáty)

Tabulka 169. Názvy konstant a hodnoty	
Název	**Hodnota**
MQFMT_NONE	"¬¬¬¬¬¬¬¬"
MQFMT_ADMIN	"MQADMIN¬"
MQFMT_CHANNEL_COMPLETED	"MQCHCOM¬"
MQFMT_CICS:	"MQCICS¬¬"
MQFMT_COMMAND_1	"MQCMD1¬¬"
MQFMT_COMMAND_2	"MQCMD2¬¬"
MQFMT_DEAD_LETTER_HEADER	"MQDEAD¬¬"
MQFMT_DIST_HEADER	"MQHDIST¬"
MQFMT_EMBEDDED_PCF	"MQHEPCF¬"
MQFMT_EVENT	"MQEVENT¬"
MQFMT_IMS	"MQIMS¬¬¬"
MQFMT_IMS_VAR_STRING	"MQIMSVS¬"
MQFMT_MD_EXTENSION	"MQHMDE¬¬"
MQFMT_PCF	"MQPCF¬¬¬"
MQFMT_REF_MSG_HEADER	"MQHREF¬¬"
MQFMT_RF_HEADER	"MQHRF¬¬¬"
MQFMT_RF_HEADER_1	"MQHRF¬¬¬"
MQFMT_RF_HEADER_2	"MQHRF2¬¬"
MQFMT_STRING	"MQSTR¬¬¬"
MQFMT_TRIGGER	"MQTRIG¬¬"
MQFMT_WORK_INFO_HEADER	"MQHWIH¬¬"
MQFMT_XMIT_Q_HEADER	"MQXMIT¬¬"
MQFMT_NONE_ARRAY	'¬','¬','¬','¬','¬','¬','¬','¬'
MQFMT_ADMIN_ARRAY	'M','Q','A','D','M','I','N','¬'
MQFMT_CHANNEL_COMPLETED_ARRAY	'M','Q','C','H','C','O','M','¬'
MQFMT_CICS_ARRAY	'M','Q','C','I','C','S','¬','¬'
MQFMT_COMMAND_1_ARRAY	'M','Q','C','M','D','1','¬','¬'
MQFMT_COMMAND_2_ARRAY	'M','Q','C','M','D','2','¬','¬'

Tabulka 169. Názvy konstant a hodnoty (pokračování)	
Název	**Hodnota**
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','¬','¬'
Pole 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','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQFUN_ * (typy aplikačních funkcí)

Tabulka 170. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
Neznámý typ MQFUN_TYPE_UNKNOWN	0	X'00000000'
MQFUN_TYPE_JVM	1	X'00000001'
MQFUN_TYPE_PROGRAM	2	X'00000002'
MQFUN_TYPE_PROCEDURE	3	X'00000003'
MQFUN_TYPE_USERDEF	4	X'00000004'
MQFUN_TYPE_COMMAND	5	X'00000005'

MQGA_ * (Selektory atributů skupiny)

Tabulka 171. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQGA_FIRST	8001	X'00001F41'
MQGA_LAST	9000	X'00002328'

MQGACF_ * (Typy parametrů skupiny ve formátu příkazu)

Tabulka 172. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQGACF_FIRST	8001	X'00001F41'
MQGACF_COMMAND_CONTEXT	8001	X'00001F41'

Tabulka 172. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQGACF_COMMAND_DATA	8002	X'00001F42'
MQGACF_TRACE_ROUTE	8003	X'00001F43'
MQGACF_OPERATION	8004	X'00001F44'
MQGACF_ACTIVITY	8005	X'00001F45'
MQGACF_EMBEDDED_MQMD	8006	X'00001F46'
MQGACF_MESSAGE	8007	X'00001F47'
MQGACF_MQMD	8008	X'00001F48'
MQGACF_VALUE_NAMING	8009	X'00001F49'
MQGACF_Q_ACCOUNTING_DATA	8010	X'00001F4A'
MQGACF_Q_STATISTICS_DATA	8011	X'00001F4B'
MQGACF_CHL_STATISTICS_DATA	8012	X'00001F4C'
MQGACF_LAST_USED	8012	X'00001F4C'

MQGI_ * (identifikátor skupiny)

Tabulka 173. Názvy konstant a hodnoty	
Název	**Hodnota**
MQGI_NONE	X'00...00' (24 nulových hodnot)
MQGI_NONE_ARRAY	'\0','\0',... (24 nulových hodnot)

MQGMO_ * (Získat volby a strukturu zprávy)

Získat strukturu voleb zprávy

Tabulka 174. Struktury konstant	
Název	**Struktura**
MQGMO_STRUC_ID	"GMO¬"
MQGMO_STRUC_ID_ARRAY	'G','M','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 175. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQGMO_VERSION_1	1	X'00000001'
MQGMO_VERSION_2	2	X'00000002'
MQGMO_VERSION_3	3	X'00000003'
MQGMO_VERSION_4	4	X'00000004'
MQGMO_CURRENT_VERSION	4	X'00000004'

Volby získání zprávy

Tabulka 176. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQGMO_WAIT	1	X'00000001'

Tabulka 176. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
MQGMO_PROPERTIES_AS_Q_DEF	0	X'00000000'
MQGMO_NONE	0	X'00000000'

MQGS_ * (Stav skupiny)

Tabulka 177. Názvy konstant a hodnoty	
Název	**Hodnota**
MQGS_NOT_IN_GROUP	'¬'
MQGS_MSG_IN_GROUP	'G'
MQGS_LAST_MSG_IN_GROUP	'L'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQHA_ * (Selektory popisovačů)

Tabulka 178. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQHA_FIRST	4001	X'00000FA1'
MQHA_BAG_HANDLE	4001	X'00000FA1'
MQHA_LAST_USED	4001	X'00000FA1'
MQHA_LAST	6000	X'00001770'

MQHB_ * (úchyty na tašky)

Tabulka 179. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQHB_UNUSABLE_HBAG	-1	X'FFFFFFFF'
MQHB_NONE	-2	X'FFFFFFFE'

MQHC_ * (obslužné rutiny připojení)

Tabulka 180. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQHC_DEF_HCONN	0	X'00000000'
MQHC_UNUSABLE_HCONN	-1	X'FFFFFFFF'
MQHC_UNASSOCIATED_HCONN	-3	X'FFFFFFFD'

MQHM_ * (popisovač zprávy)

Tabulka 181. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQHM_UNUSABLE_HMSG	-1	X'FFFFFFFF'
MQHM_NONE	0	X'00000000'

MQHO_ * (popisovač objektu)

Tabulka 182. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQHO_UNUSABLE_HOBJ	-1	X'FFFFFFFF'
MQHO_NONE	0	X'00000000'

MQHSTATE_ * (Formát příkazu Státy popisovače)

Tabulka 183. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQHSTATE_INACTIVE	0	X'00000000'
MQHSTATE_ACTIVE	1	X'00000001'

MQIA_ * (Celočíselné selektory atributů)

Tabulka 184. Hodnoty konstant

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

Tabulka 184. Hodnoty konstant (pokračování)

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

Tabulka 184. Hodnoty konstant (pokračování)

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

Tabulka 184. Hodnoty konstant (pokračování)

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

Tabulka 184. Hodnoty konstant (pokračování)

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

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

MQIACF_ * (Formát příkazu Typy celočíselných parametrů)

Tabulka 185. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIACF_FIRST	1001	X'000003E9'
MQIACF_Q_MGR_ATTRS	1001	X'000003E9'
MQIACF_Q_ATTRS	1002	X'000003EA'
MQIACF_PROCESS_ATTRS	1003	X'000003EB'
MQIACF_NAMELIST_ATTRS	1004	X'000003EC'
MQIACF_FORCE	1005	X'000003ED'
MQIACF_REPLACE	1006	X'000003EE'
MQIACF_PURGE	1007	X'000003EF'
MQIACF_QUIESCE	1008	X'000003F0'
MQIACF_MODE	1008	X'000003F0'
MQIACF_ALL	1009	X'000003F1'
MQIACF_EVENT_APPL_TYPE	1010	X'000003F2'

Tabulka 185. Hodnoty konstant (pokračování)

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

Tabulka 185. Hodnoty konstant (pokračování)

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

Tabulka 185. Hodnoty konstant (pokračování)

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

Tabulka 185. Hodnoty konstant (pokračování)

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

Tabulka 185. Hodnoty konstant (pokračování)

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

Tabulka 185. Hodnoty konstant (pokračování)

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

Tabulka 185. Hodnoty konstant (pokračování)

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

Tabulka 185. Hodnoty konstant (pokračování)

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

Tabulka 185. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIACF_LOG_IN_USE	1420	X'0000058C'
MQIACF_LOG_UTILIZATION	1421	X'0000058D'
MQIACF_IGNORE_STATE	1423	X'0000058F'
MQIACF_MOVABLE_APPL_COUNT	1424	X'00000590'
MQIACF_APPL_INFO_ATTRS	1425	X'00000591'
MQIACF_APPL_MOVABLE	1426	X'00000592'
MQIACF_REMOTE_QMGR_ACTIVE	1427	X'00000593'
MQIACF_APPL_INFO_TYPE	1428	X'00000594'
MQIACF_APPL_INFO_APPL	1429	X'00000595'
MQIACF_APPL_INFO_QMGR	1430	X'00000596'
MQIACF_APPL_INFO_LOCAL	1431	X'00000597'
MQIACF_APPL_IMMOVABLE_COUNT	1432	X'00000598'
MQIACF_VYVÁŽENO	1433	X'00000599'
MQIACF_BALSTATE	1434	X'0000059A'
MQIACF_APPL_IMMOVABLE_REASON	1435	X'0000059B'
MQIACF_DS_ENCRYPTED	1436	X'0000059C'
MQIACF_CUR_Q_FILE_SIZE	1437	X'0000059D'
MQIACF_CUR_MAX_FILE_SIZE	1438	X'0000059E'
MQIACF_BALANCING_TYPE	1439	X'0000059F'
MQIACF_BALANCING_OPTIONS	1440	X'000005A0'
MQIACF_BALANCING_TIMEOUT	1441	X'000005A1'
MQIACF_SYSP_SMF_STAT_TIME_SECS	1442	X'000005A2'
MQIACF_SYSP_SMF_ACCT_TIME_MINS	1443	X'000005A3'
MQIACF_SYSP_SMF_ACCT_TIME_SECS	1444	X'000005A4'
V 9.3.0 V 9.3.0 MQIACF_LAST_USED	1444	X'000005A4'

MQIACH_ * (Formát příkazu Typy celočíselných kanálů)

Tabulka 186. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'

Tabulka 186. Hodnoty konstant (pokračování)

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

Tabulka 186. Hodnoty konstant (pokračování)

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

Tabulka 186. Hodnoty konstant (pokračování)

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

Tabulka 186. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIACH_CHANNEL_SUMMARY_ATTRS	1642	X'0000066A'
MQIACH_PROTOCOL	1643	X'0000066B'
MQIACH_AMQPKEEPALIVE	1644	X'0000066C'
MQIACH_SECURITY_PROTOCOL	1645	X'0000066D'
z/OS MQIACH_SPL_PROTECTION	1646	X'0000066E'
MQIACH_LAST_USED	1646	X'0000066E'

MQIAMO_ * (Typy parametrů monitorování Integer formátu příkazu)

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

Tabulka 187. Hodnoty konstant (pokračování)

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

Tabulka 187. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIAMO_SUB_NDUR_LOWWATER	778	X'0000030A'
MQIAMO_TOPIC_PUTS	779	X'0000030B'
MQIAMO_TOPIC_PUTS_FAILED	780	X'0000030C'
MQIAMO_TOPIC_PUT1S	781	X'0000030D'
MQIAMO_TOPIC_PUT1S_FAILED	782	X'0000030E'
MQIAMO_PUBLISH_MSG_COUNT	784	X'00000310'
MQIAMO_UNSUBS_DUR	786	X'00000312'
MQIAMO_UNSUBS_NDUR	787	X'00000313'
MQIAMO_UNSUBS_FAILED	788	X'00000314'
MQIAMO_INTERVAL	789	X'00000315'
MQIAMO_MSGS_SENT	790	X'00000316'
MQIAMO_BYTES_SENT	791	X'00000317'
MQIAMO_REPAIR_BYTES	792	X'00000318'
MQIAMO_FEEDBACK_MODE	793	X'00000319'
MQIAMO_RELIABILITY_TYPE	794	X'0000031A'
MQIAMO_LATE_JOIN_MARK	795	X'0000031B'
MQIAMO_NACKS_RCVD	796	X'0000031C'
MQIAMO_REPAIR_PKTS	797	X'0000031D'
MQIAMO_HISTORY_PKTS	798	X'0000031E'
MQIAMO_PENDING_PKTS	799	X'0000031F'
MQIAMO_PKT_RATE	800	X'00000320'
MQIAMO_MCAST_XMIT_RATE	801	X'00000321'
MQIAMO_MCAST_BATCH_TIME	802	X'00000322'
MQIAMO_MCAST_HEARTBEAT	803	X'00000323'
MQIAMO_DEST_DATA_PORT	804	X'00000324'
MQIAMO_DEST_REPAIR_PORT	805	X'00000325'
MQIAMO_ACKS_RCVD	806	X'00000326'
MQIAMO_ACTIVE_ACKERS	807	X'00000327'
MQIAMO_PKTS_SENT	808	X'00000328'
MQIAMO_TOTAL_REPAIR_PKTS	809	X'00000329'
MQIAMO_TOTAL_PKTS_SENT	810	X'0000032A'
MQIAMO_TOTAL_MSGS_SENT	811	X'0000032B'
MQIAMO_TOTAL_BYTES_SENT	812	X'0000032C'
MQIAMO_POČET_PROUDŮ	813	X'0000032D'
ZPĚTNÁ vazba 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'

Tabulka 187. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIAMO_MSGS_DELIVERED	819	X'00000333'
MQIAMO_PKTS_PROCESSED	820	X'00000334'
MQIAMO_PKTS_DLVD	821	X'00000335'
MQIAMO_PKTS_DROPPED	822	X'00000336'
MQIAMO_PKTS_DUPLICATED	823	X'00000337'
MQIAMO_NACKS_CREATED	824	X'00000338'
MQIAMO_NACK_PKTS_SENT	825	X'00000339'
MQIAMO_REPAIR_PKTS_RQSTD	826	X'0000033A'
MQIAMO_REPAIR_PKTS_RCVD	827	X'0000033B'
MQIAMO_PKTS_REPAIRED	828	X'0000033C'
MQIAMO_TOTAL_MSGS_RCVD	829	X'0000033D'
MQIAMO_TOTAL_MSGS_BYTES_RCVD	830	X'0000033E'
MQIAMO_TOTAL_REPAIR_PKTS_RCVD	831	X'0000033F'
MQIAMO_TOTAL_REPAIR_PKTS_RQSTD	832	X'00000340'
MQIAMO_TOTAL_MSGS_PROCESSED	833	X'00000341'
MQIAMO_TOTAL_MSGS_SELECTED	834	X'00000342'
MQIAMO_TOTAL_MSGS_EXPIRED	835	X'00000343'
MQIAMO_TOTAL_MSGS_RECEIVED	836	X'00000344'
MQIAMO_TOTAL_MSGS_VRÁCENO	837	X'00000345'
MQIAMO_LAST_USED	837	X'00000345'

MQIAMO64_* (Typy parametrů monitorování 64bitového celého čísla ve formátu příkazu)

Tabulka 188. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (selektory celočíselný systém)

Tabulka 189. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIASY_FIRST	-1	X'FFFFFFFF'

Tabulka 189. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIASY_CODED_CHAR_SET_ID	-1	X'FFFFFFFF'
MQIASY_TYPE	-2	X'FFFFFFFE'
MQIASY_COMMAND	-3	X'FFFFFFFD'
MQIASY_MSG_SEQ_NUMBER	-4	X'FFFFFFFC'
MQIASY_CONTROL	-5	X'FFFFFFFB'
MQIASY_COMP_CODE	-6	X'FFFFFFFA'
MQIASY_REASON	-7	X'FFFFFFF9'
MQIASY_BAG_OPTIONS	-8	X'FFFFFFF8'
MQIASY_VERSION	-9	X'FFFFFFF7'
MQIASY_LAST_USED	-9	X'FFFFFFF7'
MQIASY_LAST	-2000	X'FFFFF830'

MQIAUT_ * (ověřovatel záhlaví informací IMS)

Tabulka 190. Názvy konstant a hodnoty	
Název	**Hodnota**
MQIAUT_NONE	"¬¬¬¬¬¬¬¬"
MQIAUT_NONE_ARRAY	'¬','¬','¬','¬','¬','¬','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQIAV_ * (Celočíselné hodnoty atributů)

Tabulka 191. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIAV_NEPOUŽITELNÉ	-1	X'FFFFFFFF'
MQIAV_UNDEFINED	-2	X'FFFFFFFE'

MQICM_ * (režimy potvrzení záhlaví informací IMS)

Tabulka 192. Názvy konstant a hodnoty	
Název	**Hodnota**
MQICM_COMMIT_THEN_SEND	'0'
MQICM_SEND_THEN_COMMIT	'1'

MQIDO_ * (Volby nejistoty formátu příkazu)

Tabulka 193. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIDO_COMMIT	1	X'00000001'
MQIDO_BACKOUT	2	X'00000002'

MQIEP_ * (Vstupní body rozhraní)

Struktura parametrů zabezpečení připojení

Tabulka 194. Struktury konstant	
Název	**Struktura**
MQIEP_STRUC_ID	"IEP¬"
MQIEP_STRUC_ID_ARRAY	'I','E','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 195. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIEP_VERSION_1	1	X'00000001'
MQDXP_CURRENT_VERSION	1	X'00000001'

MQIGQ_ * (hromadné řazení do fronty)

Tabulka 196. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIGQ_DISABLED	0	X'00000000'
MQIGQ_ENABLED	1	X'00000001'

MQIGQPA_ * (oprávnění k zařazení do fronty ve skupině)

Tabulka 197. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝCHOZÍ-MQIGQPA_DEFAULT	1	X'00000001'
MQIGQPA_CONTEXT	2	X'00000002'
MQIGQPA_ONLY_IGQ	3	X'00000003'
MQIGQPA_ALTERNATE_OR_IGQ	4	X'00000004'

MQIIH_ * (struktura a příznaky záhlaví informací IMS)

Struktura záhlaví informací IMS

Tabulka 198. Struktury konstant	
Název	**Struktura**
MQIIH_STRUC_ID	"IIH¬"
MQIIH_STRUC_ID_ARRAY	'I','I','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 199. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIIH_VERSION_1	1	X'00000001'
MQIIH_CURRENT_VERSION	1	X'00000001'
MQIIH_LENGTH_1	84	X'00000054'

Příznaky záhlaví informací IMS

Tabulka 200. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIIH_NONE	0	X'00000000'
MQIIH_PASS_EXPIRATION	1	X'00000001'
MQIIH_UNLIMITED_EXPIRATION	0	X'00000000'
MQIIH_REPLY_FORMAT_NONE	8	X'00000008'
MQIIH_IGNORE_PURG	16	X'00000010'
MQIIH_CM0_REQUEST_RESPONSE	32	X'00000020'

MQIMPO_ * (dotazovat se na volby a strukturu vlastností zprávy)

Struktura voleb vlastností dotazové zprávy

Tabulka 201. Struktury konstant	
Název	**Struktura**
MQIMPO_STRUC_ID	"IMPO"
MQIMPO_STRUC_ID_POLE	'I','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 202. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIMPO_VERSION_1	1	X'00000001'
MQIMPO_CURRENT_VERSION	1	X'00000001'

Volby vlastnosti dotazové zprávy

Tabulka 203. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIMPO_CONVERT_TYPE	2	X'00000002'
MQIMPO_QUERY_LENGTH	4	X'00000004'
MQIMPO_INQ_FIRST	0	X'00000000'
MQIMPO_INQ_NEXT	8	X'00000008'
MQIMPO_INQ_PROP_UNDER_CURSOR	16	X'00000010'
MQIMPO_CONVERT_VALUE	32	X'00000020'
MQIMPO_NONE	0	X'00000000'

MQINBD_ * (formát příkazu Příchozí dispozice)

Tabulka 204. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQINBD_Q_MGR	0	X'00000000'
MQINBD_GROUP	3	X'00000003'

MQIND_ * (Speciální hodnoty indexu)

Tabulka 205. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIND_NONE	-1	X'FFFFFFFF'
MQIND_ALL	-2	X'FFFFFFFE'

MQIPADDR_ * (Verze adresy IP)

Tabulka 206. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIPADDR_IPV4	0	X'00000000'
MQIPADDR_IPV6	1	X'00000001'

MQISS_ * (rozsahy zabezpečení záhlaví informací IMS)

Tabulka 207. Názvy konstant a hodnoty

Název	Hodnota
MQISS_CHECK	'C'
MQISS_FULL	'F'

MQIT_ * (Typy indexů)

Tabulka 208. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIT_NONE	0	X'00000000'
MQIT_MSG_ID	1	X'00000001'
MQIT_CORREL_ID	2	X'00000002'
MQIT_MSG_TOKEN	4	X'00000004'
ID_skupiny_MQIT_ID	5	X'00000005'

MQITEM_ * (Typ položky pro mqInquireItemInfo)

Tabulka 209. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQITEM_INTEGER	1	X'00000001'
MQITEM_STRING	2	X'00000002'
MQITEM_BAG	3	X'00000003'
MQITEM_BYTE_STRING	4	X'00000004'
MQITEM_INTEGER_FILTER	5	X'00000005'
MQITEM_STRING_FILTER	6	X'00000006'
MQITEM_INTEGER64	7	X'00000007'
MQITEM_BYTE_STRING_FILTER	8	X'00000008'

MQITII_ * (IMS identifikátor instance transakce záhlaví informací)

Tabulka 210. Názvy konstant a hodnoty	
Název	**Hodnota**
MQITII_NONE	`X'00...00'` (16 nulových hodnot)
MQITII_NONE_ARRAY	`'\0','\0',...` (16 nulových hodnot)

MQITS_ * (IMS stavy transakcí záhlaví informací)

Tabulka 211. Názvy konstant a hodnoty	
Název	**Hodnota**
MQITS_IN_CONVERSATION	`'C'`
MQITS_NOT_IN_CONVERSATION	`'¬'`
MQITS_ARCHITEKTED	`'A'`

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQKAI_ * (intervalKeepAlive)

Tabulka 212. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQKAI_AUTO	-1	X'FFFFFFFF'

MQMASTER_ * (hlavní administrace)

Tabulka 213. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMASTER_NO	0	X'00000000'
MQMASTER_YES	1	X'00000001'

MQMCAS_ * (Formát příkazu Stav agenta kanálu zpráv)

Tabulka 214. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMCAS_ZASTAVENO	0	X'00000000'
SPUŠTĚNÍ_MQMCAS_RUNNING	3	X'00000003'

MQMCAT_ * (typy MCA)

Tabulka 215. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMCAT_PROCESS	1	X'00000001'
MQMCAT_THREAD	2	X'00000002'

MQMCD_ * (Informace o značce voleb publikování/odběru)

Deskriptor obsahu zprávy značek voleb publikování/odběru (mcd)

Tabulka 216. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMCD_FOLDER_VERSION	1	X'00000001'

Názvy značek značek voleb publikování/odběru

Tabulka 217. Názvy konstant a hodnoty	
Název	**Hodnota**
MQMCD_MSG_DOMAIN	"Msd"
MQMCD_MSG_SET	"Set"
MQMCD_MSG_TYPE	"Type"
MQMCD_MSG_FORMAT	"Fmt"

Názvy značek XML značek voleb publikování/odběru

Tabulka 218. Názvy konstant a hodnoty	
Název	**Hodnota**
MQMCD_MSG_DOMAIN_B	"<Msd>"
MQMCD_MSG_DOMAIN_E	"</Msd>"
MQMCD_MSG_SET_B	"<Set>"
MQMCD_MSG_SET_E	"</Set>"
MQMCD_MSG_TYPE_B	"<Type>"
MQMCD_MSG_TYPE_E	"</Type>"
MQMCD_MSG_FORMAT_B	"<Fmt>"
MQMCD_MSG_FORMAT_E	"</Fmt>"

Hodnoty značek voleb publikování/odběru

Tabulka 219. Názvy konstant a hodnoty	
Název	**Hodnota**
MQMCD_DOMAIN_NONE	"none"
MQMCD_DOMAIN_NEON	"neon"
MQMCD_DOMAIN_MRM	"mrm"
MQMCD_DOMAIN_JMS_NONE	"jms_none"
MQMCD_DOMAIN_JMS_TEXT	"jms_text"
MQMCD_DOMAIN_JMS_OBJECT	"jms_object"
MQMCD_DOMAIN_JMS_MAP	"jms_map"
PROUD MQMCD_DOMAIN_JMS_STREAM	"jms_stream"
MQMCD_DOMAIN_JMS_BYTES	"jms_bytes"

MQMD_ * (Struktura deskriptoru zpráv)

Tabulka 220. Struktury konstant	
Název	**Struktura**
ID_STRUC_MQMD_STRUC_ID	"MD¬¬"
MQMD_STRUC_ID_ARRAY	'M','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 221. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMD_VERSION_1	1	X'00000001'
MQMD_VERSION_2	2	X'00000002'
MQMD_CURRENT_VERSION	2	X'00000002'

MQMDE_ * (struktura rozšíření deskriptoru zprávy)

Tabulka 222. Struktury konstant	
Název	**Struktura**
MQMDE_STRUC_ID	"MDE¬"
MQMDE_STRUC_ID_POLE	'M','D','E','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 223. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMDE_VERSION_2	2	X'00000002'
MQMDE_CURRENT_VERSION	2	X'00000002'
MQMDE_LENGTH_2	72	X'00000048'

MQMDEF_ * (Příznaky rozšíření deskriptoru zprávy)

Tabulka 224. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMDEF_NONE	0	X'00000000'

MQMDS_ * (posloupnost doručení zprávy)

Tabulka 225. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMDS_PRIORITY	0	X'00000000'
MQMDS_FIFO	1	X'00000001'

MQMF_ * (příznaky zprávy)

Tabulka 226. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMF_SEGMENTATION_INHIBITED	0	X'00000000'
MQMF_SEGMENTATION_ALLOWED	1	X'00000001'

Tabulka 226. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMF_MSG_IN_GROUP	8	X'00000008'
MQMF_LAST_MSG_IN_GROUP	16	X'00000010'
MQMF_SEGMENT	2	X'00000002'
MQMF_LAST_SEGMENT	4	X'00000004'
MQMF_NONE	0	X'00000000'

MQMHBO_ * (popisovač zprávy pro volby a strukturu vyrovnávací paměti)

Struktura popisovače zprávy pro volby vyrovnávací paměti

Tabulka 227. Struktury konstant	
Název	**Struktura**
MQMHBO_STRUC_ID	"MHBO"
MQMHBO_STRUC_ID_ARRAY	'M','H','B','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 228. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMHBO_VERSION_1	1	X'00000001'
MQMHBO_CURRENT_VERSION	1	X'00000001'

Volby popisovače zprávy do vyrovnávací paměti

Tabulka 229. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMHBO_PROPERTIES_IN_MQRFH2	1	X'00000001'
MQMHBO_DELETE_PROPERTIES	2	X'00000002'
MQMHBO_NONE	0	X'00000000'

MQMI_ * (Identifikátor zprávy)

Tabulka 230. Názvy konstant a hodnoty	
Název	**Hodnota**
MQMI_NONE	X'00...00' (24 nulových hodnot)
Pole MQMI_NONE_ARRAY	'\0','\0',... (24 nulových hodnot)

MQMMBI_ * (Značka zprávy-interval procházení)

Tabulka 231. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMMBI_UNLIMITED	-1	X'FFFFFFFF'

MQMO_ * (Volby shody)

Tabulka 232. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQMO_MATCH_MSG_ID	1	X'00000001'
MQMO_MATCH_CORREL_ID	2	X'00000002'
MQMO_MATCH_GROUP_ID	4	X'00000004'
MQMO_MATCH_MSG_SEQ_NUMBER	8	X'00000008'
MQMO_MATCH_OFFSET	16	X'00000010'
MQMO_MATCH_MSG_TOKEN	32	X'00000020'
MQMO_NONE	0	X'00000000'

MQMODE_ * (Volby režimu formátu příkazu)

Tabulka 233. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQMODE_FORCE	0	X'00000000'
MQMODE_QUIESCE	1	X'00000001'
MQMODE_TERMINATE	2	X'00000002'

MQMON_ * (Hodnoty monitorování)

Tabulka 234. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQMON_NOT_AVAILABLE	-1	X'FFFFFFFF'
MQMON_NONE	-1	X'FFFFFFFF'
MQMON_Q_MGR	-3	X'FFFFFFFD'
MQMON_OFF	0	X'00000000'
MQMON_ON	1	X'00000001'
MQMON_DISABLED	0	X'00000000'
MQMON_ENABLED	1	X'00000001'
MQMON_LOW	17	X'00000011'
MQMON_MEDIUM	33	X'00000021'
MQMON_HIGH-vysoká	65	X'00000041'

MQMT_ * (Typy zpráv)

Tabulka 235. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
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'

Tabulka 235. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMT_MQE_FIELDS	113	X'00000071'
MQMT_SYSTEM_LAST	65535	X'0000FFFF'
MQMT_APPL_FIRST	65536	X'00010000'
MQMT_APPL_LAST	999999999	X'3B9AC9FF'

MQMTOK_ * (Token zprávy)

Tabulka 236. Názvy konstant a hodnoty	
Název	**Hodnota**
MQMTOK_NONE	X'00...00' (16 nulových hodnot)
MQMTOK_NONE_ARRAY	'\0','\0',... (16 nulových hodnot)

MQNC_ * (Počet názvů)

Tabulka 237. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQNC_MAX_NAMELIST_NAME_COUNT	256	X'00000100'

MQNPM_ * (Třída přechodné zprávy)

Tabulka 238. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQNPM_CLASS_NORMAL	0	X'00000000'
MQNPM_CLASS_HIGH	10	X'0000000A'

MQNPMS_ * (NonPersistent-rychlosti zpráv)

Tabulka 239. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQNPMS_NORMAL normální	1	X'00000001'
MQNPMS_FAST	2	X'00000002'

MQNT_ * (Typy seznamu názvů)

Tabulka 240. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQNT_NONE	0	X'00000000'
MQNT_Q	1	X'00000001'
MQNT_CLUSTER	2	X'00000002'
MQNT_AUTH_INFO	4	X'00000004'
MQNT_ALL	1001	X'000003E9'

MQNVS_ * (Názvy pro název/hodnotu řetězce)

Tabulka 241. Názvy konstant a hodnoty

Název	Hodnota
MQNVS_APPL_TYPE	"OPT_APP_GRP¬"
MQNVS_MSG_TYPE	"OPT_MSG_TYPE¬"

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQOA_ * (Omezení pro selektory pro atributy objektu)

Tabulka 242. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOA_FIRST	1	X'00000001'
MQOA_LAST	9000	X'00002328'

MQOD_ * (Struktura deskriptoru objektu)

Tabulka 243. Struktury konstant

Název	Struktura
ID_STRUC_MQOD_STRUC_ID	"OD¬¬"
MQOD_STRUC_ID_ARRAY	'O','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 244. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOD_VERSION_1	1	X'00000001'
MQOD_VERSION_2	2	X'00000002'
MQOD_VERSION_3	3	X'00000003'
MQOD_VERSION_4	4	X'00000004'
MQOD_CURRENT_VERSION	4	X'00000004'
MQOD_CURRENT_LENGTH	(value differs by platform or version)	(value differs by platform or version)

MQOII_ * (Identifikátor instance objektu)

Tabulka 245. Názvy konstant a hodnoty

Název	Hodnota
MQOII_NONE	X'00...00' (24 nulových hodnot)
MQOII_NONE_ARRAY	'\0','\0',... (24 nulových hodnot)

MQOL_ * (Původní délka)

Tabulka 246. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOL_UNDEFINED	-1	X'FFFFFFFF'

MQOM_ * (Volby zastaralých zpráv Db2 ve skupině dotazů)

Tabulka 247. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOM_NO	0	X'00000000'
MQOM_YES	1	X'00000001'

MQOO_ * (Volby otevření)

Tabulka 248. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOO_BIND_AS_Q_DEF	0	X'00000000'
MQOO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQOO_INPUT_AS_Q_DEF	1	X'00000001'
MQOO_INPUT_SHARED	2	X'00000002'
MQOO_INPUT_EXCLUSIVE	4	X'00000004'
MQOO_BROWSE	8	X'00000008'
MQOO_OUTPUT	16	X'00000010'
MQOO_DOTAZOVAT	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'
Oprávnění MQOO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'
MQOO_FAIL_IF_QUIESCING	8192	X'00002000'
MQOO_BIND_ON_OPEN	16384	X'00004000'
MQOO_BIND_NOT_FIXED	32768	X'00008000'
MQOO_CO_OP	131072	X'00020000'
MQOO_RESOLVE_LOCAL_TOPIC	262144	X'00040000'
MQOO_NO_READ_AHEAD	524288	X'00080000'
MQOO_READ_AHEAD	1048576	X'00100000'
MQOO_BIND_ON_GROUP	4194304	X'00400000'

*MQOO_ * (Sledování se používá pouze v C++)*

Tabulka 249. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOO_RESOLVE_NAMES	65536	X'00010000'
MQOO_RESOLVE_LOCAL_Q	262144	X'00040000'

MQOP_ * (kódy operací pro MQCTL a MQCB)

Kódy operací pro MQCTL

Tabulka 250. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOP_START	1	X'00000001'
MQOP_START_WAIT	2	X'00000002'
MQOP_STOP	4	X'00000004'

Kódy operací pro MQCB

Tabulka 251. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOP_REGISTER	256	X'00000100'
MQOP_DEREGISTER	512	X'00000200'

Kódy operací pro MQCTL a MQCB

Tabulka 252. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOP_SUSPEND	65536	X'00010000'
MQOP_RESUME	131072	X'00020000'

MQOPEN_ * (hodnoty související se strukturou MQOPEN_PRIV)

Tabulka 253. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOPEN_PRIV_VERSION_1	1	X'00000001'
MQOPEN_PRIV_CURRENT_VERSION	1	X'00000001'

MQOPER_ * (Operace aktivity)

Tabulka 254. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOPER_SYSTEM_FIRST	0	X'00000000'
MQOPER_UNKNOWN (neznámá)	0	X'00000000'
MQOPER_BROWSE	1	X'00000001'
MQOPER_DISCARD	2	X'00000002'
MQOPER_GET	3	X'00000003'
MQOPER_PUT	4	X'00000004'
MQOPER_PUT_REPLY	5	X'00000005'
MQOPER_PUT_REPORT	6	X'00000006'
MQOPER_RECEIVE	7	X'00000007'
MQOPER_SEND	8	X'00000008'
MQOPER_TRANSFORM	9	X'00000009'

Tabulka 254. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOPER_PUBLISH	10	X'0000000A'
MQOPER_LUED_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_ * (Typy objektů a rozšířené typy objektů)

Typy objektů

Tabulka 255. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOT_NONE	0	X'00000000'
MQOT_Q	1	X'00000001'
MQOT_NAMELIST	2	X'00000002'
MQOT_PROCESS	3	X'00000003'
MQOT_STORAGE_CLASS	4	X'00000004'
MQOT_Q_MGR	5	X'00000005'
MQOT_CHANNEL	6	X'00000006'
MQOT_AUTH_INFO	7	X'00000007'
MQOT_TOPIC	8	X'00000008'
MQOT_CF_STRUC	10	X'0000000A'
MQOT_LISTENER	11	X'0000000B'
MQOT_SERVICE	12	X'0000000C'
MQOT_RESERVED_1	999	X'000003E7'

Typy rozšířených objektů

Tabulka 256. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOT_ALL	1001	X'000003E9'
MQOT_ALIAS_Q	1002	X'000003EA'
MQOT_MODEL_Q	1003	X'000003EB'
MQOT_LOCAL_Q	1004	X'000003EC'
MQOT_REMOTE_Q	1005	X'000003ED'
MQOT_SENDER_CHANNEL	1007	X'000003EF'
MQOT_SERVER_CHANNEL	1008	X'000003F0'
MQOT_REQUESTER_CHANNEL (kanál žadatele)	1009	X'000003F1'
MQOT_RECEIVER_CHANNEL	1010	X'000003F2'
MQOT_CURRENT_CHANNEL	1011	X'000003F3'
MQOT_SAVED_CHANNEL	1012	X'000003F4'

Tabulka 256. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOT_SVRCONN_CHANNEL	1013	X'000003F5'
MQOT_CLNTCONN_CHANNEL	1014	X'000003F6'
MQOT_SHORT_CHANNEL	1015	X'000003F7'
MQOT_CHLAUTH	1016	X'000003F8'
MQOT_REMOTE_Q_MGR_NAME	1017	X'000003F9'
MQOT_PROT_POLICY	1019	X'000003FB'
MQOT_TT_CHANNEL	1020	X'000003FC'
MQOT_AMQP_CHANNEL	1021	X'000003FD'
MQOT_AUTH_REC	1022	X'000003FE'

MQPA_ * (oprávnění k vložení)

Tabulka 257. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝCHOZÍ	1	X'00000001'
MQPA_CONTEXT	2	X'00000002'
MQPA_ONLY_MCA	3	X'00000003'
MQPA_ALTERNATE_OR_MCA	4	X'00000004'

MQPD_ * (deskriptor vlastnosti, podpora a kontext)

Struktura deskriptoru vlastností

Tabulka 258. Struktury konstant	
Název	**Struktura**
ID_STRUC_MQPD_STRUC_ID	"PD¬¬"
MQPD_STRUC_ID_ARRAY	'P','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 259. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_VERSION_1	1	X'00000001'
MQPD_CURRENT_VERSION	1	X'00000001'

Volby deskriptoru vlastností

Tabulka 260. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_NONE	0	X'00000000'

Volby podpory vlastností

Tabulka 261. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_SUPPORT_OPTIONAL	1	X'00000001'
MQPD_SUPPORT_REQUIRED	1048576	X'00100000'
MQPD_SUPPORT_REQUIRED_IF_LOCAL	1024	X'00000400'
MQPD_REJECT_UNSUP_MASK	-1048576	X'FFF00000'
MQPD_ACCEPT_UNSUP_IF_XMIT_MASK	1047552	X'000FFC00'
MQPD_ACCEPT_UNSUP_MASK	1023	X'000003FF'

Kontext vlastnosti

Tabulka 262. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_NO_CONTEXT	0	X'00000000'
MQPD_USER_CONTEXT	1	X'00000001'

MQPER_ * (Hodnoty perzistence)

Tabulka 263. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPER_PERSISTENCE_AS_PARENT	-1	X'FFFFFFFF'
MQPER_NOT_PERSISTENT	0	X'00000000'
MQPER_PERSISTENT	1	X'00000001'
MQPER_PERSISTENCE_AS_Q_DEF	2	X'00000002'
MQPER_PERSISTENCE_AS_TOPIC_DEF	2	X'00000002'

MQPL_ * (Platformy)

Tabulka 264. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPL_MVS	1	X'00000001'
MQPL_OS390	1	X'00000001'
MQPL_ZOS	1	X'00000001'
MQPL_OS2	2	X'00000002'
MQPL_AIX	3	X'00000003'
MQPL_UNIX	3	X'00000003'
MQPL_OS400	4	X'00000004'
MQPL_WINDOWS	5	X'00000005'
MQPL_WINDOWS_NT	11	X'0000000B'
MQPL_VMS	12	X'0000000C'
MQPL_NSK	13	X'0000000D'
MQPL_OPEN_TP1	15	X'0000000F'
MQPL_VM	18	X'00000012'

Tabulka 264. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPL_TPF	23	X'00000017'
MQPL_VSE	27	X'0000001B'
MQPL_APPLIANCE	28	X'0000001C'
MQPL_NATIVE	1	X'00000001'

MQPMO_ * (Volby vložení zprávy a struktura pro masku publikování)

Struktura voleb vložení zprávy

Tabulka 265. Struktury konstant	
Název	**Struktura**
MQPMO_STRUC_ID	"PMO¬"
MQPMO_STRUC_ID_POLE	'P','M','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 266. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMO_VERSION_1	1	X'00000001'
MQPMO_VERSION_2	2	X'00000002'
MQPMO_VERSION_3	3	X'00000003'
MQPMO_CURRENT_VERSION	3	X'00000003'
MQPMO_CURRENT_LENGTH	(value differs by platform or version)	(value differs by platform or version)

Volby vložení zprávy

Tabulka 267. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMO_SYNCPOINT	2	X'00000002'
MQPMO_NO_SYNCPOINT	4	X'00000004'
MQPMO_DEFAULT_CONTEXT	32	X'00000020'
MQPMO_NEW_MSG_ID	64	X'00000040'
MQPMO_NEW_CORREL_ID	128	X'00000080'
MQPMO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQPMO_PASS_ALL_CONTEXT	512	X'00000200'
MQPMO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQPMO_SET_ALL_CONTEXT	2048	X'00000800'
Oprávnění MQPMO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'
MQPMO_FAIL_IF_QUIESCING	8192	X'00002000'
MQPMO_NO_CONTEXT	16384	X'00004000'
MQPMO_LOGICAL_ORDER	32768	X'00008000'
MQPMO_ASYNC_RESPONSE	65536	X'00010000'

Tabulka 267. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMO_SYNC_RESPONSE	131072	X'00020000'
MQPMO_RESOLVE_LOCAL_Q	262144	X'00040000'
MQPMO_RETAIN	2097152	X'00200000'
MQPMO_MD_FOR_OUTPUT_ONLY	8388608	X'00800000'
MQPMO_SCOPE_QMGR	67108864	X'04000000'
MQPMO_SUPPRESS_REPLYTO	134217728	X'08000000'
MQPMO_NOT_OWN_SUBS	268435456	X'10000000'
MQPMO_RESPONSE_AS_Q_DEF	0	X'00000000'
MQPMO_RESPONSE_AS_TOPIC_DEF	0	X'00000000'
MQPMO_NONE	0	X'00000000'

Volby vložení zprávy pro masku publikování

Tabulka 268. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MASKA voleb MQPMO_PUB_OPTIONS_MASK	2097152	X'00200000'

MQPMRF_ * (Vložit pole záznamu zprávy)

Tabulka 269. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMRF_MSG_ID	1	X'00000001'
MQPMRF_CORREL_ID	2	X'00000002'
MQPMRF_GROUP_ID	4	X'00000004'
MQPMRF_ZPĚTNÁ vazba	8	X'00000008'
MQPMRF_ACCOUNTING_TOKEN	16	X'00000010'
MQPMRF_NONE	0	X'00000000'

MQPO_ * (Volby vymazání formátu příkazu)

Tabulka 270. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPO_YES	1	X'00000001'
MQPO_NO	0	X'00000000'

MQPRI_ * (Priorita)

Tabulka 271. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (řídicí hodnoty vlastností fronty a kanálu a maximální délka vlastností)

Řídicí hodnoty vlastností fronty a kanálu

Tabulka 272. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPROP_KOMPATIBILITA	0	X'00000000'
MQPROP_NONE	1	X'00000001'
MQPROP_ALL	2	X'00000002'
MQPROP_FORCE_MQRFH2	3	X'00000003'

Maximální délka vlastností

Tabulka 273. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPROP_UNRESTRICTED_LENGTH	-1	X'FFFFFFFF'

MQPRT_ * (Hodnoty odezvy vložení)

Tabulka 274. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPRT_RESPONSE_AS_PARENT	0	X'00000000'
MQPRT_SYNC_RESPONSE	1	X'00000001'
MQPRT_ASYNC_RESPONSE	2	X'00000002'

MQPS_ * (Vydavatel/odběr)

Formát příkazu Stav publikování/odběru

Tabulka 275. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPS_STATUS_INACTIVE	0	X'00000000'
MQPS_STATUS_STARTING	1	X'00000001'
MQPS_STATUS_STOPPING	2	X'00000002'
MQPS_STATUS_ACTIVE	3	X'00000003'
MQPS_STATUS_COMPAT	4	X'00000004'
MQPS_STATUS_CHYBA	5	X'00000005'
MQPS_STATUS_REFUSED	6	X'00000006'

Značky publikování/odběru jako řetězce

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

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

Značky publikování/odběru jako prázdné uzavřené řetězce

MQPS_COMMAND_B	"¬MQPSCommand¬"
MQPS_COMP_CODE_B	"¬MQPSCompCode¬"
MQPS_CORREL_ID_B	"¬MQPSCorrelId¬"
MQPS_DELETE_OPTIONS_B	"¬MQPSDelOpts¬"
MQPS_ERROR_ID_B	"¬MQPSErrorId¬"
MQPS_ERROR_POS_B	"¬MQPSErrorPos¬"
MQPS_INTEGER_DATA_B	"¬MQPSIntData¬"
MQPS_PARAMETER_ID_B	"¬MQPSParmId¬"
MQPS_PUBLICATION_OPTIONS_B	"¬MQPSPubOpts¬"
MQPS_PUBLISH_TIMESTAMP_B	"¬MQPSPubTime¬"
MQPS_Q_MGR_NAME_B	"¬MQPSQMgrName¬"
MQPS_Q_NAME_B	"¬MQPSQName¬"
MQPS_REASON_B	"¬MQPSReason¬"
MQPS_REASON_TEXT_B	"¬MQPSReasonText¬"
MQPS_REGISTRATION_OPTIONS_B	"¬MQPSRegOpts¬"

MQPS_SEQUENCE_NUMBER_B	"¬MQPSSeqNum¬"
MQPS_STREAM_NAME_B	"¬MQPSStreamName¬"
MQPS_STRING_DATA_B	"¬MQPSStringData¬"
MQPS_SUBSCRIPTION_IDENTITY_B	"¬MQPSSubIdentity¬"
MQPS_SUBSCRIPTION_NAME_B	"¬MQPSSubName¬"
MQPS_SUBSCRIPTION_USER_DATA_B	"¬MQPSSubUserData¬"
MQPS_TOPIC_B	"¬MQPSTopic¬"
MQPS_USER_ID_B	"¬MQPSUserId¬"

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Hodnoty značek příkazů publikování/odběru jako řetězce

MQPS_DELETE_PUBLIKACE	"DeletePub"
MQPS_DEREGISTER_PUBLISHER-vydavatel	"DeregPub"
MQPS_DEREGISTER_ODBĚRATEL	"DeregSub"
MQPS_PUBLISH	"Publish"
MQPS_REGISTER_PUBLISHER.	"RegPub"
MQPS_REGISTER_ODBĚRATEL	"RegSub"
MQPS_REQUEST_UPDATE	"ReqUpdate"

Hodnoty značek příkazu publikování/odběru jako prázdné uzavřené řetězce

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

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Hodnoty značek voleb publikování/odběru jako řetězce

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

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

Hodnoty značek voleb publikování/odběru jako prázdné uzavřené řetězce

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

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

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQPSC_ * (Složka příkazů pro publikování/odběr (PSC) Značky)

Tabulka 276. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQPSC_FOLDER_VERSION	1	X'00000001'

*MQPSC_ * (Názvy značek voleb publikování/odběru)*

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

*MQPSC_ * (Názvy značek XML značek voleb publikování/odběru)*

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

*MQPSC_ * (Hodnoty vydavatele značek voleb publikování/odběru jako řetězce)*

MQPSC_DELETE_PUBLIKACE	"DeletePub"
MQPSC_DEREGISTER_ODBĚRATEL	"DeregSub"
MQPSC_PUBLISH	"Publish"
MQPSC_REGISTER_ODBĚRATEL	"RegSub"

MQPSC_REQUEST_UPDATE	"ReqUpdate"

*MQPSC_ * (Volby publikování/odběru Hodnoty názvu značky jako řetězce)*

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

MQPSCR_ * (Volby publikování/odběru)

Volby publikování/odběru Značka Publikovat/Odebírat složku odpovědí (pscr) Značky

Tabulka 277. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQPSCR_FOLDER_VERSION	1	X'00000001'

Názvy značek značek voleb publikování/odběru

MQPSCR_DOKONČENÍ	"Completion"
MQPSCR_RESPONSE	"Response"

MQPSCR_REASON	"Reason"

Názvy značek XML značek voleb publikování/odběru

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

Hodnoty značek voleb publikování/odběru

MQPSCR_OK	"ok"
MQPSCR_VAROVÁNÍ	"warning"
MQPSCR_CHYBA	"error"

MQPSM_ * (publikování/dílčí režim)

Tabulka 278. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQPSM_DISABLED	0	X'00000000'
MQPSM_COMPAT	1	X'00000001'
MQPSM_ENABLED	2	X'00000002'

MQPSPROP_ * (Vlastnosti publikování/odběru zprávy)

Tabulka 279. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQPSPROP_NONE	0	X'00000000'
MQPSPROP_COMPAT-nejlepší volba	1	X'00000001'
MQPSPROP_RFH2	2	X'00000002'
MQPSPROP_MSGPROP	3	X'00000003'

MQPSST_ * (Typ stavu publikování/odběru ve formátu příkazu)

Tabulka 280. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQPSST_ALL	0	X'00000000'
MQPSST_LOCAL	1	X'00000001'
MQPSST_PARENT	2	X'00000002'
MQPSST_CHILD	3	X'00000003'

MQPUBO_ * (Volby publikování publikování/odběru)

Tabulka 281. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPUBO_NONE	0	X'00000000'
MQPUBO_CORREL_ID_AS_IDENTITY	1	X'00000001'
MQPUBO_RETAIN_PUBLIKACE	2	X'00000002'
MQPUBO_OTHER_SUBSCRIBERS_ONLY	4	X'00000004'
MQPUBO_NO_REGISTRATION	8	X'00000008'
MQPUBO_IS_RETAINED_PUBLIKACE	16	X'00000010'

MQPXP_ * (struktura parametru uživatelské procedury směrování pro publikování/odběr)

Tabulka 282. Struktury konstant	
Název	**Struktura**
MQPXP_STRUC_ID	"PXP¬"
MQPXP_STRUC_ID_ARRAY	'P','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 283. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPXP_VERSION_1	1	X'00000001'
MQPXP_CURRENT_VERSION	1	X'00000001'

MQQA_ * (atributy fronty)

Zakázat získání hodnot

Tabulka 284. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQA_GET_INHIBITED	1	X'00000001'
MQQA_GET_ALLOWED	0	X'00000000'

Zablokovat hodnoty vložení

Tabulka 285. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQA_PUT_BLOKOVÁNO	1	X'00000001'
MQQA_PUT_POVOLENO	0	X'00000000'

Sdílení front

Tabulka 286. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQA_SHAREABLE	1	X'00000001'
MQQA_NOT_SHAREABLE	0	X'00000000'

Back-out vytvrzování

Tabulka 287. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQA_BACKOUT_HARDENED	1	X'00000001'
MQQA_BACKOUT_NOT_HARDENED	0	X'00000000'

MQQDT_ * (Typy definic front)

Tabulka 288. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQDT_PŘEDDEFINOVANÝ	1	X'00000001'
MQQDT_PERMANENT_DYNAMIC	2	X'00000002'
MQQDT_TEMPORARY_DYNAMIC	3	X'00000003'
MQQDT_SHARED_DYNAMIC	4	X'00000004'

MQQF_ * (příznaky fronty)

Tabulka 289. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQF_LOCAL_Q	1	X'00000001'
MQQF_CLWL_USEQ_ANY	64	X'00000040'
MQQF_CLWL_USEQ_LOCAL	128	X'00000080'

MQQMDT_ * (Typy definic správce front ve formátu příkazů)

Tabulka 290. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (příznaky správce front)

Tabulka 291. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQMF_REPOSITORY_Q_MGR	2	X'00000002'
MQQMF_CLUSSDR_USER_DEFINED	8	X'00000008'
MQQMF_CLUSSDR_AUTO_DEFINED	16	X'00000010'
MQQMF_AVAILABLE	32	X'00000020'

MQQMFAC_ * (Prostředek správce front ve formátu příkazů)

Tabulka 292. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQMFAC_IMS_BRIDGE	1	X'00000001'

Tabulka 292. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQMFAC_DB2	2	X'00000002'

MQQMSTA_ * (Formát příkazu Stav správce front)

Tabulka 293. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQMSTA_STARTING	1	X'00000001'
MQQMSTA_RUNNING	2	X'00000002'
MQQMSTA_QUIESCING	3	X'00000003'

MQQMT_ * (Typy správců front ve formátu příkazů)

Tabulka 294. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQMT_NORMAL	0	X'00000000'
MQQMT_REPOSITORY	1	X'00000001'

MQQO_ * (Volby uvedení formátu příkazu do klidového stavu)

Tabulka 295. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQO_YES	1	X'00000001'
MQQO_NO	0	X'00000000'

MQQSGD_ * (dispozice skupiny sdílení front)

Tabulka 296. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQSGD_ALL	-1	X'FFFFFFFF'
MQQSGD_Q_MGR	0	X'00000000'
MQQSGD_COPY	1	X'00000001'
MQQSGD_SHARED	2	X'00000002'
MQQSGD_GROUP	3	X'00000003'
MQQSGD_PRIVATE	4	X'00000004'
MQQSGD_LIVE	6	X'00000006'

MQQSGS_ * (Stav skupiny sdílení front formátu příkazu)

Tabulka 297. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQSGS_UNKNOWN	0	X'00000000'
MQQSGS_CREATED	1	X'00000001'
MQQSGS_ACTIVE	2	X'00000002'
MQQSGS_INACTIVE	3	X'00000003'
MQQSGS_FAILED	4	X'00000004'

Tabulka 297. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQSGS_PENDING	5	X'00000005'

MQQSIE_ * (Služba fronty ve formátu příkazů-Události intervalu)

Tabulka 298. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQSIE_NONE	0	X'00000000'
MQQSIE_HIGH	1	X'00000001'
MQQSIE_OK	2	X'00000002'

MQQSO_ * (Volby otevření stavu fronty pro SET, BROWSE, INPUT)

Tabulka 299. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQSO_NO	0	X'00000000'
MQQSO_YES	1	X'00000001'
MQQSO_SHARED	1	X'00000001'
MQQSO_EXCLUSIVE	2	X'00000002'

MQQSOT_ * (Typ otevření stavu fronty formátu příkazu)

Tabulka 300. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQSOT_ALL	1	X'00000001'
MQQSOT_INPUT	2	X'00000002'
MQQSOT_OUTPUT	3	X'00000003'

MQQSUM_ * (Formát příkazu Nepotvrzené zprávy o stavu fronty)

Tabulka 301. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQSUM_YES	1	X'00000001'
MQQSUM_NO	0	X'00000000'

MQQT_ * (Typy front a rozšířené typy front)

Typy front

Tabulka 302. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQT_LOCAL	1	X'00000001'
MQQT_MODEL	2	X'00000002'
MQQT_ALIAS	3	X'00000003'
MQQT_REMOTE	6	X'00000006'
MQQT_CLUSTER	7	X'00000007'

Rozšířené typy front

Tabulka 303. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQT_ALL	1001	X'000003E9'

MQRC_ * (kódy příčiny)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRC_NO_EXTERNAL_ÚČASTNÍKŮ	2121	X'00000849'
MQRC_PARTICIPANT_NOT_AVAILABLE	2122	X'0000084A'
MQRC_OUTCOME_MIXED (smíšený)	2123	X'0000084B'
MQRC_OUTCOME_PENDING	2124	X'0000084C'
MQRC_BRIDGE_STARTED	2125	X'0000084D'
MQRC_BRIDGE_ZASTAVENO	2126	X'0000084E'
MQRC_ADAPTER_STORAGE_SHORTAGE	2127	X'0000084F'
MQRC_UOW_IN_PROGRESS	2128	X'00000850'
MQRC_ADAPTER_CONN_LOAD_ERROR	2129	X'00000851'
MQRC_ADAPTER_SERV_LOAD_ERROR	2130	X'00000852'
MQRC_ADAPTER_DEFS_ERROR	2131	X'00000853'
MQRC_ADAPTER_DEFS_LOAD_ERROR	2132	X'00000854'
MQRC_ADAPTER_CONV_LOAD_ERROR	2133	X'00000855'
MQRC_BO_ERROR	2134	X'00000856'
MQRC_DH_ERROR	2135	X'00000857'
MQRC_MULTIPLE_PŘÍČINA	2136	X'00000858'
MQRC_OPEN_FAILED	2137	X'00000859'
MQRC_ADAPTER_DISC_LOAD_ERROR	2138	X'0000085A'
MQRC_CNO_ERROR	2139	X'0000085B'
MQRC_CICS_WAIT_FAILED	2140	X'0000085C'
MQRC_DLH_ERROR	2141	X'0000085D'
MQRC_HEADER_CHYBA	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_CHYBA	2148	X'00000864'
MQRC_PCF_ERROR	2149	X'00000865'
MQRC_DBCS_ERROR	2150	X'00000866'
MQRC_OBJECT_NAME_ERROR	2152	X'00000868'
MQRC_OBJECT_Q_MGR_NAME_ERROR	2153	X'00000869'
MQRC_RECS_PRESENT_ERROR	2154	X'0000086A'
MQRC_OBJECT_RECORDS_ERROR	2155	X'0000086B'
MQRC_RESPONSE_RECORDS_ERROR	2156	X'0000086C'
MQRC_ASID_MISMATCH	2157	X'0000086D'
MQRC_PMO_RECORD_FLAGS_ERROR	2158	X'0000086E'
MQRC_PUT_MSG_RECORDS_ERROR	2159	X'0000086F'
MQRC_CONN_ID_IN_USE	2160	X'00000870'
MQRC_Q_MGR_QUIESCING	2161	X'00000871'

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

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

Tabulka 304. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRC_RFH_RESTRICTED_FORMAT_ERR	2527	X'000009DF'
MQRC_CONNECTION_STOPPED	2528	X'000009E0'
MQRC_ASYNC_UOW_CONFLICT	2529	X'000009E1'
MQRC_ASYNC_XA_CONFLICT	2530	X'000009E2'
MQRC_PUBSUB_INHIBITED	2531	X'000009E3'
MQRC_MSG_HANDLE_COPY_FAILURE	2532	X'000009E4'
MQRC_DEST_CLASS_NOT_ALTERABLE	2533	X'000009E5'
MQRC_OPERATION_NOT_ALLOWED	2534	X'000009E6'
MQRC_ACTION_ERROR	2535	X'000009E7'
MQRC_CHANNEL_NOT_AVAILABLE	2537	X'000009E9'
MQRC_HOST_NENÍ k dispozici	2538	X'000009EA'
MQRC_CHANNEL_CONFIG_ERROR	2539	X'000009EB'
MQRC_UNKNOWN_CHANNEL_NAME	2540	X'000009EC'
MQRC_LOOPING_PUBLIKACE	2541	X'000009ED'
MQRC_ALREADY_JOINED	2542	X'000009EE'
MQRC_CHANNEL_SSL_VAROVÁNÍ	2552	X'000009F8'
MQRC_OCSP_URL_ERROR	2553	X'000009F9'
MQRC_CIPHER_SPEC_NOT_SUITE_B	2591	X'00000A1F'
MQRC_SUITE_B_ERROR	2592	X'00000A20'
MQRC_PASSWORD_PROTECTION_ERROR	2594	X'00000A22'
MQRC_REOPEN_EXCL_INPUT_ERROR	6100	X'000017D4'
MQRC_REOPEN_INQUIRE_ERROR	6101	X'000017D5'
MQRC_REOPEN_SAVED_CONTEXT_ERR	6102	X'000017D6'
MQRC_REOPEN_TEMPORARY_Q_ERROR	6103	X'000017D7'
MQRC_ATTRIBUTE_LOCKED	6104	X'000017D8'
MQRC_CURSOR_NOT_VALID	6105	X'000017D9'
MQRC_ENCODING_ERROR	6106	X'000017DA'
MQRC_STRUC_ID_ERROR	6107	X'000017DB'
Ukazatel MQRC_NULL_POINTER	6108	X'000017DC'
MQRC_NO_CONNECTION_REFERENCE	6109	X'000017DD'
MQRC_NO_BUFFER	6110	X'000017DE'
MQRC_BINARY_DATA_LENGTH_ERROR	6111	X'000017DF'
MQRC_BUFFER_NOT_AUTOMATIC	6112	X'000017E0'
MQRC_INNEDOSTATEČNÁ_VYROVNÁVACÍ paměť	6113	X'000017E1'
MQRC_INDOSTATEČNÁ_DATA	6114	X'000017E2'
MQRC_DATA_TRUNCATED	6115	X'000017E3'
MQRC_ZERO_LENGTH	6116	X'000017E4'
MQRC_NEGATIVE_LENGTH	6117	X'000017E5'
MQRC_NEGATIVE_OFFSET	6118	X'000017E6'

Tabulka 304. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRC_NEKONZISTENTNÍ_FORMÁT	6119	X'000017E7'
MQRC_INCONSISTENT_OBJECT_STATE	6120	X'000017E8'
MQRC_CONTEXT_OBJECT_NOT_VALID	6121	X'000017E9'
MQRC_CONTEXT_OPEN_ERROR	6122	X'000017EA'
MQRC_STRUC_LENGTH_ERROR	6123	X'000017EB'
MQRC_NOT_CONNECTED	6124	X'000017EC'
MQRC_NOT_OPEN	6125	X'000017ED'
MQRC_DISTRIBUTION_LIST_EMPTY	6126	X'000017EE'
MQRC_INCONSISTENT_OPEN_OPTIONS	6127	X'000017EF'
MQRC_WRONG_VERSION	6128	X'000017F0'
MQRC_REFERENCE_ERROR	6129	X'000017F1'

MQRCCF_ * (kódy příčiny záhlaví formátu příkazu)

Další informace o odezvě programátora naleznete v tématu Kódy příčiny PCF.

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

Tabulka 305. Hodnoty konstant (pokračování)

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

Tabulka 305. Hodnoty konstant (pokračování)

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

Tabulka 305. Hodnoty konstant (pokračování)

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

Tabulka 305. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRCCF_FUNCTION_RESTRICTED	3227	X'00000C9B'
MQRCCF_PARM_MISSING	3228	X'00000C9C'
MQRCCF_PARM_VALUE_ERROR	3229	X'00000C9D'
MQRCCF_COMMAND_LENGTH_ERROR	3230	X'00000C9E'
MQRCCF_COMMAND_ORIGIN_ERROR	3231	X'00000C9F'
MQRCCF_LISTENER_CONFLICT	3232	X'00000CA0'
MQRCCF_LISTENER_STARTED	3233	X'00000CA1'
MQRCCF_LISTENER_ZASTAVENO	3234	X'00000CA2'
MQRCCF_CHANNEL_ERROR	3235	X'00000CA3'
MQRCCF_CF_STRUC_ERROR	3236	X'00000CA4'
MQRCCF_UNKNOWN_USER_ID	3237	X'00000CA5'
MQRCCF_UNEXPECTED_ERROR	3238	X'00000CA6'
MQRCCF_NO_XCF_PARTNER	3239	X'00000CA7'
MQRCCF_CFGR_PARM_ID_ERROR	3240	X'00000CA8'
MQRCCF_CFIF_LENGTH_ERROR	3241	X'00000CA9'
MQRCCF_CFIF_OPERATOR_ERROR	3242	X'00000CAA'
MQRCCF_CFIF_PARM_ID_ERROR	3243	X'00000CAB'
MQRCCF_CFSF_FILTER_VAL_LEN_ERR	3244	X'00000CAC'
MQRCCF_CFSF_LENGTH_ERROR	3245	X'00000CAD'
MQRCCF_CFSF_OPERATOR_ERROR	3246	X'00000CAE'
MQRCCF_CFSF_PARM_ID_ERROR	3247	X'00000CAF'
MQRCCF_TOO_MANY_FILTERS	3248	X'00000CB0'
MQRCCF_LISTENER_RUNNING	3249	X'00000CB1'
MQRCCF_LSTR_STATUS_NOT_FOUND	3250	X'00000CB2'
MQRCCF_SERVICE_RUNNING	3251	X'00000CB3'
MQRCCF_SERV_STATUS_NOT_FOUND	3252	X'00000CB4'
MQRCCF_SERVICE_ZASTAVENO	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'

Tabulka 305. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRCCF_CFBF_OPERATOR_ERROR	3266	X'00000CC2'
MQRCCF_CFBF_FILTER_VAL_LEN_ERR	3267	X'00000CC3'
MQRCCF_LISTENER_STILL_ACTIVE	3268	X'00000CC4'
MQRCCF_DEF_XMIT_Q_CLUS_ERROR	3269	X'0000CC5'
MQRCCF_TOPICSTR_ALREADY_EXISTS	3300	X'00000CE4'
MQRCCF_SHARING_CONVS_ERROR	3301	X'00000CE5'
MQRCCF_SHARING_CONVS_TYPE	3302	X'00000CE6'
MQRCCF_SECURITY_CASE_CONFLICT	3303	X'00000CE7'
MQRCCF_TOPIC_TYPE_ERROR	3305	X'00000CE9'
MQRCCF_MAX_INSTANCES_ERROR	3306	X'00000CEA'
MQRCCF_MAX_INSTS_PER_CLNT_ERR	3307	X'00000CEB'
MQRCCF_TOPIC_STRING_NOT_FOUND	3308	X'00000CEC'
MQRCCF_SUBSCRIPTION_POINT_ERR	3309	X'00000CED'
MQRCCF_SUB_ALREADY_EXISTS	3311	X'00000CEF'
MQRCCF_UNKNOWN_OBJECT_NAME	3312	X'00000CF0'
MQRCCF_REMOTE_Q_NAME_ERROR	3313	X'00000CF1'
MQRCCF_DURABILITY_NOT_ALLOWED	3314	X'00000CF2'
MQRCCF_HOBJ_ERROR	3315	X'00000CF3'
MQRCCF_DEST_NAME_ERROR	3316	X'00000CF4'
MQRCCF_INVALID_DESTINATION	3317	X'00000CF5'
MQRCCF_PUBSUB_INHIBITED	3318	X'00000CF6'
MQRCCF_CHLAUTH_TYPE_ERROR	3326	X'00000CFE'
MQRCCF_CHLAUTH_ACTION_ERROR	3327	X'00000CFF'
MQRCCF_CHLAUTH_USERSRC_ERROR	3335	X'00000D07'
MQRCCF_WRONG_CHLAUTH_TYPE	3336	X'00000D08'
MQRCCF_CHLAUTH_ALREADY_EXISTS	3337	X'00000D09'
MQRCCF_CHLAUTH_NOT_FOUND	3338	X'00000D0A'
MQRCCF_WRONG_CHLAUTH_ACTION	3339	X'00000D0B'
MQRCCF_WRONG_CHLAUTH_USERSRC	3340	X'00000D0C'
MQRCCF_CHLAUTH_WARN_ERROR	3341	X'00000D0D'
MQRCCF_WRONG_CHLAUTH_MATCH	3342	X'00000D0E'
MQRCCF_IPADDR_RANGE_CONFLICT	3343	X'00000D0F'
MQRCCF_CHLAUTH_MAX_PŘEKROČEN	3344	X'00000D10'
MQRCCF_IPADDR_ERROR	3345	X'00000D11'
MQRCCF_IPADDR_RANGE_ERROR	3346	X'00000D12'
MQRCCF_PROFILE_NAME_MISSING	3347	X'00000D13'
MQRCCF_CHLAUTH_CLNTUSER_ERROR	3348	X'00000D14'
MQRCCF_CHLAUTH_NAME_ERROR	3349	X'00000D15'
MQRCCF_SUITE_B_ERROR	3353	X'00000D19'

Tabulka 305. Hodnoty konstant (pokračování)

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

Tabulka 305. Hodnoty konstant (pokračování)

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

Tabulka 305. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRCCF_CHAD_WRONG_TYPE	4080	X'00000FF0'
MQRCCF_CHAD_EVENT_ERROR	4081	X'00000FF1'
MQRCCF_CHAD_EVENT_WRONG_TYPE	4082	X'00000FF2'
MQRCCF_CHAD_EXIT_ERROR	4083	X'00000FF3'
MQRCCF_CHAD_EXIT_WRONG_TYPE	4084	X'00000FF4'
MQRCCF_SUPPRESSED_BY_EXIT	4085	X'00000FF5'
MQRCCF_BATCH_INT_ERROR	4086	X'00000FF6'
MQRCCF_BATCH_INT_WRONG_TYPE	4087	X'00000FF7'
MQRCCF_NET_PRIORITY_ERROR	4088	X'00000FF8'
MQRCCF_NET_PRIORITY_WRONG_TYPE	4089	X'00000FF9'
MQRCCF_CHANNEL_CLOSED	4090	X'00000FFA'
MQRCCF_Q_STATUS_NOT_FOUND	4091	X'00000FFB'
MQRCCF_SSL_CIPHER_SPEC_ERROR	4092	X'00000FFC'
MQRCCF_SSL_PEER_NAME_ERROR	4093	X'00000FFD'
MQRCCF_SSL_CLIENT_AUTH_ERROR	4094	X'00000FFE'
MQRCCF_RETAINED_NOT_SUPPORTED	4095	X'00000FFF'
z/OS MQRCCF_KWD_VALUE_WRONG_TYPE	4096	X'00001000'

MQRCN_ * (Konstanty opětovného připojení klienta)

Tabulka 306. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRCN_NO	0	X'00000000'
MQRCN_YES	1	X'00000001'
MQRCN_Q_MGR	2	X'00000002'
MQRCN_DISABLED	3	X'00000003'

MQRCVTIME_ * (typy časového limitu příjmu)

Tabulka 307. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRCVTIME_MULTIPLY	0	X'00000000'
MQRCVTIME_ADD	1	X'00000001'
MQRCVTIME_EQUAL-další informace	2	X'00000002'

MQREADA_ * (Hodnoty dopředné čtení)

Tabulka 308. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQREADA_NO	0	X'00000000'
MQREADA_YES	1	X'00000001'
MQREADA_DISABLED	2	X'00000002'

Tabulka 308. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQREADA_INHIBITED	3	X'00000003'
MQREADA_BACKLOG	4	X'00000004'

MQRECORDING_ * (Volby záznamu)

Tabulka 309. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRECORDING_DISABLED	0	X'00000000'
MQRECORDING_Q	1	X'00000001'
MQRECORDING_MSG	2	X'00000002'

MQREGO_ * (Volby registrace publikování/odběru)

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

MQRFH_ * (Pravidla a formátování struktury záhlaví a příznaků)

Pravidla a struktura záhlaví formátování

Tabulka 311. Struktury konstant	
Název	**Struktura**
MQRFH_STRUC_ID	"RFH¬"
MQRFH_STRUC_ID_ARRAY	'R','F','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 312. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'

Pravidla a formátování příznaků záhlaví

Tabulka 313. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRFH_NONE	0	X'00000000'
MQRFH_NO_FLAGS	0	X'00000000'

MQRFH2_* (Značka voleb publikování/odběru RFH2 Značky složky nejvyšší úrovně)

Tabulka 314. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRFH2_NAME_VALUE_VERSION	1	X'00000001'

MQRFH2_* (Názvy značek voleb publikování/odběru)

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

MQRFH2_* (Názvy značek XML značek voleb publikování/odběru)

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_ * (Vrácená délka)

Tabulka 315. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQRL_UNDEFINED	-1	X'FFFFFFFF'

MQRMH_ * (Struktura záhlaví referenční zprávy)

Tabulka 316. Struktury konstant

Název	Struktura
MQRMH_STRUC_ID	"RMH¬"
MQRMH_STRUC_ID_ARRAY	'R','M','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 317. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQRMH_VERSION_1	1	X'00000001'
MQRMH_CURRENT_VERSION	1	X'00000001'

MQRMHF_ * (Příznaky záhlaví referenční zprávy)

Tabulka 318. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQRMHF_LAST	1	X'00000001'
MQRMHF_NOT_LAST	0	X'00000000'

MQRO_ * (Volby sestavy)

Tabulka 319. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
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'

Tabulka 319. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRO_NAN	2	X'00000002'
MQRO_ACTIVITY	4	X'00000004'
MQRO_NEW_MSG_ID	0	X'00000000'
MQRO_PASS_MSG_ID	128	X'00000080'
MQRO_COPY_MSG_ID_TO_CORREL_ID	0	X'00000000'
MQRO_PASS_CORREL_ID	64	X'00000040'
MQRO_DEAD_LETTER_Q	0	X'00000000'
MQRO_DISCARD_MSG	134217728	X'08000000'
MQRO_PASS_DISCARD_AND_EXPIRAČNÍ	16384	X'00004000'
MQRO_NONE	0	X'00000000'

*MQRO_ * (masky voleb sestavy)*

Tabulka 320. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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)

Max aktivit trasování trasy (MQIACF_MAX_ACTIVITY)

Tabulka 321. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQROUTE_UNLIMITED_ACTIVITY	0	X'00000000'

Podrobnosti trasovací trasy (MQIACF_ROUTE_DETAIL)

Tabulka 322. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQROUTE_DETAIL_LOW	2	X'00000002'
MQROUTE_DETAIL_MEDIUM	8	X'00000008'
MQROUTE_DETAIL_HIGH	32	X'00000020'

Směrování trasovací trasy (MQIACF_ROUTE_FORWARDING)

Tabulka 323. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQROUTE_FORWARD_ALL	256	X'00000100'
MQROUTE_FORWARD_IF_SUPPORTED	512	X'00000200'
MQROUTE_FORWARD_REJ_UNSUP_MASK	-65536	X'FFFF0000'

Doručení trasovací trasy (MQIACF_ROUTE_DELIVERY)

Tabulka 324. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQROUTE_DORUČENÍ_ANO	4096	X'00001000'
MQROUTE_DELIVER_NO	8192	X'00002000'
MASKA MQROUTE_DELIVER_REJ_UNSUP_MASK	-65536	X'FFFF0000'

Akumulace trasování trasy (MQIACF_ROUTE_AKUMULACE)

Tabulka 325. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQROUTE_AKUMULACE_NONE	65539	X'00010003'
MQROUTE_AKUMULATE_IN_MSG	65540	X'00010004'
MQROUTE_AKUMULATE_AND_REPLY	65541	X'00010005'

MQRP_ * (Volby nahrazení formátu příkazu)

Tabulka 326. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQRP_YES	1	X'00000001'
MQRP_NO	0	X'00000000'

MQRQ_ * (Formát příkazu Kvalifikátory příčiny)

Tabulka 327. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQRQ_CONN_NOT_AUTHORIZED	1	X'00000001'
MQRQ_OPEN_NOT_AUTHORIZED	2	X'00000002'
MQRQ_CLOSE_NOT_AUTORIZOVANÝ	3	X'00000003'
MQRQ_CMD_NOT_AUTORIZOVÁNO	4	X'00000004'
MQRQ_Q_MGR_ZASTAVENÍ	5	X'00000005'
MQRQ_Q_MGR_QUIESCING	6	X'00000006'
MQRQ_CHANNEL_STOPPED_OK	7	X'00000007'
MQRQ_CHANNEL_STOPPED_ERROR	8	X'00000008'
MQRQ_CHANNEL_STOPPED_RETRY	9	X'00000009'
MQRQ_CHANNEL_STOPPED_DISABLED	10	X'0000000A'
MQRQ_BRIDGE_STOPPED_OK	11	X'0000000B'
MQRQ_BRIDGE_STOPPED_ERROR	12	X'0000000C'
MQRQ_SSL_HANDSHAKE_ERROR	13	X'0000000D'
MQRQ_SSL_CIPHER_SPEC_ERROR	14	X'0000000E'
MQRQ_SSL_CLIENT_AUTH_ERROR	15	X'0000000F'
CHYBA MQRQ_SSL_PEER_NAME_ERROR	16	X'00000010'
MQRQ_SUB_NOT_AUTORIZOVÁNO	17	X'00000011'
MQRQ_SUB_DEST_NOT_AUTHORIZED	18	X'00000012'

Tabulka 327. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRQ_SSL_UNKNOWN_REVOCATION	19	X'00000013'
MQRQ_SYS_CONN_NOT_AUTHORIZED	20	X'00000014'
MQRQ_CHANNEL_BLOCKED_ADDRESS	21	X'00000015'
MQRQ_CHANNEL_BLOCKED_USERID	22	X'00000016'
MQRQ_CHANNEL_BLOCKED_NOACCESS	23	X'00000017'
MQRQ_MAX_ACTIVE_CHANNELS	24	X'00000018'
MQRQ_MAX_CHANNELS	25	X'00000019'
MQRQ_SVRCONN_INST_LIMIT	26	X'0000001A'
MQRQ_CLIENT_INST_LIMIT!	27	X'0000001B'
MQRQ_CAF_NOT_INSTALLED	28	X'0000001C'
MQRQ_CSP_NOT_AUTHORIZED	29	X'0000001D'
MQRQ_FAILOVER_POVOLENÉ	30	X'0000001E'
MQRQ_FAILOVER_NOT_POVOLENÉ	31	X'0000001F'
MQRQ_STANDBY_ACTIVATED	32	X'00000020'
MQRQ_REPLICA_ACTIVATED	33	X'00000021'

MQRT_ * (Typ aktualizace formátu příkazu)

Tabulka 328. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRT_CONFIGURATION	1	X'00000001'
MQRT_VYPRŠENÍ	2	X'00000002'
MQRT_NSPROC	3	X'00000003'
MQRT_PROXYSUB	4	X'00000004'

MQRU_ * (pouze požadavek)

Tabulka 329. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRU_PUBLISH_ON_REQUEST	1	X'00000001'
MQRU_PUBLISH_ALL	2	X'00000002'

MQSCA_ * (Ověření klienta TLS)

Tabulka 330. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCA_REQUIRED	0	X'00000000'
MQSCA_OPTIONAL	1	X'00000001'

MQSCO_ * (volby konfigurace TLS)

Struktura voleb konfigurace TLS

Tabulka 331. Struktury konstant	
Název	**Struktura**
MQSCO_STRUC_ID	"SCO¬"
MQSCO_STRUC_ID_ARRAY	'S','C','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 332. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCO_VERSION_1	1	X'00000001'
MQSCO_VERSION_2	2	X'00000002'
MQSCO_VERSION_3	3	X'00000003'
MQSCO_VERSION_4	4	X'00000004'
MQSCO_CURRENT_VERSION	4	X'00000004'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Počet resetování klíče voleb konfigurace TLS

Tabulka 333. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCO_RESET_COUNT_DEFAULT	0	X'00000000'

Obor definice fronty formátu příkazu

Tabulka 334. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCO_Q_MGR	1	X'00000001'
MQSCO_CELL	2	X'00000002'

MQSCOPE_ * (Rozsah publikování)

Tabulka 335. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCOPE_ALL-všechny	0	X'00000000'
MQSCOPE_AS_PARENT	1	X'00000001'
MQSCOPE_QMGR	4	X'00000004'

MQSCYC_ * (případ zabezpečení)

Tabulka 336. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCYC_UPPER	0	X'00000000'
MQSCYC_MIXED	1	X'00000001'

MQSD_ * (Struktura deskriptoru objektu)

Tabulka 337. Názvy konstant a struktury	
Název	**Struktura**
ID_STRUC_MQSD_STRUC_ID	"SD¬¬"
MQSD_STRUC_ID_POLE	'S','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 338. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSD_VERSION_1	1	X'00000001'
MQSD_CURRENT_VERSION	1	X'00000001'

MQSECITEM_ * (Položky zabezpečení formátu příkazu)

Tabulka 339. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECITEM_ALL	0	X'00000000'
MQSECITEM_MQADMIN	1	X'00000001'
MQSECITEM_MQNLIST	2	X'00000002'
MQSECITEM_MQPROC	3	X'00000003'
MQSECITEM_MQQUEUE	4	X'00000004'
MQSECITEM_MQCONN	5	X'00000005'
MQSECITEM_MQCMDS	6	X'00000006'
MQSECITEM_MXADMIN	7	X'00000007'
MQSECITEM_MXNLIST	8	X'00000008'
MQSECITEM_MXPROC	9	X'00000009'
MQSECITEM_MXQUEUE	10	X'0000000A'
MQSECITEM_MXTOPIC	11	X'0000000B'

MQSECPROT_ * (typy protokolu zabezpečení)

Tabulka 340. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECPROT_NONE	0	X'00000000'
MQSECPROT_SSLV30	1	X'00000001'
MQSECPROT_TLSV10	2	X'00000002'
MQSECPROT_TLSV12	4	X'00000004'

MQSECSW_ * (formát příkazů Přepínače zabezpečení a stavy přepínačů)

Formát příkazu Přepínače zabezpečení

Tabulka 341. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECSW_PROCESS	1	X'00000001'

Tabulka 341. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECSW_NAMELIST	2	X'00000002'
MQSECSW_Q	3	X'00000003'
MQSECSW_TOPIC	4	X'00000004'
MQSECSW_CONTEXT	6	X'00000006'
MQSECSW_ALTERNATE_USER	7	X'00000007'
MQSECSW_COMMAND	8	X'00000008'
MQSECSW_CONNECTION	9	X'00000009'
MQSECSW_SUBSYSTEM	10	X'0000000A'
MQSECSW_COMMAND_RESOURCES	11	X'0000000B'
MQSECSW_Q_MGR	15	X'0000000F'
MQSECSW_QSG	16	X'00000010'

Formát příkazu Stav přepínače zabezpečení

Tabulka 342. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECSW_OFF_FOUND	21	X'00000015'
MQSECSW_ON_FOUND	22	X'00000016'
MQSECSW_OFF_NOT_FOUND	23	X'00000017'
MQSECSW_ON_NOT_FOUND	24	X'00000018'
MQSECSW_OFF_ERROR	25	X'00000019'
MQSECSW_ON_PŘEPSÁNO	26	X'0000001A'

MQSECTYPE_ * (Typy zabezpečení formátu příkazu)

Tabulka 343. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECTYPE_AUTHSERV	1	X'00000001'
MQSECTYPE_SSL	2	X'00000002'
MQSECTYPE_CLASSES	3	X'00000003'

MQSEG_ * (Segmentace)

Tabulka 344. Názvy konstant a hodnoty	
Název	**Hodnota**
MQSEG_INHIBITED	'¬'
MQSEG_POVOLENO	'A'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQSEL_ * (Speciální hodnoty selektoru)

Tabulka 345. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSEL_ANY_SELEKTOR	-30001	X'FFFF8ACF'
MQSEL_ANY_USER_SELECTOR	-30002	X'FFFF8ACE'
MQSEL_ANY_SYSTEM_SELECTOR	-30003	X'FFFF8ACD'
MQSEL_ALL_SELECTORS	-30001	X'FFFF8ACF'
MQSEL_ALL_USER_SELECTORS	-30002	X'FFFF8ACE'
MQSEL_ALL_SYSTEM_SELECTORS	-30003	X'FFFF8ACD'

MQSELTYPE_ * (Typy selektorů)

Tabulka 346. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSELTYPE_NONE	0	X'00000000'
MQSELTYPE_STANDARD	1	X'00000001'
Rozšířené MQSELTYPE_EXTENDED	2	X'00000002'

MQSID_ * (identifikátor zabezpečení)

Tabulka 347. Názvy konstant a hodnoty	
Název	**Hodnota**
MQSID_NONE	X'00...00' (40 nulových hodnot)
MQSID_NONE_ARRAY	'\0','\0',... (40 nulových hodnot)

MQSIDT_ * (typy identifikátorů zabezpečení)

Tabulka 348. Názvy konstant a hodnoty	
Název	**Hexadecimální hodnota**
MQSIDT_NONE	X'00'
MQSIDT_NT_SECURITY_ID	X'01'
MQSIDT_WAS_SECURITY_ID	X'02'

MQSMPO_ * (Nastavení voleb a struktury vlastností zprávy)

Nastavit strukturu voleb vlastností zprávy

Tabulka 349. Struktury konstant	
Název	**Struktura**
MQSMPO_STRUC_ID	"SMPO"
MQSMPO_STRUC_ID_ARRAY	'S','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 350. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSMPO_VERSION_1	1	X'00000001'

Tabulka 350. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSMPO_CURRENT_VERSION	1	X'00000001'

Nastavit volby vlastností zprávy

Tabulka 351. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSMPO_SET_FIRST	0	X'00000000'
MQSMPO_SET_PROP_UNDER_CURSOR	1	X'00000001'
MQSMPO_SET_PROP_AFTER_CURSOR	2	X'00000002'
MQSMPO_APPEND_PROPERTY	4	X'00000004'
MQSMPO_SET_PROP_BEFORE_CURSOR	8	X'00000008'
MQSMPO_NONE	0	X'00000000'

MQSO_ * (Volby odběru)

Tabulka 352. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSO_NONE	0	X'00000000'
MQSO_NON_TRVALÝ	0	X'00000000'
MQSO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQSO_ALTER	1	X'00000001'
MQSO_CREATE	2	X'00000002'
MQSO_RESUME	4	X'00000004'
MQSO_TRVALÉ	8	X'00000008'
MQSO_GROUP_SUB	16	X'00000010'
MQSO_MANAGED	32	X'00000020'
MQSO_SET_IDENTITY_CONTEXT	64	X'00000040'
MQSO_FIXED_USERID	256	X'00000100'
MQSO_ANY_USERID	512	X'00000200'
MQSO_PUBLICATIONS_ON_REQUEST	2048	X'00000800'
MQSO_NEW_PUBLICATIONS_ONLY	4096	X'00001000'
MQSO_FAIL_IF_QUIESCING	8192	X'00002000'
MQSO_ALTERNATE_USER_AUTHORITY	262144	X'00040000'
MQSO_WILDCARD_CHAR	1048576	X'00100000'
MQSO_WILDCARD_TOPIC	2097152	X'00200000'
MQSO_SET_CORREL_ID	4194304	X'00400000'
MQSO_SCOPE_QMGR	67108864	X'04000000'
MQSO_NO_READ_AHEAD	134217728	X'08000000'
MQSO_READ_AHEAD	268435456	X'10000000'

MQSP_ * (Dostupnost synchronizačního bodu)

Tabulka 353. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQSP_AVAILABLE	1	X'00000001'
MQSP_NOT_AVAILABLE	0	X'00000000'

MQSPL_ * (Volby ochrany zásad zabezpečení)

Tabulka 354. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQSPL_PASSTHRU	0	X'00000000'
MQSPL_REMOVE	1	X'00000001'
MQSPL_AS_POLICY	2	X'00000002'

MQSQQM_ * (Název sdíleného správce front)

Tabulka 355. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQSQQM_USE	0	X'00000000'
MQSQQM_IGNORE	1	X'00000001'

MQSR_ * (Akce)

Tabulka 356. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQSR_ACTION_PUBLIKACE	1	X'00000001'

MQSRO_ * (struktura voleb požadavku na odběr)

Tabulka 357. Struktury konstant

Název	Struktura
MQSRO_STRUC_ID	"SRO¬"
MQSRO_STRUC_ID_POLE	'S','R','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 358. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQSRO_VERSION_1	1	X'00000001'
MQSRO_CURRENT_VERSION	1	X'00000001'
MQSRO_NONE	0	X'00000000'
MQSRO_FAIL_IF_QUIESCING	8192	X'00002000'

MQSS_ * (Stav segmentu)

Tabulka 359. Názvy konstant a struktury

Název	Struktura
MQSS_NOT_A_SEGMENT	'¬'

Tabulka 359. Názvy konstant a struktury (pokračování)	
Název	**Struktura**
MQSS_SEGMENT	'S'
MQSS_LAST_SEGMENT	'L'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQSSL_ * (požadavky TLS FIPS)

Poznámka: V systému AIX, Linux, and Windowsposkytuje produkt IBM MQ kompatibilitu se standardem FIPS 140-2 prostřednictvím šifrovacího modulu IBM Crypto for C (ICC) . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C (ICC) certifikát a měli by si být vědomi všech doporučení poskytnutých NIST. Náhradní modul FIPS 140-3 momentálně probíhá a jeho stav lze zobrazit jeho vyhledáním v modulech NIST CMVP v seznamu procesů.

Tabulka 360. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSSL_FIPS_NO	0	X'00000000'
MQSSL_FIPS_YES	1	X'00000001'

MQSTAT_ * (Volby Stat)

Tabulka 361. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSTAT_TYPE_ASYNC_ERROR	0	X'00000000'
MQSTAT_TYPE_RECONNECTION	0	X'00000000'
MQSTAT_TYPE_RECONNECTION_ERROR	0	X'00000000'

MQSTS_ * (struktura struktury vytváření sestav stavu)

Tabulka 362. Struktury konstant	
Název	**Struktura**
MQSTS_STRUC_ID	"STAT"
MQSTS_STRUC_ID_ARRAY	'S','T','A','T'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 363. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSTS_VERSION_1	1	X'00000001'
MQSTS_CURRENT_VERSION	1	X'00000001'

MQSUB_ * (trvalé odběry)

Trvalé povolené odběry

Tabulka 364. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUB_DURABLE_AS_PARENT	0	X'00000000'
MQSUB_DURABLE_ALLOWED	1	X'00000001'

Tabulka 364. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUB_DURABLE_INHIBITED	2	X'00000002'

Rozsah trvalých odběrů

Tabulka 365. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUB_DURABLE_ALL	-1	X'FFFFFFFF'
MQSUB_DURABLE_YES	1	X'00000001'
MQSUB_DURABLE_NO	2	X'00000002'

MQSUBTYPE_ * (typy odběrů ve formátu příkazu)

Tabulka 366. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUBTYPE_API	1	X'00000001'
MQSUBTYPE_ADMIN	2	X'00000002'
MQSUBTYPE_PROXY	3	X'00000003'
MQSUBTYPE_ALL	-1	X'FFFFFFFF'
MQSUBTYPE_USER	-2	X'FFFFFFFE'

MQSUS_ * (Formát příkazu Pozastavit stav)

Tabulka 367. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUS_YES	1	X'00000001'
MQSUS_NO	0	X'00000000'

MQSVC_ * (Služba)

Typy služeb

Tabulka 368. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSVC_TYPE_COMMAND	0	X'00000000'
MQSVC_TYPE_SERVER	1	X'00000001'

Řízení služeb

Tabulka 369. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSVC_CONTROL_Q_MGR	0	X'00000000'
MQSVC_CONTROL_Q_MGR_START	1	X'00000001'
MQSVC_CONTROL_MANUAL	2	X'00000002'

Stav služby

Tabulka 370. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSVC_STATUS_ZASTAVENO	0	X'00000000'
MQSVC_STATUS_STARTING	1	X'00000001'
MQSVC_STATUS_RUNNING	2	X'00000002'
MQSVC_STATUS_ZASTAVENÍ	3	X'00000003'
MQSVC_STATUS_RETRYING	4	X'00000004'

MQSYNCPOINT_ * (Hodnoty Syncpoint formátu příkazu pro migraci publikování/odběru)

Tabulka 371. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSYNCPOINT_YES	0	X'00000000'
MQSYNCPOINT_IFPER	1	X'00000001'

MQSYSP_ * (Systémové hodnoty parametrů formátu příkazu)

Tabulka 372. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSYSP_NO	0	X'00000000'
MQSYSP_YES	1	X'00000001'
MQSYSP_EXTENDED, rozšířené	2	X'00000002'
MQSYSP_TYPE_INITIAL	10	X'0000000A'
MQSYSP_TYPE_SET	11	X'0000000B'
MQSYSP_TYPE_LOG_COPY	12	X'0000000C'
MQSYSP_TYPE_LOG_STATUS	13	X'0000000D'
MQSYSP_TYPE_ARCHIVE_TAPE	14	X'0000000E'
MQSYSP_ALLOC_BLK	20	X'00000014'
MQSYSP_ALLOC_TRK	21	X'00000015'
MQSYSP_ALLOC_CYL	22	X'00000016'
MQSYSP_STATUS_BUSY	30	X'0000001E'
MQSYSP_STATUS_PREMOUNT	31	X'0000001F'
MQSYSP_STATUS_AVAILABLE	32	X'00000020'
MQSYSP_STATUS_UNKNOWN	33	X'00000021'
MQSYSP_STATUS_ALLOC_ARCHIVE	34	X'00000022'
MQSYSP_STATUS_COPYING_BSDS	35	X'00000023'
MQSYSP_STATUS_COPYING_LOG	36	X'00000024'

MQTA_ * (atributy tématu)

zástupné znaky

Tabulka 373. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQTA_BLOCK	1	X'00000001'
MQTA_PASSTHRU	2	X'00000002'

Povolené odběry

Tabulka 374. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQTA_SUB_AS_PARENT	0	X'00000000'
MQTA_SUB_INHIBITED	1	X'00000001'
MQTA_SUB_ALLOWED	2	X'00000002'

Dílčí šíření serveru proxy

Tabulka 375. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQTA_PROXY_SUB_FORCE	1	X'00000001'
MQTA_PROXY_SUB_FIRSTUSE	2	X'00000002'

Publikování povolena

Tabulka 376. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQTA_PUB_AS_PARENT	0	X'00000000'
MQTA_PUB_INHIBITED	1	X'00000001'
MQTA_PUB_ALLOWED	2	X'00000002'

MQTC_ * (Ovládací prvky spouštěče)

Tabulka 377. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQTC_OFF	0	X'00000000'
MQTC_ON	1	X'00000001'

MQTCPKEEP_ * (udržení aktivity TCP)

Tabulka 378. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQTCPKEEP_NO	0	X'00000000'
MQTCPKEEP_YES	1	X'00000001'

MQTCPSTACK_ * (typy zásobníku TCP)

Tabulka 379. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTCPSTACK_SINGLE	0	X'00000000'
MQTCPSTACK_MULTIPLE	1	X'00000001'

MQTIME_ * (jednotky času formátu příkazu)

Tabulka 380. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTIME_UNIT_MINS	0	X'00000000'
MQTIME_UNIT_SECS	1	X'00000001'

MQTM_ * (Struktura zprávy spouštěče)

Tabulka 381. Struktury konstant	
Název	**Struktura**
ID_STRUC_MQTM_STRUC_ID	"TM¬¬"
MQTM_STRUC_ID_ARRAY	'T','M','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 382. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTM_VERSION_1	1	X'00000001'
MQTM_CURRENT_VERSION	1	X'00000001'

MQTMC_ * (Struktura formátu znaků zprávy spouštěče)

Tabulka 383. Struktury konstant	
Název	**Struktura**
MQTMC_STRUC_ID	"TMC¬"
MQTMC_STRUC_ID_ARRAY	'T','M','C','¬'
MQTMC_VERSION_1	"¬¬¬1"
MQTMC_VERSION_2	"¬¬¬2"
MQTMC_CURRENT_VERSION	"¬¬¬2"
MQTMC_VERSION_1_ARRAY	'¬','¬','¬','1'
MQTMC_VERSION_2_ARRAY	'¬','¬','¬','2'
MQTMC_CURRENT_VERSION_ARRAY	'¬','¬','¬','2'

MQTOPT_ * (Typ tématu)

Tabulka 384. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTOPT_LOCAL-lokální	0	X'00000000'
MQTOPT_CLUSTER	1	X'00000001'
MQTOPT_ALL	2	X'00000002'

MQTRAXSTR_ * (Automatické spuštění trasování inicializátoru kanálu)

Tabulka 385. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTRAXSTR_NO	0	X'00000000'
MQTRAXSTR_YES	1	X'00000001'

MQTSCOPE_ * (Rozsah odběru)

Tabulka 386. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTSCOPE_QMGR	1	X'00000001'
MQTSCOPE_ALL	2	X'00000002'

MQTT_ * (typy spouštěčů)

Tabulka 387. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTT_NONE	0	X'00000000'
MQTT_FIRST	1	X'00000001'
MQTT_EVERY	2	X'00000002'
Hloubka MQTT_DEPTH	3	X'00000003'

MQTYPE_ * (datové typy vlastností)

Tabulka 388. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTYPE_AS_SET	0	X'00000000'
MQTYPE_NULL	2	X'00000002'
MQTYPE_BOOLEAN	4	X'00000004'
MQTYPE_BYTE_STRING	8	X'00000008'
MQTYPE_INT8	16	X'00000010'
MQTYPE_INT16	32	X'00000020'
MQTYPE_INT32	64	X'00000040'
MQTYPE_LONG	64	X'00000040'
MQTYPE_INT64	128	X'00000080'
MQTYPE_FLOAT32	256	X'00000100'
MQTYPE_FLOAT64	512	X'00000200'
MQTYPE_STRING	1024	X'00000400'

MQUA_ * (Selektory atributů uživatele publikování/odběru)

Tabulka 389. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQUA_PRVNÍ	65536	X'00010000'
MQUA_LAST	999999999	X'3B9AC9FF'

MQUIDSUPP_ * (Formát příkazu Podpora ID uživatele)

Tabulka 390. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQUIDSUPP_NO	0	X'00000000'
MQUIDSUPP_YES	1	X'00000001'

MQUNDELIVERED_ * (Nedoručené hodnoty formátu příkazu pro migraci publikování/odběru)

Tabulka 391. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQUNDELIVERED_NORMAL	0	X'00000000'
MQUNDELIVERED_SAFE	1	X'00000001'
MQUNDELIVERED_DISCARD	2	X'00000002'
MQUNDELIVERED_KEEP	3	X'00000003'

MQUOWST_ * (Formát příkazů pro stavy UOW)

Tabulka 392. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQUOWST_NONE	0	X'00000000'
MQUOWST_AKTIVNÍ	1	X'00000001'
PŘIPRAVENÉ_MQUOWST_PŘIPRAVENÉ_	2	X'00000002'
NEVYŘEŠENÉ_MQUOWST_NEVYŘEŠENO	3	X'00000003'

MQUOWT_ * (Typy UOW formátu příkazu)

Tabulka 393. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQUOWT_Q_MGR	0	X'00000000'
MQUOWT_CICS	1	X'00000001'
MQUOWT_RRS	2	X'00000002'
MQUOWT_IMS	3	X'00000003'
MQUOWT_XA	4	X'00000004'

MQUS_ * (Využití front)

Tabulka 394. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQUS_NORMAL	0	X'00000000'
MQUS_PŘENOS	1	X'00000001'

MQUSAGE_ * (Formát příkazu Hodnoty použití sady stránek a Hodnoty použití datové sady)

Formát příkazu Hodnoty použití sady stránek

Tabulka 395. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQUSAGE_PS_AVAILABLE	0	X'00000000'
MQUSAGE_PS_DEFINED	1	X'00000001'
MQUSAGE_PS_OFFLINE	2	X'00000002'
MQUSAGE_PS_NOT_DEFINED	3	X'00000003'
MQUSAGE_PS_POZASTAVENO	4	X'00000004'
MQUSAGE_EXPAND_USER	1	X'00000001'
MQUSAGE_EXPAND_SYSTEM	2	X'00000002'
MQUSAGE_EXPAND_NONE	3	X'00000003'

Formát příkazu Hodnoty použití datové sady

Tabulka 396. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (Délka hodnoty)

Tabulka 397. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQVL_NULL_UKONČENO	-1	X'FFFFFFFF'
MQVL_EMPTY_STRING	0	X'00000000'

MQVU_ * (ID uživatele proměnné)

Tabulka 398. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQVU_FIXED_USER	1	X'00000001'
MQVU_ANY_USER	2	X'00000002'

MQWDR_ * (Struktura záznamu cíle uživatelské procedury pracovní zátěže klastru)

Tabulka 399. Struktury konstant	
Název	**Struktura**
MQWDR_STRUC_ID	"WDR¬"
MQWDR_STRUC_ID_ARRAY	'W','D','R','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 400. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWDR_VERSION_1	1	X'00000001'
MQWDR_VERSION_2	2	X'00000002'
MQWDR_CURRENT_VERSION	2	X'00000002'
MQWDR_LENGTH_1	124	X'0000007C'
MQWDR_LENGTH_2	136	X'00000088'
MQWDR_CURRENT_LENGTH	136	X'00000088'

MQWI_ * (interval čekání)

Tabulka 401. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWI_UNLIMITED	-1	X'FFFFFFFF'

MQWIH_ * (Struktura záhlaví a příznaky informací o pracovní zátěži)

Struktura záhlaví informací o pracovní zátěži

Tabulka 402. Struktury konstant	
Název	**Struktura**
MQWIH_STRUC_ID	"WIH¬"
MQWIH_STRUC_ID_ARRAY	'W','I','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 403. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWIH_VERSION_1	1	X'00000001'
MQWIH_CURRENT_VERSION	1	X'00000001'
MQWIH_LENGTH_1	120	X'00000078'
MQWIH_CURRENT_LENGTH	120	X'00000078'

Příznaky záhlaví informací o pracovní zátěži

Tabulka 404. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWIH_NONE	0	X'00000000'

MQWQR_ * (struktura záznamu uživatelské procedury pracovní zátěže klastru)

Tabulka 405. Struktury konstant	
Název	**Struktura**
MQWQR_STRUC_ID	"WQR¬"
MQWQR_STRUC_ID_ARRAY	'W','Q','R','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 406. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWQR_VERSION_1	1	X'00000001'
MQWQR_VERSION_2	2	X'00000002'
MQWQR_VERSION_3	3	X'00000003'
MQWQR_CURRENT_VERSION	3	X'00000003'
MQWQR_LENGTH_1	200	X'000000C8'
MQWQR_LENGTH_2	208	X'000000D0'
MQWQR_LENGTH_3	212	X'000000D4'
MQWQR_CURRENT_LENGTH	212	X'000000D4'

MQWS_ * (schéma se zástupnými znaky)

Tabulka 407. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝCHOZÍ	0	X'00000000'
MQWS_CHAR	1	X'00000001'
MQWS_TOPIC	2	X'00000002'

MQWXP_ * (struktura parametrů uživatelské procedury pracovní zátěže klastru)

MQWXP_ * (struktura parametrů uživatelské procedury pracovní zátěže klastru)

Tabulka 408. Struktury konstant	
Název	**Struktura**
MQWXP_STRUC_ID	"WXP¬"
MQWXP_STRUC_ID_ARRAY	'W','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 409. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (příznaky pracovní zátěže klastru)

Tabulka 410. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWXP_PUT_BY_CLUSTER_CHL	2	X'00000002'

Související odkazy

“Pole v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru” na stránce 1536

Popis polí v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru

MQXACT_ * (Typy volajícího rozhraní API)

Tabulka 411. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQXACT_EXTERNÍ	1	X'00000001'
MQXACT_INTERNÍ	2	X'00000002'

MQXC_ * (Příkazy pro ukončení)

Tabulka 412. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
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_ * (Odezva ukončení)

Tabulka 413. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQXCC_OK	0	X'00000000'
MQXCC_SUPPRESS_FUNCTION	-1	X'FFFFFFFF'
Funkce MQXCC_SKIP_FUNCTION	-2	X'FFFFFFFE'
MQXCC_SEND_AND_REQUEST_SEC_MSG	-3	X'FFFFFFFD'
MQXCC_SEND_SEC_MSG	-4	X'FFFFFFFC'
MQXCC_SUPPRESS_EXIT	-5	X'FFFFFFFB'
MQXCC_UZAVŘENÝ_KANÁL	-6	X'FFFFFFFA'
MQXCC_REQUEST_ACK	-7	X'FFFFFFF9'
MQXCC_FAILED	-8	X'FFFFFFF8'

MQXDR_ * (Odezva ukončení)

Tabulka 414. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQXDR_OK	0	X'00000000'
MQXDR_CONVERSION_FAILED	1	X'00000001'

MQXE_ * (prostředí)

Tabulka 415. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQXE_OTHER	0	X'00000000'
MQXE_MCA	1	X'00000001'
MQXE_MCA_SVRCONN	2	X'00000002'
MQXE_COMMAND_SERVER	3	X'00000003'
MQXE_MQSC	4	X'00000004'

MQXEPO_ * (struktura voleb vstupního bodu registru a volby ukončení)

Struktura voleb vstupního bodu registru

Tabulka 416. Struktury konstant

Název	Struktura
MQXEPO_STRUC_ID	"XEPO"
MQXEPO_STRUC_ID_POLE	'X','E','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 417. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQXEPO_VERSION_1	1	X'00000001'
MQXEPO_CURRENT_VERSION	1	X'00000001'

Volby ukončení

Tabulka 418. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQXEPO_NONE	0	X'00000000'

MQXF_ * (identifikátory funkcí rozhraní API)

Tabulka 419. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
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'

Tabulka 419. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
Zpětné volání MQXF_CALLBACK	21	X'00000015'
MQXF_SUB	22	X'00000016'
MQXF_SUBRQ	23	X'00000017'
MQXF_XACLOSE	24	X'00000018'
MQXF_XACOMMIT	25	X'00000019'
MQXF_XACOMPLETE	26	X'0000001A'
MQXF_XAEND	27	X'0000001B'
MQXF_XAFORGET	28	X'0000001C'
MQXF_XAOPEN	29	X'0000001D'
MQXF_XAPREPARE	30	X'0000001E'
MQXF_XARECOVER	31	X'0000001F'
MQXF_XAROLLBACK	32	X'00000020'
MQXF_XASTART	33	X'00000021'
MQXF_AXREG	34	X'00000022'
MQXF_AXUNREG	35	X'00000023'

MQXP_ * (struktura parametru křížové uživatelské procedury rozhraní API)

Tabulka 420. Struktury konstant	
Název	**Struktura**
MQXP_STRUC_ID	"XP¬¬"
MQXP_STRUC_ID_ARRAY	'X','P','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 421. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXP_VERSION_1	1	X'00000001'

MQXPDA_ * (oblast určování problémů)

Tabulka 422. Názvy konstant a hodnoty	
Název	**Hodnota**
MQXPDA_NONE	X'00...00' (48 nulových hodnot)
MQXPDA_NONE_ARRAY	'\0','\0',... (48 nulových hodnot)

MQXPT_ * (Typy přenosu)

Tabulka 423. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
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_ * (struktura záhlaví přenosové fronty)

Tabulka 424. Struktury konstant

Název	Struktura
MQXQH_STRUC_ID	"XQH¬"
MQXQH_STRUC_ID_ARRAY	'X','Q','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 425. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQXQH_VERSION_1	1	X'00000001'
MQXQH_CURRENT_VERSION	1	X'00000001'

MQXR_ * (důvody ukončení)

Tabulka 426. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQXR_BEFORE (předchozí)	1	X'00000001'
MQXR_AFTER	2	X'00000002'
MQXR_CONNECTION	3	X'00000003'
MQXR_INIT	11	X'0000000B'
MQXR_TERM	12	X'0000000C'
MQXR_MSG	13	X'0000000D'
MQXR_XMIT	14	X'0000000E'
MQXR_SEC_MSG	15	X'0000000F'
MQXR_INIT_SEC	16	X'00000010'
MQXR_RETRY	17	X'00000011'
MQXR_AUTO_CLUSSDR	18	X'00000012'
MQXR_AUTO_RECEIVER	19	X'00000013'
MQXR_CLWL_OPEN	20	X'00000014'
MQXR_CLWL_PUT	21	X'00000015'
MQXR_CLWL_MOVE	22	X'00000016'

Tabulka 426. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXR_CLWL_REPOS	23	X'00000017'
MQXR_CLWL_REPOS_MOVE	24	X'00000018'
MQXR_END_BATCH	25	X'00000019'
MQXR_ACK_RECEXX_ENCODE_CASE_ONE propojení	26	X'0000001A'
MQXR_AUTO_SVRCONN	27	X'0000001B'
MQXR_AUTO_CLUSRCVR	28	X'0000001C'
MQXR_SEC_PARMS	29	X'0000001D'

MQXR2_* (Odezva ukončení 2)

Tabulka 427. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXR2_PUT_WITH_DEF_ACTION	0	X'00000000'
MQXR2_PUT_WITH_DEF_USERID	1	X'00000001'
MQXR2_PUT_WITH_MSG_USERID	2	X'00000002'
MQXR2_USE_AGENT_BUFFER	0	X'00000000'
MQXR2_USE_EXIT_BUFFER	4	X'00000004'
MQXR2_DEFAULT_CONTINUATION	0	X'00000000'
MQXR2_CONTINUE_CHAIN	8	X'00000008'
MQXR2_SUPPRESS_CHAIN	16	X'00000010'
MQXR2_STATIC_CACHE	0	X'00000000'
MQXR2_DYNAMIC_CACHE	32	X'00000020'

MQXT_ * (identifikátory ukončení)

Tabulka 428. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXT_API_CROSSING_EXIT	1	X'00000001'
MQXT_API_EXIT	2	X'00000002'
MQXT_CHANNEL_SEC_EXIT	11	X'0000000B'
MQXT_CHANNEL_MSG_EXIT	12	X'0000000C'
MQXT_CHANNEL_SEND_EXIT	13	X'0000000D'
MQXT_CHANNEL_RCV_EXIT	14	X'0000000E'
MQXT_CHANNEL_MSG_RETRY_EXIT	15	X'0000000F'
MQXT_CHANNEL_AUTO_DEF_EXIT	16	X'00000010'
MQXT_CLUSTER_WORKLOAD_EXIT	20	X'00000014'
MQXT_PUBSUB_ROUTING_EXIT	21	X'00000015'

MQXUA_ * (Hodnota uživatelské oblasti ukončení)

Tabulka 429. Názvy konstant a hodnoty	
Název	**Hodnota**
MQXUA_NONE	X'00...00' (16 nulových hodnot)

Tabulka 429. Názvy konstant a hodnoty (pokračování)	
Název	**Hodnota**
MQXUA_NONE_ARRAY	`'\0','\0',...` (16 nulových hodnot)

MQXWD_ * (Struktura deskriptoru čekání na ukončení)

Tabulka 430. Struktury konstant	
Název	**Struktura**
MQXWD_STRUC_ID	`"XWD¬"`
MQXWD_STRUC_ID_ARRAY	`'X','W','D','¬'`

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 431. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXWD_VERSION_1	1	`X'00000001'`

MQZAC_ * (Struktura kontextu aplikace)

Tabulka 432. Struktury konstant	
Název	**Struktura**
MQZAC_STRUC_ID	`"ZAC¬"`
MQZAC_STRUC_ID_ARRAY	`'Z','A','C','¬'`

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 433. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZAC_VERSION_1	1	`X'00000001'`
MQZAC_CURRENT_VERSION	1	`X'00000001'`

MQZAD_ * (struktura dat oprávnění)

Tabulka 434. Struktury konstant	
Název	**Struktura**
MQZAD_STRUC_ID	`"ZAD¬"`
MQZAD_STRUC_ID_ARRAY	`'Z','A','D','¬'`

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 435. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZAD_VERSION_1	1	`X'00000001'`
MQZAD_VERSION_2	2	`X'00000002'`
MQZAD_CURRENT_VERSION	2	`X'00000002'`

MQZAET_ * (Typy entit instalovatelných služeb)

Tabulka 436. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQZAET_NONE	0	X'00000000'
MQZAET_PRINCIPAL	1	X'00000001'
MQZAET_GROUP	2	X'00000002'
MQZAET_UNKNOWN (neznámá)	3	X'00000003'

MQZAO_ * (Oprávnění instalovatelných služeb)

Tabulka 437. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQZAO_CONNECT	1	X'00000001'
MQZAO_BROWSE	2	X'00000002'
MQZAO_INPUT	4	X'00000004'
MQZAO_OUTPUT	8	X'00000008'
MQZAO_INQUIRE	16	X'00000010'
MQZAO_SET	32	X'00000020'
MQZAO_PASS_IDENTITY_CONTEXT	64	X'00000040'
MQZAO_PASS_ALL_CONTEXT	128	X'00000080'
MQZAO_SET_IDENTITY_CONTEXT	256	X'00000100'
MQZAO_SET_ALL_CONTEXT	512	X'00000200'
OPRÁVNĚNÍ uživatele MQZAO_ALTERNATE_USER_AUTHORITY	1024	X'00000400'
MQZAO_PUBLISH	2048	X'00000800'
MQZAO_SUBSCRIBE	4096	X'00001000'
MQZAO_RESUME	8192	X'00002000'
MQZAO_ALL_MQI	16383	X'00003FFF'
MQZAO_CREATE	65536	X'00010000'
MQZAO_DELETE	131072	X'00020000'
MQZAO_DISPLAY	262144	X'00040000'
MQZAO_CHANGE	524288	X'00080000'
MQZAO_CLEAR	1048576	X'00100000'
MQZAO_CONTROL	2097152	X'00200000'
MQZAO_CONTROL_EXTENDED	4194304	X'00400000'
MQZAO_AUTHORIZE	8388608	X'00800000'
MQZAO_ALL_ADMIN	16646144	X'00FE0000'
MQZAO_ALL	16662527	X'00FE3FFF'
MQZAO_REMOVE	16777216	X'01000000'
MQZAO_NONE	0	X'00000000'

MQZAS_ * (Verze rozhraní služby instalovatelných služeb)

Tabulka 438. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
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_ * (Typy ověřování)

Tabulka 439. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQZAT_INITIAL_CONTEXT	0	X'00000000'
MQZAT_CHANGE_CONTEXT	1	X'00000001'

MQZCI_ * (indikátor pokračování instalovatelných služeb)

Tabulka 440. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
VÝCHOZÍ	0	X'00000000'
MQZCI_CONTINUE	0	X'00000000'
MQZCI_STOP	1	X'00000001'

MQZED_ * (datová struktura entity)

Tabulka 441. Struktury konstant

Název	Struktura
MQZED_STRUC_ID	"ZED¬"
MQZED_STRUC_ID_ARRAY	'Z','E','D','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 442. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQZED_VERSION_1	1	X'00000001'
MQZED_VERSION_2	2	X'00000002'
MQZED_CURRENT_VERSION	2	X'00000002'

MQZFP_ * (struktura volných parametrů)

Tabulka 443. Struktury konstant

Název	Struktura
MQZFP_STRUC_ID	"ZFP¬"
MQZFP_STRUC_ID_ARRAY	'Z','F','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 444. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZFP_VERSION_1	1	X'00000001'
MQZFP_CURRENT_VERSION	1	X'00000001'

MQZIC_ * (Struktura kontextu identity)

Tabulka 445. Struktury konstant	
Název	**Struktura**
MQZIC_STRUC_ID	"ZIC¬"
MQZIC_STRUC_ID_ARRAY	'Z','I','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 446. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZIC_VERSION_1	1	X'00000001'
MQZIC_CURRENT_VERSION	1	X'00000001'

MQZID_ * (ID funkcí pro služby)

ID funkcí společná pro všechny služby

Tabulka 447. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_INIT	0	X'00000000'
MQZID_TERM	1	X'00000001'

ID funkcí pro službu Oprávnění

Tabulka 448. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_INIT_AUTHORITY	0	X'00000000'
MQZID_TERM_AUTHORITY	1	X'00000001'
OPRÁVNĚNÍ MQZID_CHECK_AUTHORITY	2	X'00000002'
OPRÁVNĚNÍ 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'
Oprávnění MQZID_GET_EXPLICIT_AUTHORITY	7	X'00000007'
MQZID_REFRESH_CACHE	8	X'00000008'
MQZID_ENUMERATE_AUTHORITY_DATA	9	X'00000009'
MQZID_AUTHENTICATE_USER	10	X'0000000A'
MQZID_FREE_USER	11	X'0000000B'
MQZID_INQUIRE	12	X'0000000C'

Tabulka 448. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_CHECK_PRIVILEGED	13	X'0000000D'

ID funkcí pro službu názvů

Tabulka 449. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_INIT_NAME	0	X'00000000'
MQZID_TERM_NAME	1	X'00000001'
MQZID_LOOKUP_NAME	2	X'00000002'
MQZID_INSERT_NAME	3	X'00000003'
MQZID_DELETE_NAME	4	X'00000004'

ID funkcí pro službu Userid

Tabulka 450. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_INIT_USERID	0	X'00000000'
MQZID_TERM_USERID	1	X'00000001'
MQZID_FIND_USERID	2	X'00000002'

MQZIO_ * (Volby inicializace instalovatelných služeb)

Tabulka 451. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZIO_PRIMARY	0	X'00000000'
MQZIO_SECONDARY	1	X'00000001'

MQZNS_ * (verze rozhraní služby názvů)

Tabulka 452. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZNS_VERSION_1	1	X'00000001'

MQZSE_ * (indikátor spuštění instalovatelných služeb-výčet)

Tabulka 453. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZSE_START	1	X'00000001'
MQZSE_CONTINUE	0	X'00000000'

MQZSL_ * (indikátor selektoru instalovatelných služeb)

Tabulka 454. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZSL_NOT_VRÁCENO	0	X'00000000'

Tabulka 454. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZSL_VRÁCENO	1	X'00000001'

MQZTO_ * (Volby ukončení instalovatelných služeb)

Tabulka 455. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZTO_PRIMARY	0	X'00000000'
MQZTO_SECONDARY	1	X'00000001'

MQZUS_ * (Verze rozhraní služby ID uživatele)

Tabulka 456. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZUS_VERSION_1	1	X'00000001'

Datové typy použité v rozhraní MQI

Informace o datových typech, které lze použít v rozhraní MQI (Message Queue Interface). Popisy, pole a deklarace jazyků pro příslušné jazyky s každým datovým typem.

Datové typy a programování pro rozhraní MQI

Představení elementárních a strukturových datových typů a způsobu použití rozhraní MQI prostřednictvím programování v jazyku C, programování v jazyku COBOL nebo programování High Level Assembler .

Základní datové typy

Informace o datových typech používaných v rozhraní MQI nebo ve funkcích ukončení. Ty jsou podrobně popsány, následují příklady, jak deklarovat elementární datové typy v podporovaných programovacích jazycích.

Datové typy používané v rozhraní MQI nebo ve funkcích ukončení jsou následující:

- elementární datové typy nebo
- Agregace elementárních datových typů (polí nebo struktur)

V rozhraní MQI nebo ve funkcích ukončení se používají následující základní datové typy:

Tabulka 457. Názvy, typy a popisy elementárních datových typů		
Název elementárního datového typu	**Datový typ**	**Popis**
MQBOOL	Logická hodnota	Datový typ MQBOOL představuje logickou hodnotu. Hodnota 0 představuje hodnotu false. Jakákoli jiná hodnota představuje hodnotu true. Pro datový typ MQLONG musí být zarovnán objekt MQBOOL.

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQBYTE	bajt	Datový typ MQBYTE představuje jeden bajt dat. Na bajt není umístěna žádná konkrétní interpretace; je považována za řetězec bitů, a ne za binární číslo nebo znak. Není vyžadováno žádné speciální zarovnání. Když jsou data MQBYTE odesílána mezi správci front, kteří používají různé znakové sady nebo kódování, data MQBYTE nejsou žádným způsobem převedena. Pole *MsgId* a *CorrelId* ve struktuře MQMD jsou podobná tomuto. Pole MQBYTE se někdy používá k reprezentaci oblasti hlavní paměti, která není správci front známa. Oblast může například obsahovat data zprávy aplikace nebo strukturu. Zarovnání hranic této oblasti musí být kompatibilní s povahou dat v ní obsažených. V programovacím jazyku C lze pro parametry funkce, které jsou zobrazeny jako pole MQBYTE, použít libovolný datový typ. Je to proto, že tyto parametry jsou vždy předávány adresou a v jazyce C je parametr funkce deklarován jako ukazatel-na-void.

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQBYTEn	Řetězec *n* bajtů	Každý datový typ MQBYTEn představuje řetězec *n* bajtů, kde *n* může mít libovolnou z následujících hodnot: 8, 16, 24, 32, 40 nebo 128. Každý bajt je popsán datovým typem MQBYTE. Není vyžadováno žádné speciální zarovnání. Pokud jsou data v bajtovém řetězci kratší než definovaná délka řetězce, data musí být vyplněna nulami, aby se řetězec vyplnil. Když správce front vrátí aplikaci bajtové řetězce (například ve volání MQGET), jsou pro správce front k dispozici bloky s hodnotami Null v definované délce řetězce. Pojmenované konstanty jsou k dispozici pro definování délek polí bajtového řetězce. Ty jsou uvedeny v části “Konstanty” na stránce 61
MQCHAR	Znak	Datový typ MQCHAR představuje jednobajtový znak nebo jeden bajt dvoubajtového nebo vícebajtového znaku. Není vyžadováno žádné speciální zarovnání. Když jsou data MQCHAR odesílána mezi správci front, kteří používají různé znakové sady nebo kódování, data MQCHAR obvykle vyžadují převod, aby byla data správně interpretována. Správce front to provádí automaticky pro data MQCHAR ve struktuře MQMD. Převod dat MQCHAR v datech zprávy aplikace je řízen volbou MQGMO_CONVERT zadanou ve volání MQGET; další podrobnosti viz popis této volby v části “MQGMO-Volby získání zprávy” na stránce 371 .

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQCHARn	Řetězec *n* znaků	Každý datový typ MQCHARn představuje řetězec *n* znaků, kde *n* může mít libovolnou z následujících hodnot: 4, 8, 12, 20, 28, 32, 48, 64, 128 nebo 256. Každý znak je popsán datovým typem MQCHAR. Není vyžadováno žádné speciální zarovnání. Pokud jsou data v řetězci kratší než definovaná délka řetězce, musí být data vyplněna mezerami, aby se řetězec vyplnil. V některých případech lze k předčasnému ukončení řetězce použít znak null namísto vyplnění mezerami; znak null a následující znaky jsou považovány za mezery, až do definované délky řetězce. Místa, kde lze použít hodnotu null, jsou identifikována v popisech volání a datových typů. Když správce front vrátí aplikaci znakové řetězce (například ve volání MQGET), správce front vždy vyplní znaky s mezerami na definovanou délku řetězce; správce front k oddělení řetězce nepoužívá znak null. K dispozici jsou pojmenované konstanty, které definují délku polí znakového řetězce a jsou uvedeny v části “Konstanty” na stránce 61.
MQFLOAT32	32bitové číslo s pohyblivou řádovou čárkou	Datový typ MQFLOAT32 je 32bitové číslo s pohyblivou řádovou čárkou reprezentované pomocí standardního formátu IEEE s pohyblivou řádovou čárkou. MQFLOAT32 musí být zarovnán na 4bajtové hranici. Použití MQFLOAT32 v jazyku C v systému z/OS vyžaduje použití příznaku kompilátoru FLOAT (IEEE). Použití produktu MQFLOAT32 v jazyce COBOL je omezeno na kompilátory, které podporují čísla s pohyblivou řádovou čárkou ve formátu IEEE. To může vyžadovat použití příznaku kompilátoru FLOAT (NATIVE).

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQFLOAT64	64bitové číslo s pohyblivou řádovou čárkou	Datový typ MQFLOAT64 je 64bitové číslo s pohyblivou řádovou čárkou reprezentované pomocí standardního formátu IEEE s pohyblivou řádovou čárkou. MQFLOAT64 musí být zarovnán na 8bajtové hranici. Použití MQFLOAT64 v jazyku C v systému z/OS vyžaduje použití příznaku kompilátoru FLOAT (IEEE). Použití produktu MQFLOAT64 v jazyku COBOL je omezeno na kompilátory, které podporují čísla s pohyblivou řádovou čárkou ve formátu IEEE. To může vyžadovat použití příznaku kompilátoru FLOAT (NATIVE).
MQHCONFIG	Popisovač konfigurace	Datový typ MQHCONFIG představuje popisovač konfigurace, tj. komponentu, která se konfiguruje pro konkrétní instalovatelnou službu. Popisovač konfigurace musí být zarovnán na své přirozené hranici. Aplikace nesmí spoléhat na formát dat uložených v tomto popisovači. Je-li hodnota platná, je určena k použití v dalších voláních MQI, ale nemá mít kromě tohoto účelu žádný význam.
MQHCONN	Manipulátor připojení	Datový typ MQHCONN představuje manipulátor připojení, tj. připojení ke konkrétnímu správci front. Manipulátor připojení musí být zarovnán na 4bajtové hranici. Aplikace nesmí spoléhat na formát dat uložených v tomto popisovači. Je-li hodnota platná, je určena k použití v dalších voláních MQI, ale nemá mít kromě tohoto účelu žádný význam.

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQHMSG	popisovač zprávy	Datový typ MQHMSG představuje popisovač zprávy, který poskytuje přístup ke zprávě. Popisovač zprávy musí být zarovnán na 8bajtové hranici. Aplikace nesmí spoléhat na formát dat uložených v tomto popisovači. Je-li hodnota platná, je určena k použití v dalších voláních MQI, ale nemá mít kromě tohoto účelu žádný význam.
MQHOBJ	Popisovač objektu	Datový typ MQHOBJ představuje popisovač objektu, který poskytuje přístup k objektu. Popisovač objektu musí být zarovnán na 4bajtové hranici. Aplikace nesmí spoléhat na formát dat uložených v tomto popisovači. Je-li hodnota platná, je určena k použití v dalších voláních MQI, ale nemá mít kromě tohoto účelu žádný význam.
MQINT8	8bitové celé číslo se znaménkem	Datový typ MQINT8 je 8bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -128 až +127, není-li kontext jinak omezen.
MQINT16	16bitové celé číslo se znaménkem	Datový typ MQINT16 je 16bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -32 768 až +32 767, není-li kontext jinak omezen. MQINT16 musí být zarovnán na 2bajtové hranici.
MQINT32	32bitové celé číslo se znaménkem	Datový typ MQINT32 je 32bitové binární celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -2 147 483 648 až + 2 147 483 647, není-li kontext jinak omezen. Viz definice MQLONG.

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQINT64	64bitové celé číslo se znaménkem	Datový typ MQINT64 je 64bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -9 223 372 036 854 775 808 až + 9 223 372 036 854 775 807, pokud kontext nestanoví jinak. Pro COBOL je platný rozsah omezen na -999 999 999 999 999 999 999 999 až +999 999 999 999 999 999 999 999. 64bitové celé číslo musí být zarovnáno na 8bajtovou hranici.
MQLONG	32bitové celé číslo se znaménkem	Datový typ MQLONG je 32bitové binární celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -2 147 483 648 až + 2 147 483 647, není-li kontext jinak omezen. Pro COBOL je platný rozsah omezen na -999 999 999 999 až +999 999 999 999. Objekt MQLONG musí být zarovnán na 4bajtové hranici.
MQPID	Identifikátor procesu	Identifikátor procesu IBM MQ . Jedná se o stejný identifikátor, který se používá v trasování produktu MQ a výpisech paměti produktu FFST™ , ale může se lišit od identifikátoru procesu operačního systému.
MQPTR	Ukazatel	Datový typ MQPTR je adresa dat libovolného typu. Ukazatel musí být zarovnán na své přirozené hranici; jedná se o 16bajtovou hranici v systému IBM ia 8bajtovou hranici v jiných platformách. Některé programovací jazyky podporují typované ukazatele; rozhraní MQI je také používá v několika případech (například PMQCHAR a PMQLONG v programovacím jazyce C).

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQTID	Identifikátor podprocesu	Identifikátor podprocesu IBM MQ . Jedná se o stejný identifikátor použitý v trasování produktu MQ a výpisech paměti produktu FFST™ , ale může se lišit od identifikátoru podprocesu operačního systému.
MQUINT8	8bitové celé číslo bez znaménka	Datový typ MQUINT8 je 8bitové celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu od 0 do +255, není-li kontext jinak omezen.
MQUINT16	16bitové celé číslo bez znaménka	Datový typ MQUINT16 je 16bitové celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu 0 až +65 535, pokud kontext nestanoví jinak. MQUINT16 musí být zarovnán na 2bajtové hranici.
MQUINT32	32bitové celé číslo bez znaménka	Datový typ MQUINT32 je 32bitové binární celé číslo bez znaménka. Viz definice MQULONG.
MQUINT64	64bitové celé číslo bez znaménka	Datový typ MQINT64 je 64bitové celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu od 0 do +18 446 744 073 709 551 615, není-li kontext jinak omezen. Pro COBOL je platný rozsah omezen na 0 až +999 999 999 999 999 999. 64bitové celé číslo musí být zarovnáno na 8bajtovou hranici.
MQULONG (NEPOUŽITELNÝ)	32bitové celé číslo bez znaménka	Datový typ MQULONG je 32bitové binární celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu 0 až + 4 294 967 294, není-li kontext jinak omezen. Pro COBOL je platný rozsah omezen na 0 až +999 999 999 999. Hodnota MQULONG musí být zarovnána na 4bajtové hranici.
PMQACH pro všechny	Ukazatel	Ukazatel na datovou strukturu typu MQACH
PMQAIR	Ukazatel	Ukazatel na datovou strukturu typu MQAIR

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
PMQAXC	Ukazatel	Ukazatel na datovou strukturu typu MQAXC
PMQAXP	Ukazatel	Ukazatel na datovou strukturu typu MQAXP
PMQBMHO	Ukazatel	Ukazatel na datovou strukturu typu MQBMHO
PMQBO	Ukazatel	Ukazatel na datovou strukturu typu MQBO
PMQBOOL	Ukazatel	Ukazatel na data typu MQBOOL
PMQBYTE	Ukazatel	Ukazatel na data typu MQBYTE
PMQBYTEn	Ukazatel	Ukazatel na data typu MQBYTEn, kde n může být 8, 16, 24, 32, 40, 128
PMQCBC	Ukazatel	Ukazatel na datovou strukturu typu MQCBC
PMQCBD	Ukazatel	Ukazatel na datovou strukturu typu MQCBD
PMQCHAR	Ukazatel	Ukazatel na data typu MQCHAR
PMQCHARN	Ukazatel	Ukazatel na datový typ MQCHARN, kde n může být 4, 8, 12, 20, 28, 32, 48, 64, 128, 256, 264
PMQCHARV	Ukazatel	Ukazatel na datovou strukturu typu MQCHARV
PMQCIH	Ukazatel	Ukazatel na datovou strukturu typu MQCIH
PMQCMHO	Ukazatel	Ukazatel na datovou strukturu typu MQCMHO
PMQCNO	Ukazatel	Ukazatel na datovou strukturu typu MQCNO
PMQCSP.	Ukazatel	Ukazatel na datovou strukturu typu MQCSP.
PMQCTLO	Ukazatel	Ukazatel na datovou strukturu typu MQCTLO
PMQDH	Ukazatel	Ukazatel na datovou strukturu typu MQDH
PMQDHO	Ukazatel	Ukazatel na datovou strukturu typu MQDHO
PMQDLH	Ukazatel	Ukazatel na datovou strukturu typu MQDLH
PMQDMHO	Ukazatel	Ukazatel na datovou strukturu typu MQDMHO

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)		
Název elementárního datového typu	**Datový typ**	**Popis**
PMQDMPO	Ukazatel	Ukazatel na datovou strukturu typu MQDMPO
PMQEPH	Ukazatel	Ukazatel na datovou strukturu typu MQEPH
PMQFLOAT32	Ukazatel	Ukazatel na datovou strukturu typu MQFLOAT32
PMQFLOAT64	Ukazatel	Ukazatel na datovou strukturu typu MQFLOAT64
PMQFUNC-uživatelské rozhraní	Ukazatel	Ukazatel na funkci
PMQGMO	Ukazatel	Ukazatel na datovou strukturu typu MQGMO
PMQHCONFIG	Ukazatel	Ukazatel na data typu MQHCONFIG
PMQHCONN	Ukazatel	Ukazatel na data typu MQHCONN
PMQHMSG	Ukazatel	Ukazatel na data typu MQHMSG
PMQHOBJ	Ukazatel	Ukazatel na data typu MQHOBJ
PMQIIH	Ukazatel	Ukazatel na datovou strukturu typu MQIIH
PMQIMPO	Ukazatel	Ukazatel na datovou strukturu typu MQIMPO
PMQINT8	Ukazatel	Ukazatel na data typu MQINT8
PMQINT16	Ukazatel	Ukazatel na data typu MQINT16
PMQINT32	Ukazatel	Ukazatel na data typu MQINT32
PMQINT64	Ukazatel	Ukazatel na data typu MQINT64
PMQLONG	Ukazatel	Ukazatel na data typu MQLONG
PMQMD	Ukazatel	Ukazatel na strukturu typu MQMD
PMQMDE	Ukazatel	Ukazatel na datovou strukturu typu MQMDE
PMQMD1	Ukazatel	Ukazatel na datovou strukturu typu MQMD1
PMQMD2	Ukazatel	Ukazatel na datovou strukturu typu MQMD2
PMQMHBO	Ukazatel	Ukazatel na datovou strukturu typu MQMHBO
PMQOD	Ukazatel	Ukazatel na datovou strukturu typu MQOD
PMQOR	Ukazatel	Ukazatel na datovou strukturu typu MQOR
PMQPD	Ukazatel	Ukazatel na datovou strukturu typu MQPD

Tabulka 457. Názvy, typy a popisy elementárních datových typů (pokračování)

Název elementárního datového typu	Datový typ	Popis
PMQPID	Ukazatel	Ukazatel na identifikátor procesu
PMQMD	Ukazatel	Ukazatel na datovou strukturu typu MQMD
PMQPMO	Ukazatel	Ukazatel na datovou strukturu typu MQPMO
PMQPTR	Ukazatel	Ukazatel na data typu MQPTR
PMQRFH	Ukazatel	Ukazatel na datovou strukturu typu MQRFH
PMQRFH2	Ukazatel	Ukazatel na datovou strukturu typu MQRFH2
PMQRMH	Ukazatel	Ukazatel na datovou strukturu typu MQRMH
PMQRR	Ukazatel	Ukazatel na datovou strukturu typu MQRR
PMQSCO	Ukazatel	Ukazatel na datovou strukturu typu MQSCO
PMQSD	Ukazatel	Ukazatel na datovou strukturu typu MQSD
PMQSMPO	Ukazatel	Ukazatel na datovou strukturu typu MQSMPO
PMQSRO	Ukazatel	Ukazatel na datovou strukturu typu MQSRO
PMSSTS	Ukazatel	Ukazatel na datovou strukturu typu MQSTS
PMQTID	Ukazatel	Ukazatel na ID podprocesu
PMQTM	Ukazatel	Ukazatel na datovou strukturu typu MQTM
PMQTMC2	Ukazatel	Ukazatel na datovou strukturu typu MQTMC2
PMQUINT8	Ukazatel	Ukazatel na datový typ MQUINT8
PMQUINT16	Ukazatel	Ukazatel na datový typ MQUINT16
PMQUINT32	Ukazatel	Ukazatel na datový typ MQUINT32
PMQUINT64	Ukazatel	Ukazatel na datový typ MQUINT64
PMQULONG	Ukazatel	Ukazatel na datový typ MQULONG
PMQVOID	Ukazatel	
PMQWIH	Ukazatel	Ukazatel na datovou strukturu typu MQWIH
PMQXQH	Ukazatel	Ukazatel na datovou strukturu typu MQXQH

Deklarace datového typu C

Tabulka 458. Názvy a reprezentace datových typů C

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

Tabulka 458. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení
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	UNIX V 64bitovém systému UNIX:
	`typedef long;`
	AIX Na 32bitových systémech AIX:
	`typedef int64_t;`
	IBM i Linux z/OS V systémech Linux, IBM ia z/OS:
	`typedef long long;`
	Windows V systému Windows:
	`typedef _int64;`

Tabulka 458. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení

MQLONG	IBM i V systému IBM i:
	`typedef long MQLONG;`
	z/OS ALW Na jiných platformách:
	`if defined(MQ_64_BIT)`
	`  typedef int MQLONG;`
	`else`
	`  typedef long MQLONG;`

MQPID	`typedef MQLONG MQPID;`
MQPTR	`typedef void MQPOINTER MQPTR;`
MQTID	`typedef MQLONG MQTID;`
MQUINT8	`typedef unsigned char MQUINT8;`
MQUINT16	`typedef unsigned short MQUINT16;`

MQUINT64	UNIX V 64bitovém systému UNIX:
	`typedef unsigned long;`
	AIX Na 32bitových systémech AIX:
	`typedef uint64_t;`
	IBM i Linux z/OS V systémech Linux, IBM ia z/OS:
	`typedef unsigned long long;`
	Windows V systému Windows:
	`typedef unsigned _int64;`

Tabulka 458. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení

MQULONG (NEPOUŽITELNÝ)	IBM i V systému IBM i:
	`typedef unsigned long MQULONG;`
	z/OS ALW Na jiných platformách:
	`if defined(MQ_64_BIT)`
	`  typedef unsigned int MQULONG;`
	`else`
	`  typedef unsigned long MQULONG;`

PMQBO	`typedef MQBO MQPOINTER PMQBO;`
PMQBOOL	`typedef MQBOOL MQPOINTER PMQBOOL;`
PMQBYTE	`typedef MQBYTE MQPOINTER PMQBYTE;`
PMQBYTE8	`typedef MQBYTE8[8] MQPOINTER PMQBYTE8[8];`
PMQBYTE16	`typedef MQBYTE16[16] MQPOINTER PMQBYTE16[16];`
PMQBYTE24	`typedef MQBYTE24[24] MQPOINTER PMQBYTE24[24];`
PMQBYTE32	`typedef MQBYTE32[32] MQPOINTER PMQBYTE32[32];`
PMQBYTE40	`typedef MQBYTE40[40] MQPOINTER PMQBYTE40[40];`
PMQBYTE128	`typedef MQBYTE128[128] MQPOINTER PMQBYTE128[128];`
PMQCHAR	`typedef MQCHAR MQPOINTER PMQCHAR;`
PMQCHAR4	`typedef MQCHAR4[4] MQPOINTER PMQCHAR4[4];`
PMQCHAR8	`typedef MQCHAR8[8] MQPOINTER PMQCHAR8[8];`

Tabulka 458. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení
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-uživatelské rozhraní	`typedef void MQPOINTER PMQFUNC;`
PMQFLOAT32	`typedef MQFLOAT32 MQPOINTER PMQFLOAT32;`
PMQFLOAT64	`typedef MQFLOAT64 MQPOINTER PMQFLOAT64;`
PMQGMO	`typedef MQGMO MQPOINTER PMQGMO;`

Tabulka 458. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení
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;`

Tabulka 458. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení
PMQTM	`typedef MQTM MQPOINTER PMQTM;`
PMQTMC2	`typedef MQTMC2[2] MQPOINTER PMQTMC2[2];`
PMQUINT8	`typedef MQUINT8 MQPOINTER PMQUINT8;`
PMQUINT16	`typedef MQUINT16 MQPOINTER PMQUINT16;`
PMQULONG	`typedef MQULONG MQPOINTER PMQULONG;`
PMQVOID	`typedef void MQPOINTER PMQVOID;`
PMQWIH	`typedef MQWIH MQPOINTER PMQWIH;`
PMQXQH	`typedef MQXQH MQPOINTER PMQXQH;`
PPMQBO	`typedef PMQBO MQPOINTER PPMQBO;`
PPMQBYTE	`typedef PMQBYTE MQPOINTER PPMQBYTE;`
PPMQCHAR	`typedef PMQCHAR MQPOINTER PPMQCHAR;`
PPMQCNO	`typedef PMQCNO MQPOINTER PPMQCNO;`
PPMQGMO	`typedef PMQGMO MQPOINTER PPMQGMO;`
PPMQHCONN	`typedef PMQHCONN MQPOINTER PPMQHCONN;`
PPMQHOBJ	`typedef PMQHOBJ MQPOINTER PPMQHOBJ;`
PPMQLONG	`typedef PMQLONG MQPOINTER PPMQLONG;`

Tabulka 458. Názvy a reprezentace datových typů C (pokračování)	
Datový typ	**Zastoupení**
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;`
Kde `defined(MQ_64_BIT)` znamená 64bitovou platformu.	

Popis proměnné makra MQPOINTER naleznete v části “Datové typy” na stránce 263 .

Deklarace datového typu COBOL

Tabulka 459. Reprezentace a názvy datových typů v jazyce COBOL	
Datový typ	**Zastoupení**
MQBOOL	`PIC S9(9) BINARY`
MQBYTE	`PIC X`
MQBYTE8	`PIC X(8)`
MQBYTE16	`PIC X(16)`
MQBYTE24	`PIC X(24)`
MQBYTE32	`PIC X(32)`
MQBYTE40	`PIC X(40)`
MQCHAR	`PIC X`

Tabulka 459. Reprezentace a názvy datových typů v jazyce COBOL (pokračování)

Datový typ	**Zastoupení**
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	zapz/OS
	`PIC S9(9) COMP-5`
	Na jiných platformách
	`PIC S9(9) BINARY`

MQHOBJ	`PIC S9(9) BINARY`

Tabulka 459. Reprezentace a názvy datových typů v jazyce COBOL (pokračování)	
Datový typ	**Zastoupení**
MQINT8	`PIC S9(2) BINARY`
MQINT16	`PIC S9(4) BINARY`
MQINT64	`PIC S9(18) BINARY`
MQLONG	`PIC S9(9) BINARY`
MQPTR	`POINTER`
MQUINT8	`PIC 9(2) BINARY`
MQUINT16	`PIC 9(4) BINARY`
MQUINT64	`PIC 9(18) BINARY`
MQULONG (NEPOUŽITELNÝ)	`PIC 9(9) BINARY`

Deklarace datových typů PL/I

Tabulka 460. Názvy a reprezentace datových typů PL/I	
Datový typ	**Zastoupení**
MQBOOL	`fixed bin(31)`
MQBYTE	`char(1)`
MQBYTE8	`char(8)`
MQBYTE16	`char(16)`
MQBYTE24	`char(24)`

Tabulka 460. Názvy a reprezentace datových typů PL/I (pokračování)	
Datový typ	**Zastoupení**
MQBYTE32	`char(32)`
MQBYTE40	`char(40)`
MQCHAR	`char(1)`
MQCHAR4	`char(4)`
MQCHAR8	`char(8)`
MQCHAR12	`char(12)`
MQCHAR20	`char(20)`
MQCHAR28	`char(28)`
MQCHAR32	`char(32)`
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)`

Tabulka 460. Názvy a reprezentace datových typů PL/I (pokračování)	
Datový typ	**Zastoupení**
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 (NEPOUŽITELNÝ)	`fixed bin(32)`

Deklarace datového typu High Level Assembler

Tabulka 461. System/390 názvy a reprezentace datových typů asembleru	
Datový typ	**Zastoupení**
MQBOOL	`DS  F`
MQBYTE	`DS  XL1`
MQBYTE8	`DS  XL8`
MQBYTE16	`DS  XL16`

Tabulka 461. System/390 názvy a reprezentace datových typů asembleru (pokračování)	
Datový typ	**Zastoupení**
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`

Tabulka 461. System/390 názvy a reprezentace datových typů asembleru (pokračování)	
Datový typ	**Zastoupení**
MQHCONN	`DS F`
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 (NEPOUŽITELNÝ)	`DS F`

Datové typy struktury

Souhrn datových typů struktury, pravidla pro konzistentní mapování struktur MQI a konvence používané v jednotlivých popisech datových typů struktury.

- "Souhrn datových typů struktury použitých pro volání MQI nebo funkce ukončení" na stránce 259
- "Souhrn datových typů struktury použitých v datech zprávy" na stránce 261
- "Pravidla pro konzistentní mapování struktur MQI" na stránce 261
- "Konvence použité v každém popisu datového typu struktury" na stránce 261

Souhrn datových typů struktury použitých pro volání MQI nebo funkce ukončení

Tabulka 462. Datové typy struktury používané pro volání MQI nebo funkce ukončení		
Struktura	**Popis**	**Volání, kde se používá**
MQACH	Záhlaví řetězce uživatelské procedury rozhraní API	

Tabulka 462. Datové typy struktury používané pro volání MQI nebo funkce ukončení (pokračování)		
Struktura	**Popis**	**Volání, kde se používá**
MQAIR	Záznam ověřovacích informací	MQCONNX
MQAXC	Kontext uživatelské procedury rozhraní API	
MQAXP	Parametr uživatelské procedury rozhraní API	
MQBMHO	Volby zpracování vyrovnávací paměti pro zprávu	MQBUFMH
MQBO	Volby začátku	MQBEGIN
MQCBD	Deskriptor zpětného volání	MQCB
MQCBO	Volby vytvoření balíku	mqCreateBag
MQCHARV	Řetězec proměnné délky	MQINQMP
MQCNO	Volby připojení	MQCONNX
MQCSP	Parametry zabezpečení	MQCONNX
MQCTLO	Volby zpětného dotazu	MQCTL
MQDMPO	Volby vlastnosti odstranění zprávy	MQDLTMP
MQGMO	Volby získání zprávy	MQGET
MQIMPO	Volby vlastnosti dotazování zprávy	MQINQMP
MQMD	deskriptor zprávy	MQBUFMH, MQMHBUF, MQCB, MQGET, MQPUT, MQPUT1
MQMHBO	Popisovač zprávy pro volby vyrovnávací paměti	MQMHBUF
MQOD	deskriptor objektu	MQOPEN, MQPUT1
MQOR	Záznam objektu	MQOPEN, MQPUT1
MQPD	Deskriptor vlastnosti	MQSETMP
MQPMO	Volby vložení zprávy	MQPUT, MQPUT1
MQPMR	Záznam zprávy vložení	MQPUT, MQPUT1
MQRR	Záznam odpovědi	MQOPEN, MQPUT, MQPUT1
MQSCO	Volby konfigurace TLS	MQCONNX
MQSD	Deskriptor odběru	MQSUB
MQSMPO	Volba nastavení vlastnosti zprávy	MQSETMP
MQSRO	Volby požadavku na odběr	MQSUBRQ
MQSTS	Struktura vykazování stavu	MQSTAT

Souhrn datových typů struktury použitých v datech zprávy

Tabulka 463. Datové typy struktury použité v datech zprávy	
Struktura	**Popis**
MQCIH.	Záhlaví informací CICS
MQCFH	Záhlaví PCF
MQEPH	Vložené záhlaví PCF
MQDH	Záhlaví distribuce
MQDLH	Záhlaví nedoručených zpráv (nedoručená zpráva)
MQIIH.	Záhlaví informací IMS
MQMDE	Rozšíření deskriptoru zpráv
MQRFH.	Pravidla a formátování záhlaví
MQRFH2	Pravidla a formátování záhlaví 2
MQRMH	Záhlaví referenční zprávy
MQTM	zpráva spouštěče
MQTMC2	Zpráva spouštěče (znakový formát 2)
MQWIH	Záhlaví informací o práci
MQXQH	Záhlaví přenosové fronty

Poznámka: Struktura MQDXP (parametr uživatelské procedury převodu dat) je popsána v části "Uživatelská procedura převodu dat" na stránce 905spolu s přidruženými voláními převodu dat.

Pravidla pro konzistentní mapování struktur MQI

Programovací jazyky se liší svou úrovní podpory struktur a jsou přijata určitá pravidla a konvence pro konzistentní mapování struktur MQI v jednotlivých programovacích jazycích:

1. Struktury musí být sladěny na svých přirozených hranicích.
   - Většina struktur MQI vyžaduje 4bajtové zarovnání.
   - V systému IBM istruktury obsahující ukazatele vyžadují 16bajtové zarovnání: MQCNO, MQOD, MQPMO.
2. Každé pole ve struktuře musí být zarovnáno na své přirozené hranici.
   - Pole s datovými typy, které se rovnají MQLONG, musí být zarovnána na 4bajtové hranice.
   - Pole s datovými typy, které se rovnají MQPTR, musí být zarovnána s 16bajtovými hranicemi v systému IBM ia 4bajtovými hranicemi v jiných prostředích.
   - Ostatní pole jsou zarovnána na 1bajtové hranice.
3. Délka struktury musí být násobkem jejího zarovnání hranice.
   - Většina struktur MQI má délku, která je násobek 4 bajtů.
   - V systému IBM imají struktury obsahující ukazatele délku, která je násobkem 16 bajtů.
4. V případě potřeby je nutné přidat vyplňující bajty nebo pole, aby byla zajištěna shoda s předchozími pravidly.

Konvence použité v každém popisu datového typu struktury

Popis každého datového typu struktury zahrnuje:

- Přehled o účelu a využití struktury
- Popisy polí ve struktuře, ve formě, která je nezávislá na programovacím jazyku
- Příklady, jak je struktura deklarována v jednotlivých podporovaných programovacích jazycích

Popis každého datového typu struktury obsahuje následující sekce:

Název struktury
Název struktury následovaný souhrnem polí ve struktuře.

Přehled
Stručný popis účelu a použití struktury.

Pole
Popis polí. Pro každé pole je název pole následován jeho základním datovým typem v závorkách (). V textu jsou názvy polí zobrazeny kurzívou; například *Version*.

K dispozici je také popis účelu pole spolu se seznamem všech hodnot, které může pole přijmout. Názvy konstant jsou uvedeny velkými písmeny, například MQGMO_STRUC_ID. Sada konstant se stejnou předponou se zobrazí pomocí znaku *, například: MQIA_ *.

V popisech polí se používají následující výrazy:

Vstup
Informace zadáváte do pole, když voláte.

výstup
Správce front vrátí informace do pole po dokončení nebo selhání volání.

Vstup a výstup
Informace se zadají do pole při volání a správce front změní informace při dokončení nebo selhání volání.

Počáteční hodnoty
Tabulka zobrazující počáteční hodnoty pro každé pole v souborech definice dat dodaných s rozhraním MQI.

C prohlášení
Typická deklarace struktury v C.

Deklarace jazyka COBOL
Typická deklarace struktury v COBOLu.

Prohlášení PL/I
Typické prohlášení o struktuře v PL/I.

Deklarace High Level Assembler
Typická deklarace struktury v jazyku asembleru System/390 .

Vizuální základní deklarace
Typická deklarace struktury v jazyku Visual Basic.

Programování v C

Informace, které vám pomohou používat rozhraní MQI z programovacího jazyka C.

- “Soubory záhlaví” na stránce 263
- “Funkce” na stránce 263
- “Parametry s nedefinovaným datovým typem” na stránce 263
- “Datové typy” na stránce 263
- “Manipulace s binárními řetězci” na stránce 264
- “Manipulace se znakovými řetězci” na stránce 264
- “Počáteční hodnoty pro struktury” na stránce 264
- “Počáteční hodnoty pro dynamické struktury” na stránce 265
- “Použít z C++” na stránce 265

- “Konvence notace” na stránce 265

Soubory záhlaví

Tabulka 464. Soubory záhlaví C	
Soubor	**Obsah**
CMQC	Prototypy funkcí, datové typy a pojmenované konstanty pro hlavní rozhraní MQI
CMQXC	Prototypy funkcí, datové typy a pojmenované konstanty pro uživatelskou proceduru převodu dat
CMQEC	Funkční prototypy, datové typy a pojmenované konstanty pro hlavní rozhraní MQI, uživatelskou proceduru převodu dat a strukturu vstupních bodů rozhraní (CMQEC zahrnuje CMQXC a CMQC).
CMQSTRC	Funkce, které převádějí definice konstant MQI na textový ekvivalent. **Upozornění:** z/OS Použitelné pro z/OS z IBM MQ 9.1. Programy používající tento hlavičkový soubor musí být kompilovány s volbou kompilátoru LONGNAME.

Chcete-li zlepšit přenositelnost aplikací, kódujte název hlavičkového souboru malými písmeny v direktivě preprocesoru `#include` :

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

Funkce

Při každém vyvolání funkce není nutné zadávat všechny parametry předávané adresou.

- Předejte parametry, které jsou *pouze vstupní* , a typu MQHCONN, MQHOBJ nebo MQLONG podle hodnoty.
- Předejte všechny ostatní parametry podle adresy.

Pokud není konkrétní parametr povinný, použijte jako parametr při vyvolání funkce ukazatel null místo adresy dat parametru. Parametry, pro které je to možné, jsou identifikovány v popisech volání.

Jako hodnota funkce není vrácen žádný parametr; v terminologii C to znamená, že všechny funkce vrátí hodnotu `void`.

Atributy funkce jsou definovány pomocí proměnné makra MQENTRY; hodnota této proměnné makra závisí na prostředí.

Parametry s nedefinovaným datovým typem

Parametr **`Buffer`** ve funkcích MQGET, MQPUT a MQPUT1 má nedefinovaný datový typ. Tento parametr se používá k odeslání a přijetí dat zprávy aplikace.

Parametry tohoto řazení jsou zobrazeny v příkladech jazyka C jako pole MQBYTE. Tímto způsobem můžete deklarovat parametry, ale obvykle je vhodnější je deklarovat jako konkrétní strukturu, která popisuje rozvržení dat ve zprávě. Deklarujte skutečný parametr funkce jako ukazatel na neobsazený a uveďte adresu jakéhokoli druhu dat jako parametr při vyvolání funkce.

Datové typy

Definujte všechny datové typy pomocí příkazu C `typedef` . Pro každý datový typ definujte také odpovídající datový typ ukazatele. Název datového typu ukazatele je název elementárního datového typu nebo datového typu struktury s předponou s písmenem P pro označení ukazatele. Definujte atributy

ukazatele pomocí proměnné makra MQPOINTER; hodnota této proměnné makra závisí na prostředí. Následující ilustruje, jak deklarovat datové typy ukazatele:

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

Manipulace s binárními řetězci

Deklarujte řetězce binárních dat jako jeden z datových typů MQBYTEn.

Kdykoli kopírujete, porovnáváte nebo nastavujete pole tohoto typu, použijte funkce jazyka C **memcpy**, **memcmp**nebo **memset** ; například:

```
#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));
```

Nepoužívejte řetězcové funkce **strcpy**, **strcmp**, **strncpy**nebo **strncmp**, protože nefungují správně pro data deklarovaná s datovými typy MQBYTEn.

Manipulace se znakovými řetězci

Když správce front vrátí znaková data aplikaci, správce front vždy vyplní znaková data mezerami do definované délky pole. Správce front *nevrací* řetězce ukončené hodnotou null.

Proto při kopírování, porovnávání nebo zřetězení takových řetězců použijte řetězcové funkce **strncpy**, **strncmp**nebo **strncat**.

Nepoužívejte řetězcové funkce, které vyžadují ukončení řetězce hodnotou null (**strcpy**, **strcmp**, **strcat**). Také nepoužívejte funkci **strlen** k určení délky řetězce; použijte místo toho funkci **sizeof** k určení délky pole.

Počáteční hodnoty pro struktury

Soubory záhlaví definují různé proměnné maker, které lze použít k zadání počátečních hodnot pro struktury produktu MQ při deklaraci instancí těchto struktur.

Tyto proměnné maker mají názvy ve tvaru MQxxx_DEFAULT, kde MQxxx představuje název struktury. Používají se následujícím způsobem:

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

Pro některá znaková pole (například pole *StrucId* , která se vyskytují ve většině struktur, nebo pole *Format* , které se vyskytuje v deskriptoru MQMD) definuje rozhraní MQI konkrétní platné hodnoty. Pro každou z platných hodnot jsou poskytnuty *dvě* proměnné makra:

- Jedna proměnná makra definuje hodnotu jako řetězec s délkou, bez odvozených shod s hodnotou null, přesně definovanou délku pole. Například pro pole *Format* v MQMD je poskytnuta následující proměnná makra (¬ představuje jeden prázdný znak):

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

  Použijte tento formulář s funkcemi `memcpy` a `memcmp` .
- Druhá proměnná makra definuje hodnotu jako pole znaků; název této proměnné makra je název řetězce s příponou _ARRAY. Příklad:

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

  Pomocí tohoto formuláře inicializujte pole, když deklarujete instanci struktury s hodnotami odlišnými od hodnot poskytnutých proměnnou makra MQMD_DEFAULT. (To není vždy nutné; v některých prostředích můžete použít řetězcový tvar hodnoty v obou situacích. Pro deklarace však můžete použít formulář pole, protože je to nezbytné pro kompatibilitu s programovacím jazykem C + +.)

Počáteční hodnoty pro dynamické struktury

Když je požadován proměnný počet instancí struktury, instance jsou obvykle vytvořeny v hlavní paměti získané dynamicky pomocí funkcí `calloc` nebo `malloc` . Chcete-li inicializovat pole v těchto strukturách, zvažte následující techniku:

1. Deklarujte instanci struktury pomocí příslušné proměnné makra MQxxx_DEFAULT pro inicializaci struktury. Tato instance se stane modelem pro další instance:

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

   Klíčová slova `static` nebo `auto` lze kódovat v deklaraci, aby se podle potřeby poskytla statická nebo dynamická životnost instance modelu.
2. Pomocí funkcí `calloc` nebo `malloc` získáte úložiště pro dynamickou instanci struktury:

   ```
   PMQMD Instance;
   Instance = malloc(sizeof(MQMD)); /* get storage for dynamic instance */
   ```
3. Pomocí funkce `memcpy` zkopírujte instanci modelu do dynamické instance:

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

Použít z C++

Pro programovací jazyk C++ obsahují hlavičkové soubory následující další příkazy, které jsou zahrnuty pouze při použití kompilátoru C + +:

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

/* rest of header file */

#ifdef __cplusplus
  }
#endif
```

Konvence notace

Tyto informace ukazují, jak vyvolat funkce a deklarovat parametry.

V některých případech jsou parametry pole o velikosti, která není pevná. Pro tyto účely se k reprezentaci číselné konstanty používá malé písmeno n. Při kódování deklarace pro tento parametr nahraďte hodnotu n požadovanou číselnou hodnotou.

Programování v jazyce COBOL

Informace, které vám pomohou používat rozhraní MQI z programovacího jazyka COBOL.

- "Soubory COPY" na stránce 266
- "Struktury" na stránce 267
- "Ukazatele" na stránce 268
- "Pojmenované konstanty" na stránce 268
- "Konvence notace" na stránce 268

Soubory COPY

K dispozici jsou různé soubory COPY, které vám pomohou při psaní aplikačních programů v jazyce COBOL, které používají rozhraní MQI. Existují dva soubory obsahující pojmenované konstanty a dva soubory pro každou ze struktur.

Každá struktura je poskytována ve dvou formách: formulář s počátečními hodnotami a formulář bez:

- Použijte struktury s počátečními hodnotami v WORKING-STORAGE SECTION programu COBOL; jsou obsaženy v souborech COPY s názvy s příponou V (pro hodnoty).
- Použijte struktury bez počátečních hodnot v `LINKAGE SECTION` programu v jazyce COBOL; jsou obsaženy v souborech COPY s názvy s příponou L (pro propojení).

Soubory COPY jsou shrnuty v následující tabulce. Ne všechny uvedené soubory jsou k dispozici ve všech prostředích.

Tabulka 465. Soubory COPY v jazyce COBOL

Soubor (s počátečními hodnotami)	**Soubor (bez počátečních hodnot)**	**Obsah**
CMQAIRV	CMQAIRL	Záznam ověřovacích informací
CMQBOV	CMQBOL	Struktura voleb začátku
CMQCIHV	CMQCIHL	Struktura záhlaví informací CICS
CMQCNOV	CMQCNOL	Struktura voleb připojení
CMQDHV	CMQDHL	Struktura záhlaví distribuce
CMQDLHV	CMQDLHL	Struktura záhlaví nedoručovacího dopisu
CMQDXPV	CMQDXPL	Struktura parametrů uživatelské procedury pro převod dat
CMQGMOV	CMQGMOL	Získat strukturu voleb zprávy
CMQIIHV	CMQIIHL	Struktura záhlaví informací IMS
CMQMDV	CMQMDL	Struktura deskriptoru zpráv
CMQMDEV	CMQMDEL	Struktura rozšíření deskriptoru zpráv
CMQMD1V	CMQMD1L	Struktura deskriptoru zpráv verze 1
CMQODV	CMQODL	Struktura deskriptoru objektu
CMQORV	CMQORL	Struktura záznamu objektu
CMQPMOV	CMQPMOL	Struktura voleb vložení zprávy

Tabulka 465. Soubory COPY v jazyce COBOL (pokračování)		
Soubor (s počátečními hodnotami)	**Soubor (bez počátečních hodnot)**	**Obsah**
CMQRFHV	CMQRFHL	Pravidla a struktura záhlaví formátování
CMQRFH2V	CMQRFH2L	Pravidla a formátování struktury záhlaví verze 2
CMQRMHV	CMQRMHL	Struktura záhlaví referenční zprávy
CMQRRV	CMQRRL	Struktura záznamu odezvy
CMQSCOV	CMQSCOL	Volby konfigurace TLS
CMQTMV	CMQTML	Struktura zprávy spouštěče
CMQTMCV	CMQTMCL	Struktura zprávy spouštěče (znakový formát)
CMQTMC2V	CMQTMC2L	Struktura zprávy spouštěče (znakový formát) verze 2
CMQWIHV	CMQWIHL	Struktura záhlaví informací o práci
CMQXQHV	CMQXQHL	Struktura záhlaví přenosové fronty
CMQV	-	Pojmenované konstanty pro hlavní rozhraní MQI
CMQXV	-	Pojmenované konstanty pro uživatelskou proceduru převodu dat
CMQMD2V	CMQMD2L	Struktura deskriptoru zpráv verze 2

Struktury

V souboru COPY začíná každá deklarace struktury položkou level-10 . To vám umožňuje deklarovat několik instancí struktury kódováním deklarace level-01 a následným použitím příkazu COPY ke zkopírování ve zbytku deklarace struktury. Chcete-li se odkázat na příslušnou instanci, použijte klíčové slovo 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.
```

Zarovnejte struktury na příslušných hranicích. Pokud použijete příkaz COPY k zahrnutí struktury za položku, která není položkou level-01 , ujistěte se, že struktura začíná na příslušném posunutí od začátku položky level-01 . Většina struktur MQI vyžaduje 4bajtové zarovnání. Výjimkami jsou MQCNO, MQOD a MQPMO, které vyžadují 16bajtové zarovnání v systému IBM i.

V této sekci jsou názvy polí ve strukturách zobrazeny bez předpony. V COBOLu mají názvy polí předponu s názvem struktury následovaným pomlčkou. Pokud však název struktury končí číslicí, což znamená, že struktura je druhou nebo novější verzí původní struktury, číselná číslice se vynechá z předpony. Názvy polí v COBOLu jsou uvedeny velkými písmeny (i když malá nebo smíšená písmena mohou být použita v případě potřeby). Například pole *MsgType* popsané v “MQMD-Deskriptor zpráv” na stránce 424 se stane MQMD-MSGTYPE v COBOLu.

Struktury přípon V jsou deklarovány s počátečními hodnotami pro všechna pole; musíte nastavit pouze ta pole, kde chcete hodnotu, která se liší od zadané počáteční hodnoty.

Ukazatele

Některé struktury musí adresovat volitelná data, která mohou být nesouvislá se strukturou, například záznamy MQOR a MQRR adresované strukturou MQOD.

Pro adresování těchto volitelných dat obsahují struktury pole, která jsou deklarována s datovým typem ukazatele. Avšak COBOL nepodporuje datový typ ukazatele ve všech prostředích. Z tohoto důvodu lze volitelná data adresovat také pomocí polí, která obsahují posunutí dat od začátku struktury.

Chcete-li portovat aplikaci mezi prostředími, ujistěte se, zda je datový typ ukazatele dostupný ve všech zamýšlených prostředích. Pokud tomu tak není, musí aplikace adresovat volitelná data pomocí polí offsetu místo polí ukazatele.

V těch prostředích, kde nejsou podporovány ukazatele, deklarujte pole ukazatele jako bajtové řetězce odpovídající délky, přičemž počáteční hodnota je bajtový řetězec s hodnotou typu all-null. Tuto počáteční hodnotu neměňte, pokud používáte pole offsetu.

Pojmenované konstanty

V této informaci jsou zobrazeny názvy konstant, které obsahují znak podtržítka (_) jako součást názvu. V COBOLu použijte znak pomlčky (-) místo podtržítka.

Konstanty, které mají hodnoty znakových řetězců, používají znak apostrofu jako oddělovač řetězců ('). V některých prostředích může být nutné zadat odpovídající volbu kompilátoru, aby kompilátor přijal znak apostrofu jako oddělovač řetězců namísto znaku dvojitých uvozovek.

Pojmenované konstanty jsou deklarovány v souborech COPY jako položky level-10 . Chcete-li použít konstanty, deklarujte položku level-01 explicitně a pak použijte příkaz COPY ke kopírování v deklaracích konstant:

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

Předchozí metoda způsobí, že konstanty zabírají paměť v programu, i když nejsou odkazovány. Pokud zahrnete konstanty do mnoha samostatných programů v rámci stejné jednotky spuštění, existuje více kopií konstant, které zbytečně spotřebovávají hlavní paměť. Vyhněte se tomuto efektu pomocí jedné z následujících technik:

- Přidejte klauzuli GLOBAL do deklarace level-01 :

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

  Tím přidělí paměť pouze pro jednu sadu konstant v rámci jednotky spuštění. Na konstanty však může odkazovat libovolný program v rámci jednotky spuštění, nikoli pouze program, který obsahuje deklaraci level-01 .

  Poznámka: Klauzule GLOBAL není podporována ve všech prostředích.
- Ručně zkopírujte do každého programu pouze ty konstanty, na které tento program odkazuje. Nepoužívejte příkaz COPY ke kopírování všech konstant do programu.

Konvence notace

Předchozí sekce v tomto tématu ukazují, jak vyvolat volání a deklarovat parametry. V některých případech jsou parametry tabulky nebo znakové řetězce, jejichž velikost není pevná. Pro tyto účely se k reprezentaci číselné konstanty používá malé písmeno n. Při kódování deklarace pro tento parametr nahraďte hodnotu n požadovanou číselnou hodnotou.

Programování High Level Assembler

Informace, které vám pomohou s použitím rozhraní MQI z programovacího jazyka System/390 Assembler.

Makra

Existují dvě makra pro pojmenované konstanty a jedno makro pro každou ze struktur. Tyto soubory jsou shrnuty v následující tabulce.

Tabulka 466. Makra modulu sestavení

Soubor	Obsah
CMQA	Pojmenované konstanty (rovnítko) pro hlavní MQI
CMQCIHA	Struktura záhlaví informací CICS
CMQCNOA	Struktura voleb připojení
CMQDLHA	Struktura záhlaví nedoručovacího dopisu
CMQDXPA	Struktura parametrů uživatelské procedury pro převod dat
CMQGMOA	Získat strukturu voleb zprávy
CMQIIHA	Struktura záhlaví informací IMS
CMQMDA	Struktura deskriptoru zpráv
CMQMDEA	Struktura rozšíření deskriptoru zpráv
CMQODA	Struktura deskriptoru objektu
CMQPMOA	Struktura voleb vložení zprávy
CMQRFHA	Pravidla a struktura záhlaví formátování
CMQRFH2A	Pravidla a formátování struktury záhlaví verze 2
CMQRMHA	Struktura záhlaví referenční zprávy
CMQTMA	Struktura zprávy spouštěče
CMQTMC2A	Struktura zprávy spouštěče (znakový formát) verze 2
CMQVERA	Řízení verze struktury
CMQWIHA	Struktura záhlaví informací o práci
CMQXA	Pojmenované konstanty pro uživatelskou proceduru převodu dat
CMQXPA	Struktura parametru křížové uživatelské procedury rozhraní API
CMQXQHA	Struktura záhlaví přenosové fronty

Struktury

Struktury jsou generovány makry, které mají různé parametry pro řízení akce makra. Viz téma “Struktury” na stránce 270

Makro produktu CMQVERA

Toto makro umožňuje nastavit výchozí hodnotu, která má být použita pro parametr DCLVER v makrech struktury.

Hodnotu určenou funkcí CMQVERA použije makro struktury pouze v případě, že ve vyvolání makra struktury vynecháte parametr DCLVER . Výchozí hodnota je nastavena kódováním makra CMQVERA s parametrem DCLVER :

DCLVER=CURRENT
Výchozí verze je nastavena na aktuální (nejnovější) verzi.

DCLVER=ZVLÁŠTNÍ
Výchozí verze je nastavena na verzi určenou parametrem VERSION .

Musíte zadat parametr **DCLVER** a hodnota musí být velká písmena. Hodnota nastavená CMQVERA zůstává výchozí hodnotou až do dalšího vyvolání CMQVERA nebo do konce sestavení. Pokud vynecháte CMQVERA, předvolba je DCLVER=CURRENT.

Konvence notace

Další témata ukazují, jak vyvolat volání a deklarovat parametry. V některých případech jsou parametry pole nebo znakové řetězce s velikostí, která není pevná, a malá písmena n se používají k reprezentaci číselné konstanty. Při kódování deklarace pro tento parametr nahraďte hodnotu n požadovanou číselnou hodnotou.

Struktury
Struktury jsou generovány makry, které mají různé parametry pro řízení akce makra.

Poznámka: Čas od času jsou zavedeny nové verze struktur IBM MQ . Další pole v nové verzi mohou způsobit, že struktura, která byla dříve menší než 256 bajtů, bude větší než 256 bajtů. Z tohoto důvodu zapište instrukce assembleru, které jsou určeny ke kopírování struktury IBM MQ nebo k nastavení struktury IBM MQ na hodnoty null, aby správně fungovaly se strukturami, které mohou být větší než 256 bajtů. Případně můžete pomocí parametru makra DCLVER nebo makra CMQVERA s parametrem VERSION deklarovat specifickou verzi struktury.

- "Určení názvu struktury" na stránce 270
- "Určení tvaru struktury" na stránce 270
- "Řízení verze struktury" na stránce 271
- "Deklarace jedné struktury vložené do jiné struktury" na stránce 271
- "Určení počátečních hodnot pro pole" na stránce 271
- "Řízení výpisu" na stránce 272

Určení názvu struktury

Chcete-li deklarovat více než jednu instanci struktury, makro před název každého pole ve struktuře uvede řetězec určený uživatelem a podtržítko.

Použitý řetězec je popisek určený při vyvolání makra. Není-li uveden žádný popisek, použije se k vytvoření předpony název struktury:

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

Deklarace struktury zobrazené v této sekci používají výchozí předponu.

Určení tvaru struktury

Deklarace struktury mohou být generovány makrem v jednom ze dvou formulářů řízených parametrem DSECT :

DSECT = YES
Instrukce asembleru DSECT se používá ke spuštění nové datové sekce; definice struktury bezprostředně následuje za příkazem DSECT . Popisek pro vyvolání makra se používá jako název datové sekce; není-li zadán žádný popisek, použije se název struktury.

DSECT = NO
Pokyny sestavovače DC se používají k definování struktury na aktuální pozici v rutině. Pole jsou inicializována s hodnotami, které lze určit kódováním příslušných parametrů při vyvolání makra. Pole, pro která nejsou při vyvolání makra zadány žádné hodnoty, jsou inicializována s výchozími hodnotami.

Zadaná hodnota musí být velká písmena. Není-li parametr DSECT uveden, předpokládá se DSECT = NO .

Řízení verze struktury

Standardně makra vždy deklarují nejnovější verzi každé struktury.

Ačkoli můžete použít parametr makra VERSION k určení hodnoty pro pole *Version* ve struktuře, tento parametr definuje počáteční hodnotu pro pole *Version* a neřídí verzi skutečně deklarované struktury. Chcete-li řídit verzi deklarované struktury, použijte parametr DCLVER :

DCLVER=CURRENT
Deklarovaná verze je aktuální (nejnovější) verze.

DCLVER=ZVLÁŠTNÍ
Deklarovaná verze je verze určená parametrem VERSION . Pokud vynecháte parametr VERSION , předvolba je verze 1.

Zadáte-li parametr VERSION , hodnota musí být číselná konstanta, která se sama definuje, nebo uvedená konstanta pro požadovanou verzi (například MQCNO_VERSION_3). Pokud uvedete jinou hodnotu, struktura se deklaruje, jako by byla uvedena DCLVER=CURRENT , i když se hodnota VERSION interpretuje jako platná hodnota.

Zadaná hodnota musí být velká písmena. Vynecháte-li parametr DCLVER , bude použitá hodnota převzata z globální proměnné makra MQDCLVER . Tuto proměnnou můžete nastavit pomocí makra CMQVERA.

Deklarace jedné struktury vložené do jiné struktury

Chcete-li deklarovat jednu strukturu jako komponentu jiné struktury, použijte parametr VNOŘENÝ :

NESTED=ANO
Deklarace struktury je vnořena do jiného objektu.

NESTED=NO
Deklarace struktury není vnořená v jiné.

Zadaná hodnota musí být velká písmena. Pokud vynecháte parametr VNOŘENÝ , předpokládá se NESTED=NO .

Určení počátečních hodnot pro pole

Zadejte hodnotu, která má být použita k inicializaci pole ve struktuře, pomocí kódování názvu tohoto pole (bez předpony) jako parametru při vyvolání makra, spolu s požadovanou hodnotou.

Chcete-li například deklarovat strukturu deskriptoru zpráv s polem *MsgType* inicializovaným pomocí MQMT_REQUEST a s polem *ReplyToQ* inicializovaným pomocí řetězce "MY_REPLY_TO_QUEUE", použijte následující:

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

Zadáte-li pojmenovanou konstantu (rovnítko) jako hodnotu při vyvolání makra, definujte pojmenovanou konstantu pomocí makra CMQA. Neuzavírejte hodnoty znakových řetězců do jednoduchých uvozovek.

Řízení výpisu

Pomocí parametru LIST můžete řídit vzhled deklarace struktury ve výpisu modulu sestavení:

LIST = YES
Deklarace struktury se zobrazí ve výpisu modulu sestavení.

LIST = NO
Deklarace struktury se neobjevuje ve výpisu modulu sestavení.

Zadaná hodnota musí být velká písmena. Pokud vynecháte parametr LIST , předpokládá se `LIST = NO` .

MQAIR-záznam ověřovacích informací

Struktura MQAIR umožňuje aplikaci, která je spuštěna jako IBM MQ MQI client , uvést informace o ověřovateli, který se má použít pro připojení klienta. Struktura je vstupní parametr volání MQCONNX.

Dostupnost

Struktura MQAIR je k dispozici pro následující klienty:

- AIX
- Linux
- Windows

Znaková sada a kódování

Data v MQAIR musí být ve znakové sadě a kódování lokálního správce front; tato data jsou dána atributem správce front **CodedCharSetId** a MQENC_NATIVE.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 467. Pole v MQAIR

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQAIR_STRUC_ID	'AIR¬'
Verze (číslo verze struktury)	MQAIR_VERSION_1	1
AuthInfoTyp (typ ověřovacích informací)	MQAIT_CRL_LDAP	1
AuthInfoConnName (název připojení serveru LDAP CRL)	Není	Prázdný řetězec nebo mezery
LDAPUserNamePtr (adresa jména uživatele LDAP)	Není	Ukazatel Null nebo bajty s hodnotou Null
LDAPUserNameOffset (posunutí jména uživatele LDAP od začátku MQSCO)	Není	0
LDAPUserNameDélka (délka jména uživatele LDAP)	Není	0
LDAPPassword (heslo pro přístup k serveru LDAP)	Není	Prázdný řetězec nebo mezery

Tabulka 467. Pole v MQAIR (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
Poznámka: Zbývající pole jsou ignorována, pokud je hodnota *Verze* menší než hodnota MQAIR_VERSION_2.		
OCSPResponderURL (URL , na které lze kontaktovat odpovídací modul OCSP)	Není	Prázdný řetězec nebo mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQAIR_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

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

Deklarace jazyka

C prohlášení pro MQAIR

```
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 */

};
```

Deklarace jazyka COBOL pro MQAIR

```
**   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).
```

Vizuální základní deklarace pro MQAIR

```
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
```

StrucId (MQCHAR4) pro MQAIR

Jedná se o identifikátor struktury struktury záznamu ověřovacích informací. Vždy se jedná o vstupní pole. Jeho hodnota je MQAIR_STRUC_ID.

Hodnota musí být:

MQAIR_STRUC_ID
Identifikátor záznamu ověřovacích informací.

Pro programovací jazyk C je definována také konstanta MQAIR_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQAIR_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQAIR

Toto je číslo verze struktury záznamu ověřovacích informací. Vždy se jedná o vstupní pole.

Hodnota musí být jedna z následujících:

MQAIR_VERSION_1
Záznam ověřovacích informací Version-1 .

Toto je počáteční hodnota tohoto pole.

MQAIR_VERSION_2
Záznam ověřovacích informací Version-2 .

Následující konstanta určuje číslo verze aktuální verze:

MQAIR_CURRENT_VERSION
Aktuální verze záznamu ověřovacích informací.

Typ AuthInfo(MQLONG) pro MQAIR

Jedná se o typ ověřovacích informací obsažených v záznamu.

Hodnota může být jeden ze dvou následujících parametrů:

MQAIT_CRL_LDAP
Kontrola odvolání certifikátů pomocí serveru LDAP.

MQAIT_OCSP
Kontrola odvolání certifikátů pomocí protokolu OCSP.

Není-li hodnota platná, volání selže s kódem příčiny MQRC_AUTH_INFO_TYPE_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQAIT_CRL_LDAP.

AuthInfoConnName (MQCHAR264) pro MQAIR

Jedná se buď o název hostitele, nebo síťovou adresu hostitele, na kterém je spuštěn server LDAP. Za ním může následovat volitelné číslo portu uzavřené v závorkách. Výchozí číslo portu je 389.

Pokud je hodnota kratší než délka pole, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku pole. Pokud hodnota není platná, volání selže s kódem příčiny MQRC_AUTH_INFO_CONN_NAME_ERROR.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_AUTH_INFO_CONN_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.

LDAPUserNamePtr (PMQCHAR) pro MQAIR

Jedná se o jméno uživatele LDAP.

Skládá se z rozlišujícího názvu uživatele, který se pokouší o přístup k serveru CRL LDAP. Pokud je hodnota kratší než délka uvedená parametrem *LDAPUserNameLength*, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku *LDAPUserNameLength*. Je-li hodnota *LDAPUserNameLength* nula, pole se ignoruje.

Jméno uživatele LDAP můžete zadat jedním ze dvou způsobů:

- Pomocí pole ukazatele *LDAPUserNamePtr*

  V tomto případě může aplikace deklarovat řetězec, který je oddělený od struktury MQAIR, a nastavit *LDAPUserNamePtr* na adresu řetězce.

  Zvažte použití produktu *LDAPUserNamePtr* pro programovací jazyky, které podporují datový typ ukazatele způsobem, který je přenositelný do různých prostředí (například programovací jazyk C).
- Pomocí pole offsetu *LDAPUserNameOffset*

  V tomto případě musí aplikace deklarovat složenou strukturu obsahující strukturu MQSCO následovanou polem záznamů MQAIR následovaným řetězci jména uživatele LDAP a nastavit hodnotu *LDAPUserNameOffset* na posun příslušného řetězce názvu od začátku struktury MQAIR. Ujistěte se, že je tato hodnota správná a má hodnotu, kterou lze umístit v rámci MQLONG (nejrestriktivnějším programovacím jazykem je COBOL, pro který je platný rozsah -999 999 999 999 až +999 999 999 999).

  Zvažte použití *LDAPUserNameOffset* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele způsobem, který nemusí být přenositelný do různých prostředí (například programovací jazyk COBOL).

Bez ohledu na zvolenou techniku použijte pouze jednu z možností *LDAPUserNamePtr* a *LDAPUserNameOffset* ; volání selže s kódem příčiny MQRC_LDAP_USER_NAME_ERROR, pokud jsou obě nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou typu all-null.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

LDAPUserNameOffset (MQLONG) pro MQAIR

Jedná se o posun v bajtech jména uživatele LDAP od začátku struktury MQAIR.

Posun může být kladný nebo záporný. Je-li hodnota *LDAPUserNameLength* nula, pole se ignoruje.

Můžete použít buď *LDAPUserNamePtr* , nebo *LDAPUserNameOffset* , abyste uvedli jméno uživatele LDAP, ale ne obojí; podrobnosti viz popis pole *LDAPUserNamePtr* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

LDAPUserNameDélka (MQLONG) pro MQAIR

Jedná se o délku (v bajtech) jména uživatele LDAP adresovaného polem *LDAPUserNamePtr* nebo *LDAPUserNameOffset* .

Hodnota musí být v rozsahu nula až MQ_ROZLIŠISHED_NAME_LENGTH. Pokud hodnota není platná, volání selže s kódem příčiny MQRC_LDAP_USER_NAME_LENGTH_ERR.

Pokud daný server LDAP nevyžaduje jméno uživatele, nastavte toto pole na nulu.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

LDAPPassword (MQCHAR32) pro MQAIR

Toto je heslo potřebné pro přístup k serveru CRL LDAP. Pokud je hodnota kratší než délka pole, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku pole.

Pokud server LDAP nevyžaduje heslo, nebo pokud vynecháte jméno uživatele LDAP, *LDAPPassword* musí mít hodnotu null nebo musí být prázdné. Pokud vynecháte jméno uživatele LDAP a parametr *LDAPPassword* nemá hodnotu null nebo není prázdný, volání selže s kódem příčiny MQRC_LDAP_PASSWORD_ERROR.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_LDAP_PASSWORD_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.

OCSPResponderURL (MQCHAR256) pro MQAIR

Pro strukturu MQAIR, která představuje podrobnosti připojení pro odpovídací modul OCSP, toto pole obsahuje URL , na které lze odpovídací modul kontaktovat.

Hodnota tohoto pole je HTTP URL. Toto pole má prioritu nad URL v rozšíření certifikátu AuthorityInfoAccess (AIA).

Hodnota je ignorována, pokud nejsou splněny obě následující podmínky:

- Struktura MQAIR je verze 2 nebo novější (pole Verze je nastaveno na hodnotu MQAIR_VERSION_2 nebo vyšší).
- Pole typu AuthInfoje nastaveno na hodnotu MQAIT_OCSP.

Pokud pole neobsahuje adresu URL HTTP URL ve správném formátu (a není ignorováno), volání MQCONNX se nezdaří s kódem příčiny MQRC_OCSP_URL_ERROR.

V tomto poli se rozlišují malá a velká písmena. Musí začínat řetězcem http:// malými písmeny. Zbytek URL může v závislosti na implementaci serveru OCSP rozlišovat malá a velká písmena.

Toto pole není předmětem převodu dat.

MQBMHO-Volby zpracování vyrovnávací paměti pro zprávu

Struktura MQBMHO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou manipulátory zpráv vytvářeny z vyrovnávacích pamětí. Struktura je vstupní parametr volání MQBUFMH.

Znaková sada a kódování

Data v MQBMHO musí být ve znakové sadě aplikace a kódování aplikace (MQENC_NATIVE).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 468. Pole v MQBMHO

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQBMHO_STRUC_ID	'BMHO'
Verze (číslo verze struktury)	MQBMHO_VERSION_1	1
Volby (volby řídící akci MQBMHO)	MQBMHO_NONE	0

Tabulka 468. Pole v MQBMHO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**

Notes:

1. V programovacím jazyku C se jedná o proměnnou makra.MQBMHO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQBMHO MyBMHO = {MQBMHO_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQBMHO

```
typedef struct tagMQBMHO MQBMHO;
struct tagMQBMHO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQBUFMH */
};
```

Deklarace jazyka COBOL pro objekt MQBMHO

```
**   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.
```

Prohlášení PL/I pro MQBMHO

```
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 */
```

Deklarace High Level Assembler pro objekt MQBMHO

```
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)
```

StrucId (MQCHAR4) pro MQBMHO pro MQBMHO

Jedná se o identifikátor struktury vyrovnávací paměti pro strukturu popisovače zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQBMHO_STRUC_ID.

Hodnota musí být:

MQBMHO_STRUC_ID
Identifikátor vyrovnávací paměti pro strukturu popisovače zprávy.

Pro programovací jazyk C je také definována konstanta MQBMHO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQBMHO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQBMHO pro MQBMHO
Toto je číslo verze vyrovnávací paměti pro strukturu popisovače zprávy. Vždy se jedná o vstupní pole.

Hodnota musí být:

MQBMHO_VERSION_1
Číslo verze pro vyrovnávací paměť pro strukturu popisovače zprávy.

Následující konstanta určuje číslo verze aktuální verze:

MQBMHO_CURRENT_VERSION
Aktuální verze vyrovnávací paměti pro strukturu popisovače zprávy.

Volby (MQLONG) pro MQBMHO
Struktura zpracování vyrovnávací paměti pro zprávu-pole Volby

Hodnota může být následující:

MQBMHO_DELETE_PROPERTIES
Vlastnosti přidané do popisovače zprávy jsou odstraněny z vyrovnávací paměti. Pokud volání selže, nebudou odstraněny žádné vlastnosti.

Výchozí volby: Pokud nepotřebujete popsanou volbu, použijte následující volbu:

MQBMHO_NONE
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQBMHO_DELETE_PROPERTIES.

V 9.3.0 MQBNO-Volby vyvažování

Následující tabulka shrnuje pole ve struktuře.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 469. Pole v MQBNO

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQBNO_STRUC_ID	'BNO¬'
Verze (číslo verze struktury)	MQBNO_VERSION_1	1
ApplicationType (typ volby vyvážení nastavený ve struktuře)	MQBNO_VALTYPE_SIMPLE	0
Časový limit (časový limit, po kterém může nové vyvážení přerušit aktivitu aplikace)	MQBNO_TIMEOUT_AS_DEFAULT	0
BalanceOptions (volby vyvážení nastavené vydávající aplikací)	MQBNO_OPTIONS_NONE	0

Tabulka 469. Pole v MQBNO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C obsahuje proměnná makra MQBNO_DEFAULT hodnoty uvedené v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQBNO MyBNO = {MQBNO_DEFAULT};
```

Deklarace jazyka

C deklarace pro MQBNO

```
typedef struct tagMQBNO MQBNO;
struct tagMQBNO {
  MQCHAR4    StrucId;           /* Structure identifier */
  MQLONG     Version;           /* Structure version number */
  MQLONG     Type;              /* Type of balancing options set in the
                                   structure */
  MQLONG     Timeout;           /* Timeout after which re-balancing might
                                   interrupt application activity */
  MQLONG     BalanceOptions;    /* Balancing options set by the issuing
                                   application */
};
```

Deklarace jazyka COBOL pro MQBNO

```
**   MQBNO structure
  10 MQBNO.
**    Structure identifier
   15 MQBNO-STRUCID          PIC X(4).
**    Structure version number
   15 MQBNO-VERSION          PIC S9(9) BINARY.
**    Type of balancing options set in the structure
   15 MQBNO-TYPE             PIC S9(9) BINARY.
**    Timeout after which re-balancing might interrupt application activity
   15 MQBNO-TIMEOUT          PIC S9(9) BINARY.
**    Balancing options set by the issuing application
   15 MQBNO-BALANCEOPTIONS          PIC S9(9) BINARY.
```

Deklarace PL/I pro MQBNO

```
dcl
 1 MQBNO based,
  3 StrucId          char(4),       /* Structure identifier */
  3 Version          fixed bin(31), /* Structure version number */
  3 Type             fixed bin(31), /* Type of balancing options set in the
                                       structure*/
  3 Timeout          fixed bin(31), /* Timeout after which re-balancing might
                                       interrupt application activity */
  3 BalanceOptions   fixed bin(31), /* Balancing options set by the issuing
                                       application*/
```

Související odkazy

“MQCNO-Volby připojení” na stránce 318
Struktura MQCNO umožňuje aplikaci určit volby související s připojením ke správci front. Struktura je vstupní/výstupní parametr volání MQCONNX.

V 9.3.0 *StrucId (MQCHAR4) pro MQBNO*

Jedná se o identifikátor struktury struktury voleb vyvážení. Vždy se jedná o vstupní pole. Jeho počáteční hodnota je BNO.

Hodnota musí být:

BNO
Identifikátor pro strukturu voleb vyvážení.

Pro programovací jazyk C je definována také konstanta MQBNO_STRUC_ID_ARRAY. Tato konstanta má stejnou hodnotu jako BNO, ale je to pole znaků místo řetězce.

Musíte poskytnout platnou hodnotu pro **StrucId** nebo je vrácena hodnota MQRC_BNO_ERROR.

V 9.3.0 *Verze (MQLONG) pro MQBNO*

Jedná se o číslo verze struktury voleb vyvážení. Vždy se jedná o vstupní pole.

Hodnota musí být:

MQBNO_VERSION_1
Číslo verze pro strukturu voleb vyvážení.

Musíte poskytnout platnou hodnotu pro **Version** nebo je vrácena hodnota MQRC_BNO_ERROR.

V 9.3.0 *ApplicationType (MQLONG) pro MQBNO*

Typ volby vyvážení nastavený ve struktuře.

Možné hodnoty jsou:

MQBNO_BALTYPE_SIMPLE
Jednoduché vyvážení; kromě pravidel popsaných v tématu Opětovné vyvážení aplikací v uniformních klastrechse nepoužívají žádná specifická pravidla.

MQBNO_BALTYPE_REQREP
Vyvažování požadavek-odezva; po každém volání MQPUT je pro zprávu odpovědi očekáváno odpovídající volání MQGET. Vyvažování je zpožděno, dokud není taková zpráva přijata, nebo dokud není překročena hodnota EXPIRACE zprávy požadavku.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQBNO_BALTYPE_SIMPLE.

Musíte zadat pouze jednu hodnotu pro pole **ApplicationType** , nebo se vrátí MQRC_BNO_ERROR.

Poznámka: Další hodnota pro toto pole MQBNO_BALTYPE_RA_MANAGED je vyhrazena pro použití adaptérem prostředků IBM MQ pro prostředí JEE. I když je chybou, aby aplikace zadala tuto hodnotu přímo, může být například nahlášena při dotazování na stav aplikace.

V 9.3.0 *Časový limit (MQLONG) pro MQBNO*

Timeout , po kterém může opětovné vyvažování přerušit aktivitu aplikace.

Možné hodnoty jsou:

MQBNO_TIMEOUT_AS_DEFAULT
Nastavená výchozí hodnota časového limitu.

MQBNO_TIMEOUT_IMMEDIATE
Dojde k okamžitému vypršení časového limitu.

MQBNO_TIMEOUT_NEVER
Nevyskytne se žádný časový limit.

Počáteční hodnota tohoto pole je MQBNO_TIMEOUT_AS_DEFAULT.

Pro pole **Timeout** nebo pro pole MQRC_BNO_ERROR musíte zadat pouze jednu hodnotu z definovaných hodnot nebo hodnotu 0-999999999 sekund.

V 9.3.0 *BalanceOptions (MQLONG) pro MQBNO*

Volby vyvážení nastavené vydávající aplikací.

Možné hodnoty jsou:

MQBNO_OPTIONS_NONE
Nejsou nastaveny žádné volby

MQBNO_OPTIONS_IGNORE_TRANS
Nastavení této volby umožňuje vyvážit aplikace i v případě, že se nacházejí uprostřed transakce.

Počáteční hodnota tohoto pole je MQBNO_OPTIONS_NONE.

Můžete poskytnout libovolnou kombinaci definovaných hodnot pomocí logického znaku nebo znaku pro pole **BalanceOptions** . Jakékoli hodnoty, které nejsou platné, způsobí vrácení MQRC_BNO_ERROR.

MQBO-volby zahájení

Struktura MQBO umožňuje aplikaci určit volby související s vytvořením pracovní jednotky. Struktura je vstupní/výstupní parametr volání MQBEGIN.

Dostupnost

Struktura MQBO je k dispozici na následujících platformách:

- AIX
- IBM i
- Linux
- Windows

Struktura MQBO není k dispozici pro IBM MQ MQI clients.

Znaková sada a kódování

Data v obchodním objektu MQBO musí být ve znakové sadě zadané atributem správce front **CodedCharSetId** a v kódování lokálního správce front zadaném proměnnou MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 470. Pole v MQBO pro MQBO

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	ID_STRUC_MQBO_STRUC_ID	'BO¬¬'
Verze (číslo verze struktury)	MQBO_VERSION_1	1
Volby (volby, které řídí akci MQBEGIN)	MQBO_NONE	0

Tabulka 470. Pole v MQBO pro MQBO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQBO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQBO MyBO = {MQBO_DEFAULT};
```

Deklarace jazyka

Deklarace jazyka C pro objekt MQBO

```
typedef struct tagMQBO MQBO;
struct tagMQBO {
  MQCHAR4  StrucId;  /* Structure identifier */
  MQLONG   Version;  /* Structure version number */
  MQLONG   Options;  /* Options that control the action of MQBEGIN */
};
```

Deklarace jazyka COBOL pro objekt MQBO

```
**   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.
```

Deklarace PL/I pro objekt MQBO

```
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 */
```

Deklarace jazyka Visual Basic pro objekt MQBO

```
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
```

StrucId (MQCHAR4) pro MQBO

Jedná se o identifikátor struktury začátku struktury voleb. Vždy se jedná o vstupní pole. Jeho hodnota je MQBO_STRUC_ID.

Hodnota musí být:

ID_STRUC_MQBO_STRUC_ID
Identifikátor struktury voleb začátku.

Pro programovací jazyk C je definována také konstanta MQBO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQBO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro objekt MQBO

Toto je číslo verze struktury voleb začátku. Vždy se jedná o vstupní pole.

Hodnota musí být:

MQBO_VERSION_1
Číslo verze pro strukturu voleb začátku.

Následující konstanta určuje číslo verze aktuální verze:

MQBO_CURRENT_VERSION
Aktuální verze struktury voleb začátku.

Volby (MQLONG) pro MQBO

Toto pole je vždy vstupní pole. Jeho počáteční hodnota je MQBO_NONE.

Hodnota musí být:

MQBO_NONE
Nejsou uvedeny žádné volby.

MQCBC-Kontext zpětného volání

Struktura MQCBC se používá k určení informací o kontextu, které se předávají funkci zpětného volání. Struktura je vstupní/výstupní parametr volání rutiny spotřebitele zpráv.

Dostupnost

Struktura MQCBC je k dispozici na následujících platformách:

- AIX
- IBM i
- Linux
- Windows
- z/OS

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Verze

Aktuální verze MQCBC je MQCBC_VERSION_2.

Znaková sada a kódování

Data v MQCBC musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , bude struktura ve znakové sadě a kódování klienta.

Pole

Pro strukturu **MQCBC** neexistují žádné počáteční hodnoty. Struktura je předána jako parametr rutině zpětného volání. Správce front inicializuje strukturu; aplikace ji nikdy neinicializují.

Notes:

- V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

- Pro strukturu MQCBC neexistují žádné počáteční hodnoty. Struktura je předána jako parametr rutině zpětného volání. Správce front inicializuje strukturu; aplikace ji nikdy neinicializují.

Tabulka 471. Pole v MQCBC	
Pole	**Popis**
StrucID	Identifikátor struktury
verze	Číslo verze struktury
CallType	Proč byla volána funkce
HOBJ	Popisovač objektu
CallbackArea	Pole pro použití funkce zpětného volání
ConnectionArea	Pole pro použití funkce zpětného volání
CompCode	Kód dokončení
Příčina	Kód příčiny
Stav	Indikace stavu současného spotřebitele
DataLength	Délka zprávy
BufferLength	Délka vyrovnávací paměti zpráv v bajtech
Příznaky	Obecné příznaky
Poznámka: Je-li verze menší než MQCBC_VERSION_2 , bude zbývající pole ignorováno.	
ReconnectDelay	Počet milisekund před pokusem o opětovné připojení

Deklarace jazyka

C prohlášení pro MQCBC

```
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 */
```

Deklarace jazyka COBOL pro MQCBC

```
** 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 **
```

Prohlášení PL/I pro MQCBC

```
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 */
```

Deklarace High Level Assembler pro MQCBC

```
    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)
```

StrucId (MQCHAR4) pro MQCBC

Jedná se o identifikátor struktury struktury kontextu zpětného volání. Vždy se jedná o vstupní pole. Jeho hodnota je MQCBC_STRUC_ID.

Hodnota musí být:

MQCBC_STRUC_ID
Identifikátor pro strukturu kontextu zpětného volání.

Pro programovací jazyk C je definována také konstanta MQCBC_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQCBC_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQCBC
Toto je číslo verze struktury kontextu zpětného volání. Vždy se jedná o vstupní pole.

Hodnota musí být:

MQCBC_VERSION_1
Struktura kontextu zpětného volání verze 1.

Následující konstanta určuje číslo verze aktuální verze:

MQCBC_CURRENT_VERSION
Aktuální verze struktury kontextu zpětného volání.

Funkci zpětného volání je vždy předána nejnovější verze struktury.

CallType (MQLONG) pro MQCBC
Pole obsahující informace o tom, proč byla tato funkce volána; jsou definovány následující hodnoty.

Typy volání doručení zpráv: Tyto typy volání obsahují informace o zprávě. Parametry **DataLength** a **BufferLength** jsou platné pro tyto typy volání.

MQCBCT_MSG_REMOVED
Funkce spotřebitele zpráv byla vyvolána se zprávou, která byla destruktivně odebrána z popisovače objektu.

Je-li hodnota *CompCode* MQCC_WARNING, hodnota pole *Reason* je MQRC_TRUNCATED_MSG_ACCEPTED nebo jeden z kódů označujících problém s převodem dat.

MQCBCT_MSG_NOT_REMOVED
Funkce spotřebitele zpráv byla vyvolána se zprávou, která ještě nebyla destruktivně odebrána z popisovače objektu. Zprávu lze destruktivně odebrat z popisovače objektu pomocí konzoly *MsgToken*.

Zpráva možná nebyla odebrána, protože:

- Volby MQGMO požadovali operaci procházení, MQGMO_BROWSE_ *
- Zpráva je větší než dostupná vyrovnávací paměť a volby MQGMO neuvádějí MQGMO_ACCEPT_TRUNCATED_MSG

Je-li hodnota *CompCode* MQCC_WARNING, hodnota pole *Reason* je MQRC_TRUNCATED_MSG_FAILED nebo jeden z kódů označujících problém s převodem dat.

Typy volání řízení zpětného volání: Tyto typy volání obsahují informace o ovládacím prvku zpětného volání a neobsahují podrobnosti o zprávě. Tyto typy volání jsou vyžadovány pomocí Volby ve struktuře MQCBD.

Parametry **DataLength** a **BufferLength** nejsou pro tyto typy volání platné.

MQCBCT_REGISTER_CALL
Účelem tohoto typu volání je umožnit funkci zpětného volání provést počáteční nastavení.

Funkce zpětného volání je vyvolána okamžitě po registraci zpětného volání, tj. po návratu z volání MQCB pomocí hodnoty pole *Operation* MQOP_REGISTER.

Tento typ volání se používá jak pro spotřebitele zpráv, tak pro obslužné rutiny událostí.

Je-li požadováno, jedná se o první vyvolání funkce zpětného volání.

Hodnota pole *Reason* je MQRC_NONE.

MQCBCT_START_CALL
Účelem tohoto typu volání je umožnit funkci zpětného volání provést při spuštění určité nastavení, například obnovení prostředků, které byly vyčištěny při předchozím zastavení.

Funkce zpětného volání je vyvolána při spuštění připojení pomocí příkazu MQOP_START nebo MQOP_START_WAIT.

Je-li funkce zpětného volání registrována v rámci jiné funkce zpětného volání, je tento typ volání vyvolán při vrácení zpětného volání.

Tento typ volání se používá pouze pro spotřebitele zpráv.

Hodnota pole *Reason* je MQRC_NONE.

MQCBCT_STOP_CALL
Účelem tohoto typu volání je umožnit funkci zpětného volání provést určité vyčištění, když je na chvíli zastavena, například vyčistit další prostředky, které byly získány během příjmu zpráv.

Funkce zpětného volání je vyvolána při zadání volání MQCTL s použitím hodnoty pole *Operation* příkazu MQOP_STOP.

Tento typ volání se používá pouze pro spotřebitele zpráv.

Hodnota pole *Reason* je nastavena tak, aby označovala příčinu zastavení.

MQCBCT_DEREGISTER_CALL
Účelem tohoto typu volání je umožnit funkci zpětného volání provést konečné vyčištění na konci procesu spotřeby. Funkce zpětného volání je vyvolána při:

- Registrace funkce zpětného volání je zrušena pomocí volání MQCB s MQOP_DEREGISTER.
- Fronta je zavřená, což způsobuje implicitní zrušení registrace. V této instanci je funkce zpětného volání předána jako popisovač objektu MQHO_UNUSABLE_HOBJ.
- Volání MQDISC je dokončeno, což způsobí implicitní zavření, a tedy zrušení registrace. V tomto případě není připojení okamžitě odpojeno a žádná probíhající transakce není dosud potvrzena.

Pokud jsou některé z těchto akcí provedeny uvnitř samotné funkce zpětného volání, akce se vyvolá, jakmile se zpětné volání vrátí.

Tento typ volání se používá jak pro spotřebitele zpráv, tak pro obslužné rutiny událostí.

Je-li požadováno, jedná se o poslední vyvolání funkce zpětného volání.

Hodnota pole *Reason* je nastavena tak, aby označovala příčinu zastavení.

MQCBCT_EVENT_CALL
Funkce obslužné rutiny událostí

Funkce obslužné rutiny událostí byla vyvolána bez zprávy, když se správce front nebo připojení zastaví nebo uvede do klidového stavu.

Toto volání lze použít k provedení příslušné akce pro všechny funkce zpětného volání.

Funkce spotřebitele zpráv

Funkce spotřebitele zpráv byla vyvolána bez zprávy, když byla zjištěna chyba (*CompCode* = MQCC_FAILED), která je specifická pro popisovač objektu; například *Reason* code = MQRC_GET_INHIBITED.

Hodnota pole *Reason* je nastavena tak, aby označovala příčinu volání.

MQCBCT_MC_EVENT_CALL

Funkce obslužné rutiny událostí byla vyvolána pro události výběrového vysílání. Obslužná rutina událostí je odesílána IBM MQ Události výběrového vysílání namísto 'normálních' IBM MQ událostí.

Další informace o MQCBCT_MC_EVENT_CALL naleznete v tématu Vykazování výjimek výběrového vysílání.

Hobj (MQHOBJ) pro MQCBC
Jedná se o popisovač objektu pro volání na spotřebitele zpráv.

Pro obslužnou rutinu událostí je tato hodnota MQHO_NONE

Aplikace může použít tento popisovač a token zprávy v bloku Volby získání zprávy k získání zprávy, pokud zpráva nebyla odebrána z fronty.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQHO_UNUSABLE_HOBJ.

CallbackArea (MQPTR) pro MQCBC
Toto pole je k dispozici pro použití funkcí zpětného volání.

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a předává se beze změny z pole CallbackArea ve struktuře MQCBD, což je parametr volání MQCB použitého k definování funkce zpětného volání.

Změny v souboru *CallbackArea* jsou zachovány v rámci vyvolání funkce zpětného volání pro položku *HObj*. Toto pole není sdíleno s funkcemi zpětného volání pro jiné manipulátory.

Toto je vstupní/výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

ConnectionArea (MQPTR) pro MQCBC
Toto pole je k dispozici pro použití funkcí zpětného volání.

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a předává se beze změny z pole ConnectionArea ve struktuře MQCTLO, což je parametr volání MQCTL použitého k řízení funkce zpětného volání.

Veškeré změny provedené v tomto poli funkcemi zpětného volání jsou zachovány v rámci vyvolání funkce zpětného volání. Tuto oblast lze použít k předávání informací, které mají být sdíleny všemi funkcemi zpětného volání. Na rozdíl od *CallbackArea*je tato oblast společná pro všechna zpětná volání pro manipulátor připojení.

Toto je vstupní a výstupní pole. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

CompCode (MQLONG) pro MQCBC
Toto pole je kód dokončení. Označuje, zda se vyskytly problémy se spotřebou zprávy.

Hodnota je jedna z následujících:

MQCC_OK
Úspěšné dokončení

MQCC_VAROVÁNÍ
Varování (částečné dokončení)

MQCC_FAILED
Volání selhalo

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQCC_OK.

Příčina (MQLONG) pro MQCBC
Toto je kód příčiny, který kvalifikuje *CompCode*.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQRC_NONE.

Stav (MQLONG) pro MQCBC
Indikace stavu aktuálního spotřebitele. Toto pole má největší hodnotu pro aplikaci, když je funkci spotřebitele předán nenulový kód příčiny.

Toto pole můžete použít ke zjednodušení programování aplikací, protože nemusíte kódovat chování pro každý kód příčiny.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQCS_NONE.

Tabulka 472.		
Stav	**Akce správce front**	**Hodnota konstanty**
MQCS_NONE Tento kód příčiny představuje normální volání bez dalších informací o příčině.	Není; jedná se o normální operaci.	0
MQCS_SUSPENDED_TEMPORARY Tyto kódy příčiny představují dočasné podmínky.	Rutina zpětného volání je volána k ohlášení podmínky a poté pozastavena. Po určité době se může systém znovu pokusit o operaci, což může vést k opětovnému vytvoření stejné podmínky.	1
MQCS_SUSPENDED_USER_ACTION Tyto kódy příčiny představují podmínky, kdy musí zpětné volání provést akci k vyřešení podmínky.	Odběratel je pozastaven a je volána rutina zpětného volání pro ohlášení podmínky. Rutina zpětného volání by měla vyřešit podmínku, je-li to možné, a buď POKRAČOVAT, nebo ukončit připojení.	2
MQCS_SUSPENDED Tyto kódy příčiny představují selhání, která zabraňují dalším zpětným voláním zpráv.	Správce front automaticky pozastaví funkci zpětného volání. Je-li funkce zpětného volání obnovena, je pravděpodobné, že znovu obdrží stejný kód příčiny.	3
MQCS_STOPPED Tyto kódy příčiny představují konec spotřeby zpráv.	Doručeno obslužné rutině výjimek a zpětným voláním, která určila MQCBDO_STOP_CALL. Nelze přijímat žádné další zprávy.	4

DataLength (MQLONG) pro MQCBC

Jedná se o délku dat aplikace ve zprávě v bajtech. Je-li hodnota nula, znamená to, že zpráva neobsahuje žádná data aplikace.

Pole DataLength obsahuje délku zprávy, ale ne nutně délku dat zprávy předaných spotřebiteli. Je možné, že zpráva byla oříznuta. Pomocí pole ReturnedLength v MQGMO určete, kolik dat bylo skutečně předáno spotřebiteli.

Pokud kód příčiny označuje, že zpráva byla oříznuta, můžete pomocí pole DataLength určit, jak velká je skutečná zpráva. To vám umožňuje určit velikost vyrovnávací paměti požadované pro uložení dat zprávy a poté zadat volání MQCB pro aktualizaci MaxMsgLength s odpovídající hodnotou.

Je-li zadána volba MQGMO_CONVERT, může být převedená zpráva větší než hodnota vrácená pro DataLength. V takových případech aplikace pravděpodobně potřebuje zadat volání MQCB, aby aktualizovala MaxMsgLength tak, aby byla větší než hodnota vrácená správcem front pro DataLength.

Chcete-li se vyhnout problémům s oříznutím zpráv, zadejte hodnotu MaxMsgLength jako MQCBD_FULL_MSG_LENGTH. To způsobí, že správce front přidělí vyrovnávací paměť pro celou délku zprávy po převodu dat. Uvědomte si však, že i když je tato volba uvedena, je stále možné, že není k dispozici dostatek paměti pro správné zpracování požadavku. Aplikace by měly vždy zkontrolovat vrácený kód příčiny. Pokud například není možné přidělit dostatek paměti pro převod zprávy, zprávy se vrátí do aplikace nepřevedené.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny událostí.

BufferLength (MQLONG) pro MQCBC

Toto pole je délka vyrovnávací paměti zpráv v bajtech, která byla předána této funkci.

Vyrovnávací paměť může být větší než hodnota MaxMsgLength definovaná pro spotřebitele a hodnota ReturnedLength v MQGMO.

Skutečná délka zprávy je uvedena v poli DataLength .

Aplikace může používat celou vyrovnávací paměť pro své vlastní účely po dobu trvání funkce zpětného volání.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny výjimek.

Příznaky (MQLONG) pro MQCBC

Příznaky obsahující informace o tomto spotřebiteli.

Je definována následující volba:

MQCBCF_READA_BUFFER_EMPTY

Tento příznak může být vrácen, pokud předchozí volání MQCLOSE pomocí volby MQCO_QUIESCE selhalo s kódem příčiny MQRC_READ_AHEAD_MSGS.

Tento kód označuje, že je vrácena poslední dopředná zpráva a že vyrovnávací paměť je nyní prázdná. Pokud aplikace vydá další volání MQCLOSE pomocí volby MQCO_QUIESCE), bude úspěšná.

Všimněte si, že aplikaci není zaručeno, že obdrží zprávu s touto sadou příznaků, protože stále mohou existovat zprávy ve vyrovnávací paměti pro čtení napřed, které neodpovídají aktuálním kritériím výběru. V této instanci je funkce spotřebitele vyvolána s kódem příčiny MQRC_HOBJ_QUIESCED.

Pokud je vyrovnávací paměť dopředného čtení zcela prázdná, je spotřebitel vyvolán s příznakem MQCBCF_READA_BUFFER_EMPTY a kódem příčiny MQRC_HOBJ_QUIESCED_NO_MSGS.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny událostí.

ReconnectDelay (MQLONG) pro MQCBC

ReconnectDelay označuje, jak dlouho bude správce front čekat, než se pokusí znovu připojit. Pole může být upraveno obslužnou rutinou událostí, aby se zcela změnila prodleva nebo zastavilo opětovné připojení.

Pole `ReconnectDelay` použijte pouze v případě, že hodnota pole `Příčina` v kontextu zpětného volání je `MQRC_RECONNECTING`.

Při zadávání do obslužné rutiny událostí je hodnota `ReconnectDelay` počet milisekund, po které bude správce front čekat před provedením pokusu o opětovné připojení. Tabulka 473 na stránce 290 obsahuje seznam hodnot, které lze nastavit pro úpravu chování správce front při návratu z obslužné rutiny událostí.

Tabulka 473. `ReconnectDelay` hodnoty

Název	Hodnota	Popis
`MQRD_NO_RECONNECT`	`-1`	Neprovádět žádné další pokusy o opětovné připojení. Aplikaci se vrátí chyba.
`MQRD_NO_DELAY`	`0`	Pokuste se okamžitě znovu připojit.
`Milliseconds`	`>0`	Před opětovným pokusem o připojení počkejte na tento počet milisekund.

MQCBD-Deskriptor zpětného volání

Struktura MQCBD se používá k určení funkce zpětného volání a voleb, které řídí její použití správcem front. Struktura je vstupní parametr volání MQCB.

Dostupnost

Struktura MQCBD je k dispozici na následujících platformách:

- AIX
- IBM i

- Linux
- Windows
- z/OS

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Verze

Aktuální verze MQCBD je MQCBD_VERSION_1.

Znaková sada a kódování

Data v MQCBD musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 474. Pole v MQCBD

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucID (identifikátor struktury)	MQCBD_STRUC_ID	'CBD¬'
Verze (číslo verze struktury)	MQCBD_VERSION_1	1
CallbackType (typ funkce zpětného volání)	MQCBT_MESSAGE_CONSUMER	1
Volby (volby řídící spotřebu zpráv)	MQCBDO_NONE	0
CallbackArea (pole pro použití funkce zpětného volání)	Není	Prázdný ukazatel nebo prázdné znaky
CallbackFunction (zda je funkce vyvolána jako volání rozhraní API)	Není	Prázdný ukazatel nebo prázdné znaky
CallbackName (zda je funkce vyvolána jako dynamicky propojený program)	Není	Prázdný řetězec nebo mezery
MaxMsgDélka (délka nejdelší zprávy, kterou lze číst)	MQCBD_FULL_MSG_LENGTH	-1

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje hodnotu null v programovacím jazyku C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQCBD_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCBD MyCBD = {MQCBD_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQCBD

```
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 */
 };
```

Deklarace jazyka COBOL pro MQCBD

```
** 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.
```

Prohlášení PL/I pro MQCBD

```
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 */
```

StrucId (MQCHAR4) pro MQCBD

Jedná se o identifikátor struktury struktury deskriptoru zpětného volání. Vždy se jedná o vstupní pole. Jeho hodnota je MQCBD_STRUC_ID.

Hodnota musí být:

MQCBD_STRUC_ID
Identifikátor pro strukturu deskriptoru zpětného volání.

Pro programovací jazyk C je také definována konstanta MQCBD_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQCBD_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQCBD

Jedná se o číslo verze struktury deskriptoru zpětného volání. Vždy se jedná o vstupní pole.

Hodnota musí být:

MQCBD_VERSION_1
Struktura deskriptoru zpětného volání verze 1.

Následující konstanta určuje číslo verze aktuální verze:

MQCBD_CURRENT_VERSION
Aktuální verze struktury deskriptoru zpětného volání.

CallbackType (MQLONG) pro MQCBD

Struktura deskriptoru zpětného volání-pole CallbackType

Jedná se o typ funkce zpětného volání. Hodnota musí být jedna z následujících:

MQCBT_MESSAGE_CONSUMER
Definuje toto zpětné volání jako funkci spotřebitele zpráv.

Funkce zpětného volání spotřebitele zpráv je volána, když je zpráva splňující zadaná kritéria výběru k dispozici na popisovači objektu a připojení je spuštěno.

Obslužná rutina MQCBT_EVENT_HANDLER
Definuje toto zpětné volání jako rutinu asynchronních událostí; není řízeno tak, aby spotřebovávalo zprávy pro manipulátor.

Ve volání MQCB, které definuje obslužnou rutinu událostí, není hodnota *Hobj* vyžadována a je-li zadána, je ignorována.

Obslužná rutina událostí je volána pro podmínky, které ovlivňují celé prostředí spotřebitele zpráv. Funkce spotřebitele je vyvolána bez zprávy, dojde-li k události, například k zastavení správce front nebo připojení, nebo k uvedení do klidového stavu. Není volána pro podmínky, které jsou specifické pro jednoho spotřebitele zpráv, například MQRC_GET_INHIBITED.

Události jsou doručeny do aplikace bez ohledu na to, zda je připojení spuštěno nebo zastaveno, s výjimkou následujících prostředí:

- CICS v prostředí z/OS
- aplikace bez podprocesů

Pokud volající nepředá jednu z těchto hodnot, volání selže s kódem *Reason* MQRC_CALLBACK_TYPE_ERROR

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCBT_MESSAGE_CONSUMER.

Volby (MQLONG) pro MQCBD

Struktura deskriptoru zpětného volání-pole Volby

Můžete uvést jednu nebo více těchto voleb. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

MQCBDO_FAIL_IF_QUIESCING
Volání MQCB se nezdaří, pokud se správce front nachází ve stavu uvedení do klidového stavu.

V systému z/OStato volba také vynutí selhání volání MQCB, pokud je připojení (pro aplikaci CICS nebo IMS) v klidovém stavu.

Zadejte volbu MQGMO_FAIL_IF_QUIESCING ve volbách MQGMO předaných pro volání MQCB, abyste způsobili oznámení spotřebitelům zpráv při jejich uvedení do klidového stavu.

Volby řízení: Následující volby řídí, zda je volána funkce zpětného volání bez zprávy, když se změní stav spotřebitele:

MQCBDO_REGISTER_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT_REGISTER_CALL.

MQCBDO_START_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT_START_CALL.

MQCBDO_STOP_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT_STOP_CALL.

MQCBDO_DEREGISTER_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT_DEREGISTER_CALL.

MQCBDO_EVENT_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT_EVENT_CALL.

MQCBDO_MC_EVENT_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT__MC_EVENT_CALL.

Další podrobnosti o těchto typech volání viz CallType .

Výchozí volba: Pokud nepotřebujete žádnou z popsaných voleb, použijte následující volbu:

MQCBDO_NONE
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

MQCBDO_NONE je definován jako pomůcka pro dokumentaci k programu. Není zamýšleno, aby byla tato volba použita společně s jinou, ale protože má nulovou hodnotu, nelze takové použití zjistit.

Toto je vstupní pole. Počáteční hodnota pole *Options* je MQCBDO_NONE.

CallbackArea (MQPTR) pro MQCBD
Struktura deskriptoru zpětného volání-pole CallbackArea

Jedná se o pole, které je k dispozici pro použití funkcí zpětného volání.

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a je předán beze změny z pole CallbackArea ve struktuře MQCBC, což je parametr deklarace funkce zpětného volání.

Hodnota se používá pouze v systému *Operation* , který má hodnotu MQOP_REGISTER, bez aktuálně definovaného zpětného volání, nenahradí předchozí definici.

Toto je vstupní a výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

CallbackFunction (MQPTR) pro MQCBD
Struktura deskriptoru zpětného volání-pole CallbackFunction

Funkce zpětného volání je vyvolána jako volání funkce.

Toto pole slouží k určení ukazatele na funkci zpětného volání.

Musíte zadat buď *CallbackFunction* , nebo *CallbackName*. Zadáte-li obojí, vrátí se kód příčiny MQRC_CALLBACK_ROUTINE_ERROR.

Není-li nastavena hodnota *CallbackName* ani *CallbackFunction* , volání se nezdaří s kódem příčiny MQRC_CALLBACK_ROUTINE_ERROR.

Tato volba není podporována v následujícím prostředí: Programovací jazyky a kompilátory, které nepodporují odkazy na ukazatel funkce. V takových situacích volání selže s kódem příčiny MQRC_CALLBACK_ROUTINE_ERROR.

z/OS V systému z/OSmusí funkce očekávat, že bude volána s konvencemi sestavení operačního systému. Například v programovacím jazyku C zadejte:

```
#pragma linkage(MQCB_FUNCTION,OS)
```

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

Poznámka: Při použití produktu CICS s produktem IBM WebSphere MQ 7.0.1je asynchronní spotřeba podporována, pokud:

- Apar PK66866 se použije na CICS TS 3.2
- Apar PK89844 se použije na CICS TS 4.1

CallbackName (MQCHAR128) pro MQCBD

Struktura deskriptoru zpětného volání-pole CallbackName

Funkce zpětného volání je vyvolána jako dynamicky propojený program.

Musíte zadat buď *CallbackFunction* , nebo *CallbackName*. Zadáte-li obojí, vrátí se kód příčiny MQRC_CALLBACK_ROUTINE_ERROR.

Není-li nastavena hodnota *CallbackName* ani *CallbackFunction* , volání se nezdaří s kódem příčiny MQRC_CALLBACK_ROUTINE_ERROR.

Modul je načten při registraci první rutiny zpětného volání, která má být použita, a uvolněn při zrušení registrace poslední rutiny zpětného volání.

S výjimkou, kde je uvedeno v následujícím textu, je název v poli zarovnán vlevo bez vložených mezer; samotný název je vyplněn mezerami na délku pole. V popisech, které následují, hranaté závorky ([]) označují nepovinné informace:

IBM i

Název zpětného volání může mít jeden z následujících formátů:

- Knihovna "/" Program
- Knihovna "/" ServiceProgram "("FunctionName")"

Například `MyLibrary/MyProgram(MyFunction)`.

Název knihovny může být *LIBL. Názvy knihoven i programů jsou omezeny na maximálně 10 znaků.

AIX and Linux

Název zpětného volání je název dynamicky zaveditelného modulu nebo knihovny s příponou názvu funkce nacházející se v této knihovně. Název funkce musí být uzavřen v závorkách. Název knihovny může mít volitelně předponu s cestou k adresáři:

```
[path]library(function)
```

Není-li cesta uvedena, použije se systémová vyhledávací cesta.

Délka názvu je omezena na maximálně 128 znaků.

Windows

Název zpětného volání je název knihovny dynamického propojení doplněný názvem funkce nacházející se v této knihovně. Název funkce musí být uzavřen v závorkách. Název knihovny může být volitelně uvozen cestou k adresáři a jednotkou:

```
[d:][path]library(function)
```

Pokud jednotka a cesta nejsou uvedeny, použije se systémová vyhledávací cesta.

Délka názvu je omezena na maximálně 128 znaků.

z/OS

Název zpětného volání je název zaváděcího modulu, který je platný pro specifikaci parametru EP makra LINK nebo LOAD.

Délka názvu je omezena na maximálně 8 znaků.

z/OS CICS

Název zpětného volání je název zaváděcího modulu, který je platný pro specifikaci parametru PROGRAM makra příkazu EXEC CICS LINK.

Délka názvu je omezena na maximálně 8 znaků.

Program lze definovat jako vzdálený pomocí volby REMOTESYTEM nainstalované definice PROGRAM nebo pomocí dynamického směrovacího programu.

Vzdálená oblast CICS musí být připojena k produktu IBM MQ , pokud má program používat volání rozhraní API IBM MQ . Všimněte si však, že pole Hobj ve struktuře MQCBC není ve vzdáleném systému platné.

Dojde-li při pokusu o načtení souboru *CallbackName*k selhání, vrátí se aplikaci jeden z následujících kódů chyb:

- MQRC_MODULE_NOT_FOUND
- MQRC_MODULE_INVALID
- MQRC_MODULE_ENTRY_NOT_FOUND

Do protokolu chyb se také zapíše zpráva obsahující název modulu, pro který byl proveden pokus o načtení, a kód příčiny selhání z operačního systému.

Toto je vstupní pole. Počáteční hodnota tohoto pole je prázdný řetězec nebo mezery.

MaxMsgDélka (MQLONG) pro MQCBD

Jedná se o délku nejdelší zprávy v bajtech, kterou lze načíst z manipulátoru a předat rutině zpětného volání. Struktura deskriptoru zpětného volání-pole MaxMsgLength

Pokud má zpráva delší délku, rutina zpětného volání obdrží *MaxMsgLength* bajtů zprávy a kód příčiny:

- MQRC_TRUNCATED_MSG_FAILED nebo
- MQRC_TRUNCATED_MSG_ACCEPTED, pokud jste zadali MQGMO_ACCEPT_TRUNCATED_MSG.

Skutečná délka zprávy je dodána v poli DataLength struktury MQCBC.

Je definována následující speciální hodnota:

MQCBD_FULL_MSG_LENGTH
Délka vyrovnávací paměti je systémem upravena tak, aby vracela zprávy bez oříznutí.

Pokud není k dispozici dostatek paměti pro přidělení vyrovnávací paměti pro přijetí zprávy, systém volá funkci zpětného volání s kódem příčiny MQRC_STORAGE_NOT_AVAILABLE.

Pokud například požadujete převod dat a není k dispozici dostatek paměti pro převod dat zprávy, nepřevedená zpráva se předá do funkce zpětného volání.

Toto je vstupní pole. Počáteční hodnota pole *MaxMsgLength* je MQCBD_FULL_MSG_LENGTH.

MQCHARV-Řetězec délky proměnné

Použijte strukturu MQCHARV k popisu řetězce proměnné délky.

Dostupnost

Struktura MQCHARV je k dispozici na následujících platformách:

- AIX
- IBM i
- Linux
- Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Znaková sada a kódování

Data v MQCHARV musí být v kódování lokálního správce front, které je dáno hodnotou MQENC_NATIVE a znakovou sadou pole VSCCSID v rámci struktury. Pokud je aplikace spuštěna jako klient produktu MQ , musí být struktura v kódování klienta. Některé znakové sady mají reprezentaci, která závisí na kódování. Je-li jednou z těchto znakových sad identifikátor VSCCSID, bude použito stejné kódování jako u ostatních

polí v tabulce MQCHARV. Znaková sada identifikovaná pomocí VSCCSID může být dvoubajtová znaková sada (DBCS).

Použití

Struktura MQCHARV adresuje data, která mohou být nesouvislá se strukturou, která ji obsahuje. Pro adresování těchto dat lze použít pole deklarovaná s datovým typem ukazatele. Uvědomte si, že COBOL nepodporuje datový typ ukazatele ve všech prostředích. Z tohoto důvodu lze data adresovat také pomocí polí, která obsahují posunutí dat od začátku struktury obsahující MQCHARV.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 475. Pole v MQCHARV

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
VSPtr (ukazatel na řetězec proměnné délky)	Není	Nulový ukazatel nebo nulový počet bajtů.
VSOffset (posun v bajtech řetězce proměnné délky od začátku struktury, která obsahuje tuto strukturu MQCHARV)	Není	0
VSBufSize (velikost vyrovnávací paměti adresované polem VSPtr nebo VSOffset v bajtech)	MQVS_USE_VSLENGTH	0
VSLength (délka řetězce proměnné délky v bajtech adresovaného polem VSPtr nebo VSOffset)	Není	0
VSCCSID (identifikátor znakové sady řetězce proměnné délky adresovaný polem VSPtr nebo VSOffset)	MQCCSI_APPL	-3

Poznámka: V programovacím jazyku C obsahuje proměnná makra MQCHARV_DEFAULT hodnoty uvedené v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQCHARV MyVarStr = {MQCHARV_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQCHARV

```
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 */
};
```

Deklarace jazyka COBOL pro 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.
```

Poznámka: Chcete-li portovat aplikaci v jazyce COBOL mezi prostředími, musíte zjistit, zda je datový typ ukazatele k dispozici ve všech zamýšlených prostředích. Pokud ne, musí aplikace adresovat data pomocí polí offsetu místo polí ukazatele. V prostředích, kde nejsou podporovány ukazatele, můžete pole ukazatele deklarovat jako bajtové řetězce odpovídající délky, přičemž počáteční hodnota je bajtový řetězec s hodnotou typu all-null. Tuto počáteční hodnotu neměňte, pokud používáte pole offsetu. Jedním ze způsobů, jak to provést bez změny dodaných kopírovacích knih, je použít následující:

```
COPY CMQCHRVV REPLACING POINTER BY ==BINARY PIC S9(9)==.
```

kde CMQCHRVV lze vyměnit za knihu kopií, která má být použita.

Prohlášení PL/I pro MQCHARV

```
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 */
```

Deklarace High Level Assembler pro MQCHARV

```
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)
```

VSPtr (MQPTR) pro MQCHARV

Jedná se o ukazatel na řetězec proměnné délky.

Můžete použít buď pole VSPtr, nebo VSOffset, abyste uvedli řetězec proměnné délky, ale ne obojí.

Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

VSOffset (MQLONG) pro MQCHARV

Posun může být kladný nebo záporný. Můžete použít buď pole VSPtr, nebo VSOffset, abyste uvedli řetězec proměnné délky, ale ne obojí. Posun v bajtech řetězce proměnné délky od začátku MQCHARV nebo struktury, která jej obsahuje.

Je-li struktura MQCHARV vložena do jiné struktury, je tato hodnota posunutím řetězce proměnné délky od začátku struktury, která obsahuje tuto strukturu MQCHARV, v bajtech. Není-li struktura MQCHARV vložena do jiné struktury, například je-li zadána jako parametr pro volání funkce, je posun relativní ke začátku struktury MQCHARV.

Počáteční hodnota tohoto pole je 0.

VSBufSize (MQLONG) pro MQCHARV

Velikost vyrovnávací paměti adresované polem VSPtr nebo VSOffset v bajtech.

Je-li struktura MQCHARV použita jako výstupní pole pro volání funkce, musí být toto pole inicializováno s uvedenou délkou vyrovnávací paměti. Pokud je hodnota VSLength větší než VSBufSize , pak se volajícímu ve vyrovnávací paměti vrátí pouze VSBufSize bajtů dat.

Tato hodnota musí být větší nebo rovna nule nebo následující speciální hodnota, která je rozpoznána:

MQVS_USE_VSLENGTH
Je-li zadán, je délka vyrovnávací paměti převzata z pole VSLlength ve struktuře MQCHARV. Tuto hodnotu nepoužívejte, pokud používáte strukturu jako výstupní pole a je poskytnuta vyrovnávací paměť.

Toto je počáteční hodnota tohoto pole.

VSLength (MQLONG) pro MQCHARV

Délka řetězce proměnné délky v bajtech adresovaného polem VSPtr nebo VSOffset.

Počáteční hodnota tohoto pole je 0. Hodnota musí být větší nebo rovna nule nebo následující speciální hodnota, která je rozpoznána:

MQVS_NULL_UKONČENO

Není-li zadána hodnota MQVS_NULL_TERMINATED, budou jako součást řetězce zahrnuty bajty VSLength. Jsou-li přítomny znaky null, neoddělují řetězec.

Je-li zadána hodnota MQVS_NULL_TERMINATED, je řetězec oddělen první hodnotou Null zjištěnou v řetězci. Hodnota null sama o sobě není zahrnuta jako součást tohoto řetězce.

Poznámka: Znak null použitý k ukončení řetězce, pokud je zadána hodnota MQVS_NULL_TERMINATE, má hodnotu null z kódové sady určené hodnotou VSCCSID.

Například v UTF-16 (CCSID 1200, 13488 a 17584) se jedná o dvoubajtové kódování Unicode, kde je hodnota null reprezentována 16bitovým číslem všech nul. V souboru UTF-16 je běžné najít jednotlivé bajty nastavené na nulu, které jsou součástí znaků (například 7bitové znaky ASCII), ale řetězce budou ukončeny pouze v případě, že jsou na sudé hranici bajtů nalezeny dva bajty 'nula'. Je možné získat dva 'nula' bajty na liché hranici, když jsou každou částí platných znaků. Například x '01' x '00 x' 00 'x' 30 ' představuje dva platné znaky Unicode a neukončuje řetězec hodnotou null.

VSCCSID (MQLONG) pro MQCHARV

Jedná se o identifikátor znakové sady řetězce proměnné délky adresovaného polem **VSPtr** nebo **VSOffset** .

Počáteční hodnota tohoto pole je *MQCCSI_APPL*, která je definována produktem MQ a označuje, že by měla být změněna na skutečný identifikátor znakové sady aktuálního procesu. V důsledku toho není hodnota konstanty *MQCCSI_APPL* nikdy přidružena k řetězci s proměnnou délkou.

Počáteční hodnotu tohoto pole lze změnit definováním jiné hodnoty pro konstantu *MQCCSI_APPL* pro vaši kompilační jednotku. To, jak to provedete, závisí na programovacím jazyku vaší aplikace.

z/OS V systémech z/OS je výchozí aplikace CCSID používaná produktem *MQCCSI_APPL* definována takto:

- Pro dávkové aplikace LE používající rozhraní DLL je výchozí hodnota CODESET přidružená k aktuálnímu národnímu prostředí v době vydání **MQCONN** (výchozí hodnota je 1047).
- Pro dávkové aplikace LE svázané s jedním z dávkových stubů MQ je výchozí hodnotou hodnota CODESET přidružená k aktuálnímu národnímu prostředí v době prvního volání MQI vydaného po **MQCONN** (výchozí hodnota je 1047).
- V případě dávkových aplikací jiných než LE spuštěných v podprocesu z/OS UNIX System Services je výchozí hodnotou hodnota THLICCSID v době prvního volání MQI vydaného po **MQCONN** (výchozí hodnota je 1047).
- U ostatních dávkových aplikací je výchozí hodnotou hodnota CCSID správce front.

Opětovná definice MQCCSI_APPL

Následující příklady ukazují, jak lze přepsat hodnotu MQCCSI_APPL v různých programovacích jazycích. Můžete změnit hodnotu MQCCSI_APPL, čímž odeberete potřebu nastavit identifikátor VSCCSID pro každý řetězec s proměnnou délkou odděleně. V těchto příkladech je CCSID nastaven na 1208; změňte jej na požadovanou hodnotu. Tato hodnota se stane výchozí hodnotou, kterou můžete přepsat nastavením identifikátoru VSCCSID v libovolné specifické instanci MQCHARV.

Využití jazyka C

```
#define MQCCSI_APPL 1208
#include <cmqc.h>
```

Použití jazyka COBOL

```
COPY CMQXYZV REPLACING -3 BY 1208.
```

Použití PL/I

```
%MQCCSI_APPL = '1208';
%include syslib(cmqp);
```

High Level Assembler využití

```
MQCCSI_APPL EQU 1208
CMQA LIST=NO
```

Záhlaví MQCIH- CICS bridge

Struktura MQCIH popisuje informace záhlaví pro zprávu odeslanou do CICS přes CICS bridge.

Pro libovolnou podporovanou platformu IBM MQ můžete vytvořit a přenést zprávu, která obsahuje strukturu MQCIH, ale pouze IBM MQ for z/OS správce front může použít CICS bridge. Proto, aby se zpráva dostala do produktu CICS ze správce front jiného nežz/OS , musí síť správců front obsahovat alespoň jednoho správce front z/OS , jehož prostřednictvím může být zpráva směrována.

Všechny verze produktu CICS podporované produktem IBM MQ 9.0.0a novější používají dodanou verzi mostu CICS . Další informace o konfiguraci adaptéru IBM MQ CICS a komponent produktu IBM MQ CICS bridge naleznete v části Konfigurace připojení k produktu MQ v dokumentaci k produktu CICS .

Dostupnost

Struktura MQCIH je k dispozici na následujících platformách:

- AIX
- Linux
- Windows
- z/OS

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Název formátu

MQFMT_CICS:

Verze

Aktuální verze MQCIH je MQCIH_VERSION_2. Pole, která existují pouze v novější verzi struktury, jsou identifikována jako taková v popisech, které následují.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi MQCIH s počáteční hodnotou pole *Version* nastavenou na MQCIH_VERSION_2.

Znaková sada a kódování

Pro znakovou sadu a kódování použité pro strukturu MQCIH a data zpráv aplikace platí zvláštní podmínky:

- Aplikace, které se připojují ke správci front vlastnícímu frontu CICS bridge , musí poskytovat strukturu MQCIH, která je ve znakové sadě a kódování správce front. Důvodem je skutečnost, že v tomto případě není proveden převod dat struktury MQCIH.
- Aplikace, které se připojují k jiným správcům front, mohou poskytovat strukturu MQCIH, která je v libovolné z podporovaných znakových sad a kódování; přijímající agent kanálu zpráv připojený ke správci front, který vlastní frontu CICS bridge , převádí strukturu MQCIH.
- Data zprávy aplikace následující po struktuře MQCIH musí být ve stejné znakové sadě a kódování jako struktura MQCIH. Nemůžete použít pole *CodedCharSetId* a *Encoding* ve struktuře MQCIH k uvedení znakové sady a kódování dat zprávy aplikace.

  Chcete-li převést data zprávy aplikace, která nejsou jedním z vestavěných formátů podporovaných správcem front, musíte poskytnout uživatelskou proceduru pro převod dat.

Pouľití

Pokud aplikace vyžaduje hodnoty, které jsou stejné jako počáteční hodnoty zobrazené v souboru Tabulka 477 na stránce 302, a most je spuštěn s AUTH=LOCAL nebo AUTH=IDENTIFY, můžete ve zprávě vynechat strukturu MQCIH. Ve všech ostatních případech musí být struktura přítomna.

Most přijímá strukturu MQCIH version-1 nebo version-2 , ale pro transakce 3270 musíte použít strukturu version-2 .

Aplikace musí zajistit, aby pole dokumentovaná jako pole požadavku měla ve zprávě odeslané do mostu odpovídající hodnoty; tato pole jsou vstupní do mostu.

Pole dokumentovaná jako pole odezvy jsou nastavena pomocí CICS bridge ve zprávě odpovědi, kterou most odesílá aplikaci. Informace o chybě jsou vráceny v polích *ReturnCode*, *Function*, *CompCode*, *Reason*a *AbendCode* , ale ne všechny jsou nastaveny ve všech případech. Následující tabulka zobrazuje, která pole jsou nastavena pro různé hodnoty *ReturnCode*.

Tabulka 476. Obsah polí s informacemi o chybách ve struktuře MQCIH pro MQCIH

ReturnCode	***Function***	***CompCode***	***Reason***	***AbendCode***
MQCRC_OK	-	-	-	-
MQCRC_BRIDGE_ERROR	-	-	MQFB_CICS_ *	-
MQCRC_MQ_API_ERROR MQCRC_BRIDGE_TIMEOUT	Název volání 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

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 477. Pole v MQCIH pro MQCIH

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQCIH_STRUC_ID	'CIH¬'
Verze (číslo verze struktury)	MQCIH_VERSION_2	2
StrucLength (délka struktury MQCIH)	MQCIH_LENGTH_2	180
Kódování (vyhrazeno)	Není	0
CodedCharSetId (vyhrazeno)	Není	0
Formát (MQ název formátu dat, která následují za MQCIH)	MQFMT_NONE	Mezery
Příznaky (příznaky)	MQCIH_NONE	0
ReturnCode (návratový kód z mostu)	MQCRC_OK	0
CompCode (MQ kód dokončení nebo CICS EIBRESP)	MQCC_OK	0
Příčina (MQ kód příčiny nebo zpětné vazby nebo CICS EIBRESP2)	MQRC_NONE	0
UOWControl (řízení jednotky práce)	MQCUOWC_ONLY	273
GetWaitInterval (interval čekání pro volání MQGET vydané úlohou mostu)	VÝCHOZÍ-MQCGWI_DEFAULT	-2
LinkType (typ odkazu)	MQCLT_PROGRAM	1
OutputData (délka výstupních dat COMMAREA)	MQCODL_AS_INPUT	-1
FacilityKeepČas (čas uvolnění prostředku mostu)	Není	0
ADSDescriptor (odeslat/přijmout deskriptor ADS)	MQCADSD_NONE	0
ConversationalTask (zda může být úloha konverzační)	MQCCT_NO	0
TaskEndStav (stav na konci úlohy)	MQCTES_NOSYNC	0
Prostředek (token prostředku mostu)	MQCFAC_NONE	Hodnoty null
Funkce (název voláníMQ nebo funkce CICS EIBFN)	MQCFUNC_NONE	Mezery
AbendCode (kód nestandardního ukončení)	Není	Mezery
Authenticator (heslo nebo přístupový prvek)	Není	Mezery
Reserved1 (vyhrazeno)	Není	Mezery
ReplyToFormát (název zprávy odpovědi ve formátuMQ)	MQFMT_NONE	Mezery
RemoteSysvzdálených systémů (ID vzdáleného systému CICS , které se má použít)	Není	Mezery

Tabulka 477. Pole v MQCIH pro MQCIH (pokračování)

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
RemoteTransID (CICS RTRANSID, který se má použít)	Není	Mezery
TransactionId (transakce, která se má připojit)	Není	Mezery
FacilityLike (atributy emulované terminálem)	Není	Mezery
AttentionId (klíč AID)	Není	Mezery
StartCode (kód spuštění transakce)	MQCSC_NONE	Mezery
CancelCode (kód nestandardního ukončení transakce)	Není	Mezery
NextTransactionID (další transakce, která se má připojit)	Není	Mezery
Reserved2 (vyhrazeno)	Není	Mezery
Reserved3 (vyhrazeno)	Není	Mezery
Poznámka: Zbývající pole nejsou přítomna, pokud je *Version* menší než MQCIH_VERSION_2.		
CursorPosition (pozice kurzoru)	Není	0
ErrorOffset (posun chyby ve zprávě)	Není	0
InputItem (položka vstupu)	Není	0
Reserved4 (vyhrazeno)	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQCIH_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCIH MyCIH = {MQCIH_DEFAULT};
```

Deklarace jazyka

Deklarace jazyka C pro MQCIH

```
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 */
};
```

Deklarace jazyka COBOL pro MQCIH

```
**   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.
```

Deklarace PL/I pro MQCIH

```
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 */
```

Deklarace High Level Assembler pro MQCIH

```
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)
```

Vizuální základní deklarace pro MQCIH

```
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
```

StrucId (MQCHAR4) pro MQCIH

Jedná se o identifikátor struktury struktury záhlaví informací CICS . Vždy se jedná o vstupní pole. Jeho hodnota je MQCIH_STRUC_ID.

Hodnota musí být:

MQCIH_STRUC_ID
Identifikátor pro strukturu záhlaví informací CICS .

Pro programovací jazyk C je definována také konstanta MQCIH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQCIH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQCIH

Toto pole je pole požadavku. Jeho počáteční hodnota je MQCIH_VERSION_2.

Hodnota musí být jedna z následujících:

MQCIH_VERSION_1
Version-1 CICS struktura záhlaví informací.

MQCIH_VERSION_2
Version-2 CICS struktura záhlaví informací.

Pole, která existují pouze v novější verzi struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MQCIH_CURRENT_VERSION
Aktuální verze struktury záhlaví informací CICS .

StrucLength (MQLONG) pro MQCIH

Toto pole je pole požadavku s počáteční hodnotou MQCIH_LENGTH_2.

Hodnota musí být jedna z následujících:

MQCIH_LENGTH_1
Délka struktury záhlaví informací version-1 CICS .

MQCIH_LENGTH_2
Délka struktury záhlaví informací version-2 CICS .

Následující konstanta určuje délku aktuální verze:

MQCIH_CURRENT_LENGTH
Délka aktuální verze struktury záhlaví informací CICS .

Kódování (MQLONG) pro MQCIH

Toto pole je vyhrazené pole; jeho hodnota není významná. Jeho počáteční hodnota je 0.

Kódování pro podporované struktury, které následují za strukturou MQCIH, je stejné jako kódování samotné struktury MQCIH a je převzato z předchozího záhlaví IBM MQ .

CodedCharSetId (MQLONG) pro MQCIH

CodedCharSetId je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

ID znakové sady pro podporované struktury, které následují za strukturou MQCIH, je stejné jako ID znakové sady samotné struktury MQCIH a je převzato z předchozího záhlaví IBM MQ .

Formát (MQCHAR8) pro MQCIH

Toto pole zobrazuje název formátu IBM MQ dat, která následují za strukturou MQCIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro kódování pole *Format* v deskriptoru MQMD.

Tento název formátu se také používá pro zprávu odpovědi, pokud má pole *ReplyToFormat* hodnotu MQFMT_NONE.

- Pro požadavky DPL musí být *Format* název formátu COMMAREA.
- Pro požadavky 3270 *Format* musí být CSQCBDCIa most nastaví formát CSQCBDC0 pro zprávy odpovědi.

Uživatelské procedury pro převod dat pro tyto formáty musí být nainstalovány ve správci front, kde mají být spuštěny.

Pokud zpráva požadavku vygeneruje chybovou zprávu odpovědi, má zpráva odpovědi na chybu název formátu MQFMT_STRING.

Toto pole je pole požadavku. Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQCIH

Toto pole je pole požadavku. Počáteční hodnota tohoto pole je MQCIH_NONE.

Hodnota musí být:

MQCIH_NONE
Žádné příznaky.

MQCIH_PASS_EXPIRATION
Zpráva odpovědi obsahuje:

- Stejné volby sestavy vypršení platnosti jako zpráva požadavku.
- Zbývající čas vypršení platnosti ze zprávy požadavku bez úpravy pro dobu zpracování mostu.

Pokud tuto hodnotu vynecháte, doba vypršení platnosti se nastaví na *neomezeno*.

MQCIH_REPLY_WITHOUT_NULLS
Délka zprávy odpovědi požadavku programu CICS DPL je upravena tak, aby vyloučila koncové hodnoty null (X'00 ') na konci COMMAREA vrácené programem DPL. Není-li tato hodnota nastavena, mohou být hodnoty Null významné a je vrácena úplná hodnota COMMAREA.

MQCIH_SYNC_ON_RETURN
Odkaz CICS pro požadavky DPL používá volbu SYNCONRETURN, což způsobí, že produkt CICS provede synchronizační bod po dokončení programu, pokud je dodán do jiné oblasti CICS . Most neurčuje, do které oblasti CICS se má odeslat požadavek; to je řízeno definicí programu CICS nebo prostředky pro vyrovnávání pracovní zátěže.

ReturnCode (MQLONG) pro MQCIH

Hodnota tohoto pole je návratový kód ze souboru CICS bridge , který popisuje výsledek zpracování provedeného mostem. Toto pole je pole odezvy s počáteční hodnotou MQCRC_OK.

Pole *Function*, *CompCode*, *Reason*a *AbendCode* mohou obsahovat další informace (viz Tabulka 476 na stránce 301). Hodnota je jedna z následujících:

MQCRC_APPLICATION_ABEND
(5, X'005 ') Aplikace skončila abnormálně.

MQCRC_BRIDGE_ABEND
(4, X'004 ') CICS bridge skončilo abnormálně.

MQCRC_BRIDGE_ERROR
(3, X'003 ') CICS bridge zjistil chybu.

MQCRC_BRIDGE_TIMEOUT
(8, X'008 ') Druhá nebo pozdější zpráva v rámci aktuální pracovní jednotky nebyla přijata v uvedeném čase.

MQCRC_CICS_EXEC_ERROR
(1, X'001 ') Příkaz EXEC CICS zjistil chybu.

MQCRC_MQ_API_ERROR
(2, X'002 ') Volání MQ zjistilo chybu.

MQCRC_OK
(0, X'000 ') Žádná chyba.

MQCRC_PROGRAM_NENÍ k dispozici
(7, X'007 ') Program není k dispozici.

MQCRC_SECURITY_ERROR
(6, X'006 ') Došlo k chybě zabezpečení.

MQCRC_TRANSID_NOT_AVAILABLE
(9, X'009 ') Transakce není k dispozici.

CompCode (MQLONG) pro MQCIH
Toto pole je pole odpovědi. Jeho počáteční hodnota je MQCC_OK

Hodnota vrácená v tomto poli závisí na *ReturnCode* ; viz Tabulka 476 na stránce 301.

Příčina (MQLONG) pro MQCIH
Toto pole je pole odpovědi. Jeho počáteční hodnota je MQRC_NONE.

Hodnota vrácená v tomto poli závisí na *ReturnCode* ; viz Tabulka 476 na stránce 301.

UOWControl (MQLONG) pro MQCIH
Toto pole je pole požadavku, které řídí zpracování jednotky práce prováděné produktem CICS bridge. Počáteční hodnota tohoto pole je MQCUOWC_ONLY.

Můžete požádat most o spuštění jedné transakce nebo jednoho či více programů v rámci pracovní jednotky. Pole označuje, zda CICS bridge spouští jednotku práce, provádí požadovanou funkci v rámci aktuální jednotky práce nebo ukončuje jednotku práce jejím potvrzením nebo zálohováním. Pro optimalizaci toků přenosu dat jsou podporovány různé kombinace.

Hodnota musí být jedna z následujících:

MQCUOWC_ONLY
Spusťte pracovní jednotku, proveďte funkci a poté potvrďte pracovní jednotku.

MQCUOWC_CONTINUE
Další data pro aktuální pracovní jednotku (pouze 3270).

MQCUOWC_FIRST
Spusťte jednotku práce a proveďte funkci.

MQCUOWC_MIDDLE
Provést funkci v rámci aktuální pracovní jednotky

MQCUOWC_LAST
Proveďte funkci a poté potvrďte jednotku práce.

MQCUOWC_COMMIT
Potvrdit pracovní jednotku (pouze DPL).

MQCUOWC_BACKOUT
Vycouvat jednotku práce (pouze DPL).

GetWaitInterval (MQLONG) pro MQCIH
Toto pole je pole požadavku. Jeho počáteční hodnota je MQCGWI_DEFAULT.

Toto pole se použije pouze v případě, že má parametr *UOWControl* hodnotu MQCUOWC_FIRST. Umožňuje odesílající aplikaci určit přibližnou dobu v milisekundách, po kterou budou volání MQGET vydaná mostem čekat na druhou a následnou zprávu požadavku pro pracovní jednotku spuštěnou touto zprávou. Tento prostředek potlačí výchozí interval čekání používaný mostem. Můžete použít následující speciální hodnoty:

VÝCHOZÍ-MQCGWI_DEFAULT
Výchozí interval čekání.

Tato hodnota způsobí, že agent CICS bridge bude čekat po dobu určenou při spuštění mostu.

MQWI_UNLIMITED
Neomezený interval čekání.

LinkType (MQLONG) pro MQCIH
Toto pole je pole požadavku. Jeho počáteční hodnota je MQCLT_PROGRAM.

Tato hodnota označuje typ objektu, který se most pokouší propojit. Musí to být jedna z následujících hodnot:

MQCLT_PROGRAM
Program pro DPL.

MQCLT_TRANSACTION
Transakce 3270.

OutputDataDélka (MQLONG) pro MQCIH
Toto pole je pole požadavku použité pouze pro programy DPL. Jeho počáteční hodnota je MQCODL_AS_INPUT.

Tato hodnota je délka uživatelských dat, která mají být vrácena klientovi ve zprávě odpovědi. Tato délka zahrnuje název 8bajtového programu. Délka COMMAREA předaná propojenému programu je maximum tohoto pole a délka uživatelských dat ve zprávě požadavku minus 8.

Poznámka: Délka uživatelských dat ve zprávě je délka zprávy bez struktury MQCIH.

Pokud je délka uživatelských dat ve zprávě požadavku menší než *OutputDataLength*, použije se volba DATALENGTH příkazu LINK , která umožní efektivní dodání funkce LINK do jiné oblasti CICS .

Můžete použít následující speciální hodnotu:

MQCODL_AS_INPUT
Délka výstupu je stejná jako délka vstupu.

Tato hodnota může být potřebná i v případě, že není požadována žádná odpověď, aby se zajistilo, že COMMAREA předaná propojenému programu má dostatečnou velikost.

FacilityKeepČas (MQLONG) pro MQCIH
FacilityKeepČas je doba v sekundách, po kterou je zařízení mostu uchováno po ukončení transakce uživatele.

Pro pseudokonverzační transakce zadejte hodnotu, která odpovídá očekávané době trvání pseudokonverzace; pro poslední transakci pseudokonverzace zadejte hodnotu nula a pro ostatní typy transakcí zadejte hodnotu nula.

Toto pole je pole požadavku používané pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0.

ADSDescriptor (MQLONG) pro MQCIH

Toto pole je indikátorem, který určuje, zda se mají odesílat deskriptory ADS na požadavky SEND a RECEIVE BMS.

Jsou definovány tyto hodnoty:

MQCADSD_NONE
Neodesílejte ani nepřijímejte deskriptory ADS.

MQCADSD_SEND
Odeslat deskriptory ADS.

MQCADSD_RECV
Přijmout deskriptory ADS.

MQCADSD_MSGFORMAT
Použijte formát zprávy pro deskriptory ADS.

Tento příkaz odesílá nebo přijímá deskriptory ADS s použitím dlouhé formy deskriptoru ADS. Dlouhý formulář má pole, která jsou zarovnána na 4bajtové hranice.

Nastavte pole *ADSDescriptor* následujícím způsobem:

- Pokud nepoužíváte deskriptory ADS, nastavte pole na MQCADSD_NONE.
- Používáte-li deskriptory ADS s *stejným* CCSID v každém prostředí, nastavte pole na součet hodnot MQCADSD_SEND a MQCADSD_RECV.
- Používáte-li deskriptory ADS s *různými* identifikátory CCSID v jednotlivých prostředích, nastavte pole na součet hodnot MQCADSD_SEND, MQCADSD_RECV a MQCADSD_MSGFORMAT.

Toto pole požadavku se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je MQCADSD_NONE.

ConversationalTask (MQLONG) pro MQCIH

Toto pole je indikátorem, který uvádí, zda povolit úloze vydávat požadavky na další informace, nebo zastavit úlohu a vydat nerovnou zprávu.

Hodnota musí být jedna z následujících voleb:

MQCCT_YES
Úloha je konverzační.

MQCCT_NO
Úloha není konverzační.

Toto pole je pole požadavku používané pouze pro transakce 3270. Počáteční hodnota tohoto pole je MQCCT_NO.

Stav TaskEnd(MQLONG) pro MQCIH

Toto pole je pole odezvy zobrazující stav transakce uživatele na konci úlohy. Pole se používá pouze pro transakce 3270 a jeho počáteční hodnota je MQCTES_NOSYNC.

Vrátí se jedna z následujících hodnot:

MQCTES_NOSYNC
Nesynchronizováno.

Transakce uživatele dosud nebyla dokončena a nebyla synchronizována. Pole *MsgType* v MQMD je v tomto případě MQMT_REQUEST.

MQCTES_COMMIT
Potvrdit pracovní jednotku.

Transakce uživatele ještě nebyla dokončena, ale synchronizovala první pracovní jednotku. Pole *MsgType* v MQMD je v tomto případě MQMT_DATAGRAM.

MQCTES_BACKOUT
Vycouvat z jednotky práce.

Transakce uživatele dosud nebyla dokončena. Aktuální jednotka práce je vrácena zpět. Pole *MsgType* v MQMD je v tomto případě MQMT_DATAGRAM.

MQCTES_ENDTASK
Ukončení úlohy.

Transakce uživatele byla ukončena (nebo neukončena). Pole *MsgType* v deskriptoru MQMD má v tomto případě hodnotu MQMT_REPLY.

Zařízení (MQBYTE8) pro MQCIH

V tomto poli je zobrazen 8bajtový token prostředku mostu.

Token prostředku mostu umožňuje vícenásobným transakcím v pseudokonverzaci používat stejný prostředek mostu (virtuální terminál 3270). V první nebo jediné zprávě v pseudokonverzaci nastavte hodnotu MQCFAC_NONE. Tato hodnota sděluje produktu CICS , aby pro tuto zprávu přidělil nový prostředek mostu. Token prostředku mostu je vrácen ve zprávách odezvy, je-li ve vstupní zprávě zadána nenulová hodnota *FacilityKeepTime* . Následné vstupní zprávy v rámci pseudokonverzace musí poté používat stejný token prostředku mostu.

Je definována následující speciální hodnota:

MQCFAC_NONE
Nebyl určen žádný token prostředku.

Pro programovací jazyk C je definována také konstanta MQCFAC_NONE_ARRAY a má stejnou hodnotu jako MQCFAC_NONE, ale je to pole znaků místo řetězce.

Toto pole je pole požadavku i pole odpovědi používané pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_FACILITY_LENGTH. Počáteční hodnota tohoto pole je MQCFAC_NONE.

Funkce (MQCHAR4) pro MQCIH

Toto pole je pole odpovědi. Délka tohoto pole je dána hodnotou MQ_FUNCTION_LENGTH. Počáteční hodnota tohoto pole je MQCFUNC_NONE.

Hodnota vrácená v tomto poli závisí na *ReturnCode* ; viz Tabulka 476 na stránce 301. Následující hodnoty jsou možné, když *Function* obsahuje název volání IBM MQ :

MQCFUNC_MQCONN
Volání MQCONN.

MQCFUNC_MQGET
Volání MQGET.

MQCFUNC_MQINQ
Volání MQINQ.

MQCFUNC_MQOPEN
Volání MQOPEN.

MQCFUNC_MQPUT
Volání MQPUT.

MQCFUNC_MQPUT1
Volání MQPUT1 .

MQCFUNC_NONE
Žádné volání.

Ve všech případech jsou pro programovací jazyk C definovány také konstanty MQCFUNC_ * _ARRAY; tyto konstanty mají stejné hodnoty jako odpovídající konstanty MQCFUNC_ *, ale jsou to pole znaků místo řetězců.

AbendCode (MQCHAR4) pro MQCIH

AbendCode je pole odpovědi. Délka tohoto pole je dána hodnotou MQ_ABEND_CODE_LENGTH. Počáteční hodnota tohoto pole je 4 prázdné znaky.

Hodnota vrácená v tomto poli je významná pouze v případě, že pole *ReturnCode* má hodnotu MQCRC_APPLICATION_ABEND nebo MQCRC_BRIDGE_ABEND. Pokud ano, *AbendCode* obsahuje hodnotu CICS ABCODE.

Ověřovatel (MQCHAR8) pro MQCIH
Hodnota tohoto pole je heslo nebo heslo.

Pokud je ověření identifikátoru uživatele pro CICS bridgeaktivní, *Authenticator* se použije s identifikátorem uživatele v kontextu identity MQMD k ověření odesilatele zprávy.

Toto je pole požadavku. Délka tohoto pole je dána hodnotou MQ_AUTHENTICATOR_LENGTH. Počáteční hodnota tohoto pole je 8 mezer.

Reserved1 (MQCHAR8) pro MQCIH
Toto pole je vyhrazené pole. Hodnota musí být 8 mezer.

ReplyToFormát (MQCHAR8) pro MQCIH
Hodnota tohoto pole je název formátu IBM MQ zprávy odpovědi, která je odeslána jako odpověď na aktuální zprávu.

Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro kódování pole *Format* v deskriptoru MQMD.

Toto pole je pole požadavku použité pouze pro programy DPL. Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

RemoteSysID (MQCHAR4) pro MQCIH
Toto pole zobrazuje identifikátor systému CICS systému CICS , který zpracovává požadavek.

Je-li toto pole prázdné, systémový požadavek CICS se zpracuje na stejném systému CICS jako monitor mostu. Použité ID SYSID je vráceno ve zprávě odpovědi.

V případě pseudokonverzace 3270 musí všechny následné zprávy v konverzaci určovat vzdálený identifikátor SYSID vrácený v počáteční odpovědi. Je-li uveden, musí SYSID:

- Buďte aktivní.
- Mít přístup k frontě požadavků IBM MQ .
- Jsou přístupné prostřednictvím odkazů konzoly ISC CICS ze systému CICS monitoru mostu.

RemoteTransID (MQCHAR4) pro MQCIH
Toto pole je volitelné pole Požadavek. Délka tohoto pole je dána hodnotou MQ_TRANSACTION_ID_LENGTH.

Je-li uvedeno, pole se použije jako hodnota RTRANSID CICS START.

TransactionId (MQCHAR4) pro MQCIH
Toto pole je pole požadavku. Jeho délka je dána hodnotou MQ_TRANSACTION_ID_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

Má-li parametr *LinkType* hodnotu MQCLT_TRANSACTION, *TransactionId* je identifikátor transakce uživatelské transakce, která se má spustit; v tomto případě zadejte neprázdnou hodnotu.

Má-li parametr *LinkType* hodnotu MQCLT_PROGRAM, *TransactionId* je kód transakce, pod kterým mají být spuštěny všechny programy v rámci pracovní jednotky. Zadáte-li prázdnou hodnotu, použije se výchozí kód transakce mostu DPL CICS (CKBP). Je-li hodnota neprázdná, musíte ji definovat pro CICS jako lokální transakci s počátečním programem, který je CSQCBP00. Toto pole platí pouze v případě, že má parametr *UOWControl* hodnotu MQCUOWC_FIRST nebo MQCUOWC_ONLY.

FacilityLike (MQCHAR4) pro MQCIH
FacilityLike je název instalovaného terminálu, který se má použít jako model pro zařízení mostu.

Hodnota mezer znamená, že hodnota *FacilityLike* je převzata z definice profilu transakce mostu nebo je použita výchozí hodnota.

Toto pole je pole požadavku používané pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_FACILITY_LIKE_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

AttentionId (MQCHAR4) pro MQCIH

Hodnota v tomto poli určuje počáteční hodnotu klíče AID při spuštění transakce. Jedná se o hodnotu 1 bajt, zarovnanou vlevo.

AttentionId je pole požadavku, které se používá pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_ATTENTION_ID_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

StartCode (MQCHAR4) pro MQCIH

Hodnota tohoto pole je indikátorem, který určuje, zda most emuluje transakci terminálu nebo transakci zahájenou příkazem START.

Hodnota musí být jedna z následujících:

MQCSC_START
Spustit.

MQCSC_STARTDATA
Počáteční data.

MQCSC_TERMINPUT
Vstup terminálu.

MQCSC_NONE
Není.

Ve všech případech jsou pro programovací jazyk C definovány také konstanty MQCSC_ * _ARRAY; tyto konstanty mají stejné hodnoty jako odpovídající konstanty MQCSC_ *, ale jsou to pole znaků místo řetězců.

V odezvě z mostu je toto pole nastaveno na počáteční kód odpovídající dalšímu ID transakce obsaženému v poli *NextTransactionId* . V odezvě jsou možné následující počáteční kódy:

- MQCSC_START
- MQCSC_STARTDATA
- MQCSC_TERMINPUT

Pro systém CICS Transaction Server 1.2je toto pole pouze pole požadavku; jeho hodnota v odezvě není definována.

Pro produkt CICS Transaction Server 1.3 a následná vydání je toto pole jak pole požadavku, tak pole odezvy.

Toto pole se používá pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_START_CODE_LENGTH. Počáteční hodnota tohoto pole je MQCSC_NONE.

CancelCode (MQCHAR4) pro MQCIH

Hodnota v tomto poli je nestandardní kód, který se má použít k ukončení transakce (obvykle konverzační transakce, která požaduje více dat). Jinak je toto pole nastaveno na mezery.

Toto pole je pole požadavku používané pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_CANCEL_CODE_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

NextTransactionId (MQCHAR4) pro MQCIH

Tato hodnota je název další transakce vrácené transakcí uživatele (obvykle EXEC CICS RETURN TRANSID). Pokud neexistuje žádná další transakce, toto pole se nastaví na mezery.

Toto pole je pole odpovědi používané pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_TRANSACTION_ID_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

Reserved2 (MQCHAR8) pro MQCIH
Toto pole je vyhrazené pole. Hodnota musí být 8 mezer.

Reserved3 (MQCHAR8) pro MQCIH
Toto pole je vyhrazené pole. Hodnota musí být 8 mezer.

CursorPosition (MQLONG) pro MQCIH
Hodnota v tomto poli zobrazuje počáteční pozici kurzoru při spuštění transakce. Pro konverzační transakce je pozice kurzoru ve vektoru RECEIVE.

Toto pole je pole požadavku používané pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0. Toto pole není přítomno, pokud je hodnota *Version* menší než MQCIH_VERSION_2.

ErrorOffset (MQLONG) pro MQCIH
Pole ErrorOffset zobrazuje pozici neplatných dat zjištěných uživatelskou procedurou mostu. Toto pole poskytuje posun od začátku zprávy k umístění neplatných dat.

ErrorOffset je pole odezvy používané pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0. Toto pole není přítomno, pokud je hodnota *Version* menší než MQCIH_VERSION_2.

InputItem (MQLONG) pro MQCIH
Toto pole je vyhrazené pole. Hodnota musí být 0.

Toto pole není přítomno, pokud je hodnota *Version* menší než MQCIH_VERSION_2.

Reserved4 (MQLONG) pro MQCIH
Toto pole je vyhrazené pole. Hodnota musí být 0.

Toto pole není přítomno, pokud je hodnota *Version* menší než MQCIH_VERSION_2.

MQCMHO-Volby vytvoření manipulátoru zprávy

Struktura **MQCMHO** umožňuje aplikacím určit volby, které řídí způsob vytváření manipulátorů zpráv. Struktura je vstupní parametr volání **MQCRTMH** .

Dostupnost
Struktura **MQCMHO** je k dispozici na následujících platformách:

- AIX
- IBM i
- Linux
- Windows
- z/OS

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Znaková sada a kódování

Data v souboru **MQCMHO** musí být ve znakové sadě aplikace a kódování aplikace (**MQENC_NATIVE**).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 478. Pole v MQCMHO

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQCMHO_STRUC_ID	'CMHO'
Verze (číslo verze struktury)	MQCMHO_VERSION_1	1
Volby (volby)	MQCMHO_DEFAULT_VALIDATION	0

Notes:

1. V programovacím jazyku C se jedná o proměnnou makra.MQCMHO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQCMHO MyCMHO = {MQCMHO_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQCMHO

```
struct tagMQCMHO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQCRTMH */
};
```

Deklarace jazyka COBOL pro objekt MQCMHO

```
**   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.
```

Deklarace PL/I pro MQCMHO

```
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 */
```

Deklarace High Level Assembler pro objekt MQCMHO

```
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)
```

StrucId (MQCHAR4) pro MQCMHO

Jedná se o identifikátor struktury struktury voleb popisovače vytvoření zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQCMHO_STRUC_ID.

Hodnota musí být:

MQCMHO_STRUC_ID
Identifikátor pro vytvoření struktury voleb popisovače zprávy.

Pro programovací jazyk C je definována také konstanta **MQCMHO_STRUC_ID_ARRAY** . Má stejnou hodnotu jako **MQCMHO_STRUC_ID**, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQCMHO

Toto pole je vždy vstupní pole. Jeho počáteční hodnota je MQCMHO_VERSION_1.

Toto je číslo verze struktury; hodnota musí být:

MQCMHO_VERSION_1
Version-1 vytvořte strukturu voleb popisovače zprávy.

Následující konstanta určuje číslo verze aktuální verze:

MQCMHO_CURRENT_VERSION
Aktuální verze struktury voleb popisovače vytvoření zprávy.

Volby (MQLONG) pro MQCMHO

Toto pole je vždy vstupní pole. Jeho počáteční hodnota je MQCMHO_DEFAULT_VALIDATION.

Lze zadat jednu z následujících voleb:

MQCMHO_VALIDATE

Při volání **MQSETMP** pro nastavení vlastnosti v tomto popisovači zprávy je název vlastnosti ověřen, aby se zajistilo, že:

- neobsahuje žádné neplatné znaky.
- nezačíná na JMS nebo usr.JMS s výjimkou následujících:
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType.
  - JMSXGroupID
  - JMSXGroupSeq

  Tyto názvy jsou vyhrazeny pro vlastnosti JMS .
- není jedním z následujících klíčových slov, v žádné směsi velkých nebo malých písmen:
  - AND
  - BETWEEN
  - Esc
  - NEPRAVDA
  - IN
  - IS

  - Jako
  - NE
  - NEDEFINOVÁNO
  - NEBO
  - PRAVDA
- nezačíná tělo. nebo Root. (s výjimkou Root.MQMD.).

Je-li vlastnost MQ-defined (mq. *) a název je rozpoznán, pole deskriptoru vlastností jsou nastavena na správné hodnoty pro vlastnost. Není-li vlastnost rozpoznána, pole *Support* deskriptoru vlastnosti je nastaveno na hodnotu **MQPD_OPTIONAL**.

MQCMHO_DEFAULT_VALIDATION

Tato hodnota určuje, že dojde k výchozí úrovni ověřování názvů vlastností.

Výchozí úroveň ověření je ekvivalentní úrovni určené parametrem **MQCMHO_VALIDATE**.

Tato hodnota je výchozí hodnota.

MQCMHO_NO_VALIDATION

Nedojde k žádnému ověření názvu vlastnosti. Viz popis položky **MQCMHO_VALIDATE**.

Výchozí volba: Pokud není vyžadována žádná z výše uvedených voleb, lze použít následující volbu:

MQCMHO_NONE

Všechny volby předpokládají své výchozí hodnoty. Pomocí této hodnoty označíte, že nebyly zadány žádné další volby. Produkt **MQCMHO_NONE** pomáhá s dokumentací k programům; není určeno, aby tato volba byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

MQCNO-Volby připojení

Struktura MQCNO umožňuje aplikaci určit volby související s připojením ke správci front. Struktura je vstupní/výstupní parametr volání MQCONNX.

Další informace o použití sdílených manipulátorů a volání MQCONNX naleznete v tématu Sdílená připojení (nezávislá na podprocesu) s produktem MQCONNX.

Dostupnost

Struktura MQCNO je k dispozici na následujících platformách:

- AIX
- IBM i
- Linux
- Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Verze

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi MQCNO, ale s počáteční hodnotou pole *Version* nastavenou na MQCNO_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole *Version* na požadované číslo verze.

Znaková sada a kódování

Data v MQCNO musí být ve znakové sadě zadané atributem správce front **CodedCharSetId** a v kódování lokálního správce front poskytnutém proměnnou MQENC_NATIVE. Pokud je však aplikace spuštěna jako IBM MQ MQI client, musí být struktura ve znakové sadě a kódování klienta.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 479. Pole v MQCNO pro MQCNO

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQCNO_STRUC_ID	'CNO¬'
Verze (číslo verze struktury)	MQCNO_VERSION_1	1
Volby (volby, které řídí akci MQCONNX)	MQCNO_NONE	0
Poznámka: Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než MQCNO_VERSION_2.		
ClientConnOffset (Offset struktury MQCD pro připojení klienta)	Není	0
ClientConnPtr (adresa struktury MQCD pro připojení klienta)	Není	Ukazatel Null nebo bajty s hodnotou Null
Poznámka: Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než hodnota MQCNO_VERSION_3.		
ConnTag (značka připojení správce front)	MQCT_NONE	Hodnoty null
Poznámka: Zbývající pole jsou ignorována, pokud je *Version* menší než MQCNO_VERSION_4.		
SSLConfigPtr (adresa struktury MQSCO pro připojení klienta)	Není	Ukazatel Null nebo bajty s hodnotou Null
SSLConfigOffset (posun struktury MQSCO pro připojení klienta)	Není	0
Poznámka: Zbývající pole jsou ignorována, pokud je *Version* menší než MQCNO_VERSION_5.		
ConnectionId (jedinečné ID připojení)	Není	Ukazatel Null nebo bajty s hodnotou Null
SecurityParmsOffset (posun struktury MQSCO pro parametry zabezpečení)	Není	Ukazatel Null nebo bajty s hodnotou Null
SecurityParmsPtr (adresa struktury MQSCO pro parametry zabezpečení)	Není	Ukazatel Null nebo bajty s hodnotou Null
Poznámka: Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než hodnota MQCNO_VERSION_6.		
Vyhrazeno (vyhrazené pole)	Není	Vyhrazené pole pro vyplnění struktury na 64bitovou hranici.

Tabulka 479. Pole v MQCNO pro MQCNO (pokračování)

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
CCDTUrlLength (délka adresy URL URL)	Není	Délka řetězce identifikovaného pomocí *CCDTUrlPtr* nebo *CCDTUrlOffset*
CCDTUrlPtr (ukazatel na adresu URL URL)	Není	Ukazatel na řetězec, který obsahuje URL, který slouží k identifikaci umístění tabulky kanálů připojení klienta, jež má být použita pro připojení.
CCDTUrlOffset (posunutí adresy URL URL)	Není	Posun v bajtech od řetězce, který obsahuje URL identifikující umístění tabulky kanálu připojení klienta, která se má použít pro připojení.
Poznámka: Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než MQCNO_VERSION_7.		
ApplName (název nastavený aplikací)	Není	Název nastavený aplikací pro identifikaci připojení ke správci front
Reserved2 (vyhrazené pole)	Není	Vyhrazené pole pro vyplnění struktury na 64bitovou hranici.
V 9.3.0 V 9.3.0 **Poznámka:** Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než hodnota MQCNO_VERSION_8.		
V 9.3.0 BalanceParms| Offset	Není	Posun v bajtech ke struktuře MQBNO
V 9.3.0 BalanceParmsPtr	Není	Ukazatel na umístění struktury MQBNO

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQCNO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCNO MyCNO = {MQCNO_DEFAULT};
```

Deklarace jazyka

Poznámka: V 9.3.0 V každé z následujících deklarací byly v adresáři IBM MQ 9.2.4 přidány poslední dva řádky (`Offset of the MQBMO structure` a `Address of the location of the MQBMO structure`) pro uživatele CD a v adresáři IBM MQ 9.3.0 pro uživatele LTS.

C prohlášení pro MQCNO

```
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 managerconnection tag */
  PMQSCO     SSLConfigPtr;      /* Address of MQSCO structure for client
                                   connection */
  MQLONG     SSLConfigOffset;   /* Offset of MQSCO structure for client
                                   connection */
  MQBYTE24   ConnectionId;      /* Unique connection identifier */
  MQLONG     SecurityParmsOffset /* Security fields */
  PMQCSP     SecurityParmsPtr   /* Security parameters */
  MQLONG     CCDTUrlLength      /* Length of string identified by Ptr or offset */
  MQLONG     CCDTUrlOffset      /* Offset in bytes to URL of client connection channel */
  PMQURL     CCDTUrlPtr         /* Address of string containing URL */
  MQBYTE4    Reserved           /* Reserved field to pad out to 64 bit boundary */
  MQCHAR28   ApplName           /* Name set by the application to identify the connection to
                                   the queue manager */
  MQBYTE4    Reserved2          /* Reserved field to pad out to 64 bit boundary */
  MQLONG     BalanceParmsOffset /* Offset of the MQBMO structure */
  PMQBMO     BalanceParmsPtr    /* Address of the location of the MQBMO structure */
};
```

Deklarace jazyka COBOL pro MQCNO

```
**   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.
**    Length of string identified by CCDTURLOFFSET or CCDTURLPTR
   15 MQCNO-CCDTURLLENGTH
**    Pointer to a string which contains a URL, to identify the location of the client
connection channel
   15 MQCNO-CCDTURLPTR
**    Address of string which contains a URL that identifies the location of the client
connection channel table
   15 MQCNO-CCDTURLOFFSET
**    Reserved field to pad to 64 bit boundary
   15 MQCNO-RESERVED
**    Name set by the application to identify the connection to the queue manager
   15 MQCNO-APPLNAME
**    Reserved field to pad to 64 bit boundary
```

```
   15 MQCNO-RESERVED2
**    Address of the MQBMO structure
   15 MQCNO-BALANCEPARMSOFFSET
**    Pointer to the MQBMO structure
   15 MQCNO-BALANCEPARMSPTR
```

Prohlášení PL/I pro MQCNO

```
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 managerconnection tag */
  3 SSLConfigPtr       pointer,        /* Address of MQSCO structure for
                                          client connection */
  3 SSLConfigOffset    fixed bin(31),  /* Offset of MQSCO structure for
                                          client connection */
  3 ConnectionId       char(24),       /* Unique connection identifier
  3 SecurityParmsOffset fixed bin(31)  /* Offset of MQCSP structure for
                                          security parameters */
  3 SecurityParmsPtr   pointer,        /* Address of MQCSP structure for
                                          security parameters */
  3 CCDTUrlLength      fixed bin(31)   /* Length of string identified by CCDTUrlPtr
                                          or CCDTUrlOffset */
  3 CCDTUrlOffset      fixed bin(31)   /* Offset in bytes to URL of client connection channel */
  3 CCDTUrlPtr         pointer         /* Pointer to string containing URL */
  3 Reserved           char(4)         /* Reserved field to pad out to 64 bit boundary */
  3 ApplName           char(28)        /* Name set by the application to identify the
connection to
                                          the queue manager */
  3 Reserved2          char(4)         /* Reserved field to pad out to 64 bit boundary */
  3 BalanceParmsOffset fixed bin(31)   /* Offset of the MQBMO structure */
  3 BalanceParmsPtr    pointer         /* Address of the MQBMO structure */
```

Deklarace High Level Assembler pro MQCNO

```
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)
MQCNO_CCDTURLLENGTH      DS   F      Length of string identified by CCDTURLPTR or
*                                     CCDTURLOFFSET
MQCNO_CCDTURLOFFSET      DS   F      Offset in bytes to URL of client connection channel
MQCNO_CCDTURLPTR         DS   F      Pointer to string containing URL
RESERVED                 DS   XL4    Reserved field to pad out to 64 bit boundary
APPLNAME                 DS   CL28   Name set by the application to identify the connection to
*                                     the queue manager
RESERVED2                DS   XL4    Reserved field to pad out to 64 bit boundary
MQCNO_BALANCEPARMSOFFSET DS   F      Offset of the MQBMO structure
MQCNO_BALANCEPARMSPTR    DS   F      Address of the MQBMO structure
```

Vizuální základní deklarace pro MQCNO

```
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'
  CCDTUrlLength       As Long      'Length of string identified by CCDTUrlPtr'
                                   'or CCDTUrlOffset'
  CCDTUrlOffset       As Long      'Offset in bytes to URL of client connection channel'
  CCDTUrlPtr          As MQPTR     'Pointer to string containing URL'
  Reserved            As MQBYTE4   'Reserved field to pad out to 64 bit boundary'
  ApplName            As String*28 'Name set by the application to identify the connection to'
                                   'the queue manager'
  Reserved2           As MQBYTE4   'Reserved field to pad out to 64 bit boundary'
  BalanceParmsOffset  As Long      'Offset in bytes to MQBNO structure'
  BalanceParmsPtr     As MQPTR     'Address of MQBNO structure'
  End Type
```

Související úlohy

Použití MQCONNX

StrucId (MQCHAR4) pro MQCNO

Jedná se o identifikátor struktury struktury voleb připojení. Vždy se jedná o vstupní pole. Jeho hodnota je MQCNO_STRUC_ID.

Hodnota musí být:

MQCNO_STRUC_ID
Identifikátor struktury voleb připojení.

Pro programovací jazyk C je definována také konstanta MQCNO_STRUC_ID_ARRAY. Tato konstanta má stejnou hodnotu jako MQCNO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQCNO

Verze je vždy vstupní pole. Jeho počáteční hodnota je MQCNO_VERSION_1.

Hodnota musí být jedna z následujících:

MQCNO_VERSION_1
Version-1 struktura voleb připojení.

MQCNO_VERSION_2
Version-2 struktura voleb připojení.

MQCNO_VERSION_3
Struktura voleb připojení Version-3 .

MQCNO_VERSION_4
Struktura voleb připojení Version-4 .

MQCNO_VERSION_5
Version-5 struktura voleb připojení.

MQCNO_VERSION_6
Struktura voleb připojení Version-6 .

MQCNO_VERSION_7
Version-7 struktura voleb připojení.

V 9.3.0 **MQCNO_VERSION_8**
Version-8 struktura voleb připojení.

Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MQCNO_CURRENT_VERSION
Aktuální verze struktury voleb připojení.

Volby (MQLONG) pro MQCNO

Volby, které řídí akci MQCONNX.

Volby účtování

Následující volby řídí typ evidence, pokud je atribut správce front **AccountingConnOverride** nastaven na hodnotu MQMON_ENABLED:

MQCNO_ACCOUNTING_MQI_ENABLED
Je-li v definici správce front zakázáno monitorování shromažďování dat nastavením atributu **MQIAccounting** na hodnotu MQMON_OFF, nastavení tohoto příznaku povolí shromažďování dat evidence MQI.

MQCNO_ACCOUNTING_MQI_DISABLED
Je-li v definici správce front zakázáno shromažďování dat monitorování nastavením atributu **MQIAccounting** na hodnotu MQMON_OFF, nastavení tohoto příznaku zastaví shromažďování dat evidence MQI.

MQCNO_ACCOUNTING_Q_ENABLED
Je-li v definici správce front zakázáno shromažďování dat evidence front nastavením atributu **MQIAccounting** na hodnotu MQMON_OFF, nastavení tohoto příznaku povolí shromažďování dat evidence pro fronty, které určují správce front v poli *MQIAccounting* příslušné definice fronty.

MQCNO_ACCOUNTING_Q_DISABLED
Je-li v definici správce front zakázáno shromažďování dat evidence front nastavením atributu **MQIAccounting** na hodnotu MQMON_OFF, nastavení tohoto příznaku vypne shromažďování dat evidence pro fronty, které určují správce front v poli *MQIAccounting* příslušné definice fronty.

Pokud není definován žádný z těchto příznaků, evidence pro připojení je definována v atributech správce front.

Volby vazeb

Následující volby řídí typ vazby IBM MQ , která se má použít. Zadejte pouze jednu z těchto voleb:

MQCNO_STANDARDNÍ_VAZBA
Aplikace a lokální agent správce front (komponenta, která spravuje operace řazení do fronty) jsou spouštěny v oddělených jednotkách provedení (obvykle v oddělených procesech). Toto uspořádání udržuje integritu správce front; to znamená, že chrání správce front před chybným programem.

Pokud správce front podporuje více typů vazeb a nastavíte volbu MQCNO_STANDARD_BINDING, správce front použije k výběru skutečného typu vazby atribut **DefaultBindType** v sekci `Connection` v souboru `qm.ini` . Není-li tato sekce definována nebo nelze-li hodnotu použít nebo není-li pro aplikaci vhodná, správce front vybere příslušný typ vazby. Správce front nastaví skutečný typ vazby použitý ve volbách připojení.

Volbu MQCNO_STANDARD_BINDING použijte v situacích, kdy aplikace možná nebyla plně otestována nebo byla nespolehlivá či nedůvěryhodná. MQCNO_STANDARD_BINDING je výchozí.

Tato volba je podporována ve všech prostředích.

Pokud odkazujete na knihovnu `mqm` , bude nejprve proveden pokus o standardní připojení k serveru s použitím výchozího typu vazby. Pokud se nepodařilo načíst základní knihovnu serveru, dojde k pokusu o připojení klienta.

- Chcete-li změnit chování MQCONN (nebo MQCONNX, je-li zadána volba MQCNO_STANDARD_BINDING), nastavte proměnnou prostředí MQ_CONNECT_TYPE na jednu

z následujících voleb. Všimněte si, že existuje výjimka v tomto: Pokud je parametr MQCNO_FASTPATH_BINDING zadán s parametrem MQ_CONNECT_TYPE nastaveným na hodnotu LOCAL nebo STANDARD, může administrátor snížit úroveň připojení rychlé cesty bez související změny aplikace.

Tabulka 480. Hodnoty pro MQ_CONNECT_TYPE, které mění chování MQCONN nebo MQCONNX

Hodnota	**Význam**
CLIENT	Byl učiněn pokus pouze o připojení klienta.
Rychlý	Tato hodnota byla podporována v předchozích verzích, ale nyní je ignorována, pokud je zadána.
LOKÁLNÍ	Byl učiněn pokus pouze o připojení k serveru. Zkrácené cesty připojení jsou sníženy na standardní připojení serveru.
STANDARD	Podporováno pro kompatibilitu s předchozími verzemi. Tato hodnota je nyní považována za hodnotu LOCAL.

- Není-li proměnná prostředí MQ_CONNECT_TYPE nastavena při volání MQCONNX, dojde k pokusu o standardní připojení k serveru s použitím výchozího typu vazby. Pokud se načtení knihovny serveru nezdaří, dojde k pokusu o připojení klienta.

MQCNO_FASTPATH_BINDING

Aplikace a lokální agent správce front jsou součástí stejné jednotky provedení. To je v rozporu s typickou metodou vazby, kde aplikace a lokální agent správce front běží v oddělených jednotkách provedení.

MQCNO_FASTPATH_BINDING je ignorována, pokud správce front nepodporuje tento typ vazby; zpracování pokračuje, jako by volba nebyla určena.

MQCNO_FASTPATH_BINDING může být výhodné v situacích, kdy více procesů spotřebuje více prostředků než celkový prostředek používaný aplikací.Aplikace, která používá vazbu zkrácené cesty, se nazývá *důvěryhodná aplikace*.

Při rozhodování o tom, zda použít vazbu rychlé cesty, zvažte následující důležité body:

- Použití volby MQCNO_FASTPATH_BINDING nebrání aplikaci změnit nebo porušit zprávy a další datové oblasti patřící správci front. Tuto volbu použijte pouze v situacích, kdy jste tyto problémy plně vyhodnotili.
- Aplikace nesmí používat asynchronní signály nebo přerušení časovače (například `sigkill`) s volbou MQCNO_FASTPATH_BINDING. Existují také omezení použití segmentů sdílené paměti.
- K odpojení od správce front musí aplikace použít volání MQDISC.
- Aplikace musí být dokončena před ukončením správce front pomocí příkazu `endmqm` .
- IBM i V systému IBM imusí být úloha spuštěna pod profilem uživatele, který patří do skupiny `QMQMADM` . Program se také nesmí zastavit abnormálně, jinak se mohou vyskytnout nepředvídatelné výsledky.
- Linux AIX V systému AIX and Linuxmusí být identifikátor uživatele `mqm` efektivním identifikátorem uživatele a identifikátor skupiny `mqm` musí být efektivním identifikátorem skupiny. Chcete-li aplikaci spustit tímto způsobem, nakonfigurujte program tak, aby byl vlastněn identifikátorem uživatele `mqm` a identifikátorem skupiny `mqm` , a poté nastavte v programu bity oprávnění `setuid` a `setgid` .

  Produkt IBM MQ Object Authority Manager (OAM) stále používá skutečné ID uživatele pro kontrolu oprávnění.
- Windows V systému Windowsmusí být program členem skupiny `mqm` . Vazba rychlé cesty není podporována pro 64bitové aplikace.

Volba MQCNO_FASTPATH_BINDING je podporována v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

z/OS V systému z/OSje volba přijata, ale ignorována.

Další informace o důsledcích používání důvěryhodných aplikací naleznete v tématu Omezení pro důvěryhodné aplikace.

MQCNO_SHARED_BINDING

Pomocí příkazu MQCNO_SHARED_BINDING sdílí aplikace a lokální agent správce front některé prostředky. MQCNO_SHARED_BINDING se ignoruje, pokud správce front nepodporuje tento typ vazby. Zpracování pokračuje, jako by volba nebyla uvedena.

MQCNO_ISOLATED_BINDING

V tomto případě jsou proces aplikace a lokální agent správce front vzájemně izolováni v tom, že nesdílejí prostředky. MQCNO_ISOLATED_BINDING se ignoruje, pokud správce front nepodporuje tento typ vazby. Zpracování pokračuje, jako by volba nebyla uvedena.

MQCNO_CLIENT_BINDING

Tuto volbu uveďte, chcete-li provést pokus aplikace pouze o připojení klienta. Tato volba má následující omezení:

- z/OS MQCNO_CLIENT_BINDING je v systému z/OSignorováno.
- Volba MQCNO_CLIENT_BINDING je odmítnuta s volbou MQRC_OPTIONS_ERROR, pokud je určena s jinou volbou vazby MQCNO než MQCNO_STANDARD_BINDING.
- MQCNO_CLIENT_BINDING není k dispozici pro Java nebo .NET , protože mají vlastní mechanizmy pro výběr typu vazby.

MQCNO_LOCAL_BINDING

Tuto volbu uveďte, chcete-li provést pokus aplikace o připojení k serveru. Je-li zadána také volba MQCNO_FASTPATH_BINDING, MQCNO_ISOLATED_BINDING nebo MQCNO_SHARED_BINDING, je připojení tohoto typu namísto toho dokumentováno v této sekci. Jinak dojde k pokusu o standardní připojení k serveru s použitím výchozího typu vazby. MQCNO_LOCAL_BINDING má následující omezení:

- z/OS MQCNO_LOCAL_BINDING je v systému z/OSignorováno.
- Volba MQCNO_LOCAL_BINDING je odmítnuta s volbou MQRC_OPTIONS_ERROR, pokud je zadána s jinou volbou opětovného připojení MQCNO než MQCNO_RECONNECT_AS_DEF.
- MQCNO_LOCAL_BINDING není k dispozici pro Java nebo .NET , protože mají své vlastní mechanizmy pro výběr typu vazby.

Na následujících platformách můžete k řízení použitého typu vazby použít proměnnou prostředí `MQ_CONNECT_TYPE` s typem vazby určeným v poli `Options` .

- AIX
- Linux
- Windows

Zadáte-li tuto proměnnou prostředí, musí mít hodnotu FASTPATH nebo STANDARD ; pokud má jinou hodnotu, je ignorována. Hodnota proměnné prostředí rozlišuje velikost písmen; další informace viz Proměnná prostředí MQCONNX .

Proměnná prostředí a pole *Options* interagují takto:

- Pokud vynecháte proměnnou prostředí nebo jí dáte hodnotu, která není podporována, použití vazby rychlé cesty je určeno pouze polem Options .
- Zadáte-li pro proměnnou prostředí podporovanou hodnotu, použije se vazba zkrácené cesty pouze v případě, že je v proměnné prostředí i v poli Options určena vazba zkrácené cesty.

Volby značky připojení

Tyto volby jsou podporovány pouze při připojování ke správci front z/OS a řídí použití značky připojení ConnTag. Můžete uvést pouze jednu z těchto voleb.

Přesná implementace značek připojení se liší mezi IBM MQ for z/OS a IBM MQ for Multiplatforms:

- z/OS Následující volby, kromě volby *MQCNO_GENERATE_CONN_TAG*, jsou podporovány pouze při připojování ke správci front z/OS a řídí použití značky připojení. Můžete zadat pouze jednu z podporovaných voleb.
- ALW *MQCNO_GENERATE_CONN_TAG* je podporován pouze na jiných platformách než z/OS.

ALW **MQCNO_GENERATE_CONN_TAG**

Vrací značku připojení, kterou správce front přidružil k tomuto připojení, ve výstupní struktuře MQCNO.

Vrácená značka připojení bude identická pro všechna připojení, která správce front považuje za jedinou instanci aplikace.

z/OS **MQCNO_SERIALIZE_CONN_TAG_Q_MGR**

Tato volba vyžaduje výhradní použití značky připojení v lokálním správci front. Pokud je značka připojení již používána v lokálním správci front, volání MQCONNX se nezdaří s kódem příčiny MQRC_CONN_TAG_IN_USE. Výsledek volání není ovlivněn použitím značky připojení jinde ve skupině sdílení front, do které patří lokální správce front.

z/OS **MQCNO_SERIALIZE_CONN_TAG_QSG**

Tato volba vyžaduje výhradní použití značky připojení v rámci skupiny sdílení front, do které patří lokální správce front. Pokud se značka připojení již používá ve skupině sdílení front, volání MQCONNX se nezdaří s kódem příčiny MQRC_CONN_TAG_IN_USE.

z/OS **MQCNO_RESTRICT_CONN_TAG_Q_MGR**

Tato volba vyžaduje sdílené použití značky připojení v lokálním správci front. Pokud je značka připojení již používána v lokálním správci front, může být volání MQCONNX úspěšné, pokud je požadující aplikace spuštěna ve stejném rozsahu zpracování jako existující uživatel značky. Není-li tato podmínka splněna, volání MQCONNX selže s kódem příčiny MQRC_CONN_TAG_IN_USE. Výsledek volání není ovlivněn použitím značky připojení jinde ve skupině sdílení front, do které patří lokální správce front.

- Aplikace musí být spuštěny ve stejném adresním prostoru MVS, aby bylo možné sdílet značku připojení. Pokud je aplikace používající značku připojení klientskou aplikací, není MQCNO_RESTRICT_CONN_TAG_Q_MGR povolena.

z/OS **MQCNO_RESTRICT_CONN_TAG_QSG**

Tato volba vyžaduje sdílené použití značky připojení v rámci skupiny sdílení front, do které patří lokální správce front. Pokud je značka připojení již používána ve skupině sdílení front, může být volání MQCONNX úspěšné za předpokladu, že požadující aplikace je spuštěna ve stejném rozsahu zpracování a je připojena ke stejnému správci front jako existující uživatel značky.

Nejsou-li tyto podmínky splněny, volání MQCONNX selže s kódem příčiny MQRC_CONN_TAG_IN_USE.

- Aplikace musí být spuštěny ve stejném adresním prostoru MVS, aby bylo možné sdílet značku připojení. Pokud je aplikace používající značku připojení klientskou aplikací, není MQCNO_RESTRICT_CONN_TAG_QSG povolena.

Pokud není uvedena žádná z těchto voleb, `ConnTag` se nepoužije. Tyto volby nejsou platné, pokud je hodnota `Version` menší než MQCNO_VERSION_3.

Volby sdílení popisovače

Tyto volby jsou podporovány v následujících prostředích:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

Řídí sdílení manipulátorů mezi různými podprocesy (jednotkami paralelního zpracování) v rámci stejného procesu. Můžete uvést pouze jednu z těchto voleb:

MQCNO_HANDLE_SHARE_NONE

Tato volba označuje, že manipulátory připojení a objektu mohou být použity pouze podprocesem, který způsobil přidělení manipulátoru (tj. podprocesem, který vydal volání MQCONN, MQCONNX nebo MQOPEN). Manipulátory nemohou být použity jinými podprocesy náležejícími ke stejnému procesu.

MQCNO_HANDLE_SHARE_BLOCK

Tato volba označuje, že obslužné rutiny připojení a objektu přidělené jedním podprocesem procesu mohou být použity jinými podprocesy náležejícími ke stejnému procesu. Avšak pouze jeden podproces v daném okamžiku může použít jakýkoli konkrétní manipulátor; to znamená, že je povoleno pouze sériové použití manipulátoru. Pokud se podproces pokusí použít manipulátor, který je již používán jiným podprocesem, bude volání blokovat (čekat), dokud nebude manipulátor k dispozici.

MQCNO_HANDLE_SHARE_NO_BLOCK

Toto je stejné jako MQCNO_HANDLE_SHARE_BLOCK, kromě toho, že pokud manipulátor používá jiný podproces, volání se okamžitě dokončí s MQCC_FAILED a MQRC_CALL_IN_PROGRESS namísto blokování, dokud nebude manipulátor k dispozici.

Podproces může mít nula nebo jeden nesdílený popisovače:

- Každé volání MQCONN nebo MQCONNX, které určuje MQCNO_HANDLE_SHARE_NONE, vrací nový nesdílený manipulátor pro první volání a stejný nesdílený manipulátor pro druhé a pozdější volání (za předpokladu, že neprobíhá volání MQDISC). Kód příčiny je MQRC_ALREADY_CONNECTED pro druhé a pozdější volání.
- Každé volání MQCONNX, které určuje MQCNO_HANDLE_SHARE_BLOCK nebo MQCNO_HANDLE_SHARE_NO_BLOCK, vrací při každém volání nový sdílený popisovač.

Manipulátory objektů dědí stejné vlastnosti sdílení jako manipulátor připojení určený ve volání MQOPEN, které vytvořilo manipulátor objektu. Jednotky práce také dědí stejné vlastnosti sdílení jako manipulátor připojení použitý ke spuštění jednotky práce; pokud je jednotka práce spuštěna v jednom podprocesu pomocí sdíleného manipulátoru, lze jednotku práce aktualizovat v jiném podprocesu pomocí stejného manipulátoru.

Pokud nezadáte volbu sdílení manipulátoru, výchozí nastavení bude určeno prostředím:

- Windows V prostředí serveru Microsoft Transaction Server (MTS) je výchozí hodnota stejná jako hodnota MQCNO_HANDLE_SHARE_BLOCK.
- V jiných prostředích je výchozí hodnota stejná jako hodnota MQCNO_HANDLE_SHARE_NONE.

Volby opětovného připojení

Volby opětovného připojení určují, zda lze připojení znovu připojit. Lze znovu připojit pouze klientská připojení.

MQCNO_RECONNECT_AS_DEF

Volba opětovného připojení se interpretuje na výchozí hodnotu. Není-li nastavena žádná výchozí hodnota, hodnota této volby se interpretuje jako DISABLED. Hodnota volby je předána serveru a může být dotazována PCF a MQSC.

MQCNO_RECONNECT

Aplikaci lze znovu připojit k libovolnému správci front, který je konzistentní s hodnotou parametru **QmgrName** parametru MQCONNX. Volbu MQCNO_RECONNECT použijte pouze v případě, že mezi klientskou aplikací a správcem front, se kterým původně navázala připojení, neexistuje žádná afinita. Hodnota volby je předána serveru a může být dotazována PCF a MQSC.

MQCNO_RECONNECT_DISABLED

Aplikaci nelze znovu připojit. Hodnota volby není předána serveru.

MQCNO_RECONNECT_Q_MGR

Aplikaci lze znovu připojit pouze ke správci front, s nímž byla původně připojena. Tuto hodnotu použijte, pokud lze klienta znovu připojit, ale existuje afinita mezi aplikací klienta a správcem front, se kterým původně navázal připojení. Tuto hodnotu zvolte tehdy, chcete-li, aby se klient automaticky připojil znovu k instanci značně dostupného správce front, která je v pohotovostním režimu. Hodnota volby je předána serveru a může být dotazována PCF a MQSC.

Volby MQCNO_RECONNECT, MQCNO_RECONNECT_DISABLED a MQCNO_RECONNECT_Q_MGR použijte pouze pro připojení klienta. Pokud jsou volby použity pro připojení vazby, MQCONNX selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_OPTIONS_ERROR. Automatické opětovné připojení klienta není produktem IBM MQ classes for Java podporováno.

Možnosti sdílení konverzace

Následující volby platí pouze pro připojení klienta TCP/IP. Pro kanály SNA, SPX a NetBios jsou tyto hodnoty ignorovány a kanál je spuštěn stejně jako v předchozích verzích produktu.

MQCNO_NO_CONV_SHARING

Tato volba nepovoluje sdílení konverzace.

Volbu MQCNO_NO_CONV_SHARING můžete použít v situacích, kdy jsou konverzace silně zatíženy, a proto je možné soupeření na konci připojení serveru instance kanálu, na které existují konverzace sdílení. MQCNO_NO_CNV_SHARING se chová jako SHARECNV (1) při připojení ke kanálu, který podporuje sdílení konverzace, a SHARECNV (0) při připojení ke kanálu, který nepodporuje sdílení konverzace.

MQCNO_ALL_CONVS_SHARE

Tato volba povoluje sdílení konverzace; aplikace nepovoluje žádné omezení počtu připojení v instanci kanálu. Tato volba je výchozí hodnota.

Pokud aplikace označuje, že instance kanálu může sdílet sdílení, ale definice *SharingConversations* (SHARECNV) na konci připojení serveru kanálu je nastavena na hodnotu jedna, nedojde k žádnému sdílení a aplikaci se nevydá žádné varování.

Podobně, pokud aplikace označuje, že je povoleno sdílení, ale definice připojení serveru *SharingConversations* je nastavena na nulu, není zobrazeno žádné varování a aplikace vykazuje stejné chování jako klient ve verzích starších než IBM WebSphere MQ 7.0; nastavení aplikace týkající se sdílení konverzací je ignorováno.

MQCNO_NO_CONV_SHARING a MQCNO_ALL_CONVS_SHARE se vzájemně vylučují. Jsou-li pro konkrétní připojení zadány obě volby, bude připojení odmítnuto s kódem příčiny MQRC_OPTIONS_ERROR.

Volby definice kanálu

Následující volby řídí použití struktury definice kanálu předané v MQCNO:

MQCNO_CD_FOR_OUTPUT_ONLY

Tato volba povoluje použití struktury definice kanálu v MQCNO pouze k vrácení názvu kanálu použitého při úspěšném volání MQCONNX.

Pokud není poskytnuta platná struktura definice kanálu, volání se nezdaří s kódem příčiny MQRC_CD_ERROR.

Pokud aplikace není spuštěna jako klient, volba se ignoruje.

Vrácený název kanálu lze použít při následném volání MQCONNX pomocí volby MQCNO_USE_CD_SELECTION, abyste se znovu připojili pomocí stejné definice kanálu. To může být užitečné v případě, že tabulka kanálů klienta obsahuje více použitelných definic kanálů.

MQCNO_USE_CD_SELECTION

Tato volba umožňuje volání MQCONNX připojit se pomocí názvu kanálu obsaženého ve struktuře definice kanálu předané v MQCNO.

Je-li nastavena proměnná prostředí MQSERVER, použije se definice kanálu, kterou definuje. Není-li MQSERVER nastaven, použije se tabulka kanálů klienta.

Není-li nalezena definice kanálu s odpovídajícím názvem kanálu a názvem správce front, volání se nezdaří s kódem příčiny MQRC_Q_MGR_NAME_ERROR.

Pokud není poskytnuta platná struktura definice kanálu, volání se nezdaří s kódem příčiny MQRC_CD_ERROR.

Pokud aplikace není spuštěna jako klient, volba se ignoruje.

Výchozí volba

Pokud nevyžadujete žádnou z výše uvedených voleb, můžete použít následující volbu:

MQCNO_NONE

Nejsou zadány žádné volby.

K podpoře dokumentace programu použijte MQCNO_NONE. Není zamýšleno, aby tato volba byla použita s jinou volbou MQCNO_ *, ale protože její hodnota je nula, nelze takové použití zjistit.

ClientConnOffset (MQLONG) pro MQCNO

ClientConnOffset je posun v bajtech struktury definice kanálu MQCD od začátku struktury MQCNO. Posun může být kladný nebo záporný. Toto pole je vstupní pole s počáteční hodnotou 0.

Parametr *ClientConnOffset* používejte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client. Chcete-li získat informace o tom, jak používat toto pole, prohlédněte si popis pole *ClientConnPtr* .

Toto pole je ignorováno, pokud je *Version* menší než MQCNO_VERSION_2.

ClientConnPtr (MQPTR) pro MQCNO

ClientConnPtr je vstupní pole. Jeho počáteční hodnotou je ukazatel Null v programovacích jazycích, které podporují ukazatele, a řetězec bajtů s hodnotou all-null, jinak.

ClientConnOffset a *ClientConnPtr* používejte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client. Zadáním jednoho nebo více těchto polí může aplikace řídit definici kanálu připojení klienta poskytnutím struktury definice kanálu MQCD, která obsahuje požadované hodnoty.

Pokud je aplikace spuštěna jako IBM MQ MQI client, ale neposkytuje strukturu MQCD, použije se k výběru definice kanálu proměnná prostředí MQSERVER . Není-li parametr MQSERVER nastaven, použije se tabulka kanálů klienta.

Pokud aplikace není spuštěna jako IBM MQ MQI client, *ClientConnOffset* a *ClientConnPtr* jsou ignorovány.

Pokud aplikace poskytuje strukturu MQCD, nastavte pole uvedená na požadované hodnoty; ostatní pole v MQCD jsou ignorována. Můžete vyplnit znakové řetězce mezerami do délky pole, nebo je ukončit znakem null. Další informace o polích ve struktuře MQCD naleznete v části “Pole” na stránce 1472 .

Tabulka 481. Pole v MQCD

Pole v MQCD	Hodnota
ChannelName	Název kanálu.
Version	Číslo verze struktury. Nesmí být menší než MQCD_VERSION_7.
TransportType	Libovolný podporovaný typ přenosu.
ModeName	Název režimu LU 6.2 .
TpName	Název transakčního programu LU 6.2 .
SecurityExit	Název uživatelské procedury pro zabezpečení zprávy kanálu.
SendExit	Název uživatelské procedury pro odesílání kanálu.
ReceiveExit	Název uživatelské procedury pro příjem kanálu.
MaxMsgLength	Maximální délka zpráv v bajtech, které lze odeslat prostřednictvím kanálu připojení klienta.
SecurityUserData	Uživatelská data pro uživatelskou proceduru zabezpečení.
SendUserData	Uživatelská data pro uživatelskou proceduru odeslání.
ReceiveUserData	Uživatelská data pro uživatelskou proceduru příjmu.
UserIdentifier	Identifikátor uživatele, který se má použít k zavedení relace LU 6.2 .
Password	Heslo, které má být použito k ustanovení relace LU 6.2 .
ConnectionName	Název připojení.
HeartbeatInterval	Doba v sekundách mezi toky prezenčního signálu.
StrucLength	Délka struktury MQCD.
ExitNameLength	Délka názvů uživatelských procedur adresovaných znaky *SendExitPtr* a *ReceiveExitPtr*. Musí být větší než nula, pokud je parametr *SendExitPtr* nebo *ReceiveExitPtr* nastaven na hodnotu, která není ukazatelem Null.
ExitDataLength	Délka dat ukončení adresovaných pomocí *SendUserDataPtr* a *ReceiveUserDataPtr*. Musí být větší než nula, pokud je parametr *SendUserDataPtr* nebo *ReceiveUserDataPtr* nastaven na hodnotu, která není ukazatelem Null.

Tabulka 481. Pole v MQCD (pokračování)

Pole v MQCD	Hodnota
SendExitsDefined	Počet uživatelských procedur odeslání adresovaných pomocí *SendExitPtr*. Je-li nula, *SendExit* a *SendUserData* poskytují název uživatelské procedury a data. Je-li větší než nula, *SendExitPtr* a *SendUserDataPtr* poskytují názvy a data ukončení a *SendExit* a *SendUserData* musí být prázdné.
ReceiveExitsDefined	Počet uživatelských procedur pro příjem adresovaných pomocí *ReceiveExitPtr*. Je-li nula, *ReceiveExit* a *ReceiveUserData* poskytují název uživatelské procedury a data. Je-li větší než nula, *ReceiveExitPtr* a *ReceiveUserDataPtr* poskytují názvy a data ukončení a *ReceiveExit* a *ReceiveUserData* musí být prázdné.
SendExitPtr	Adresa jména první uživatelské procedury pro odeslání.
SendUserDataPtr	Adresa dat pro první uživatelskou proceduru odeslání.
ReceiveExitPtr	Adresa jména první uživatelské procedury pro příjem.
ReceiveUserDataPtr	Adresa dat pro první uživatelskou proceduru pro příjem.
LongRemoteUserIdLength	Délka dlouhého identifikátoru vzdáleného uživatele.
LongRemoteUserIdPtr	Adresa dlouhého identifikátoru vzdáleného uživatele.
RemoteSecurityId	Identifikátor vzdáleného zabezpečení.
SSLCipherSpec	TLS CipherSpec.
SSLPeerNamePtr	Adresa názvu partnera TLS.
SSLPeerNameLength	Délka názvu partnera TLS.
KeepAliveInterval	Hodnota předaná do zásobníku komunikací pro časování udržení aktivity pro kanál
LocalAddress	Lokální komunikační adresa, včetně adresy IP adaptéru lokální sítě, který se má použít, a rozsah portů, které se mají použít pro odchozí připojení.

Zadejte strukturu definice kanálu jedním ze dvou způsobů:

- Pomocí pole offsetu *ClientConnOffset*

  V tomto případě musí aplikace deklarovat složenou strukturu obsahující MQCNO následovanou strukturou definice kanálu MQCD a nastavit hodnotu *ClientConnOffset* na posun struktury definice kanálu od začátku MQCNO. Ujistěte se, že je toto posunutí správné. Parametr *ClientConnPtr* musí být nastaven na nulový ukazatel nebo nulový počet bajtů.

  Použijte *ClientConnOffset* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele způsobem, který není přenositelný do různých prostředí (například programovací jazyk COBOL).

  Pro programovací jazyk Visual Basic se jedná o složenou strukturu s názvemMQCNOCD je uveden v souboru záhlaví CMQXB.BAS; tato struktura obsahuje strukturu MQCNO následovanou strukturou MQCD. Inicializujte MQCNOCD vyvoláním subrutiny MQCNOCD_DEFAULTS. MQCNOCD se používá sVarianta MQCONNXAny volání MQCONNX; další podrobnosti naleznete v popisu volání MQCONNX.
- Pomocí pole ukazatele *ClientConnPtr*

  V tomto případě může aplikace deklarovat strukturu definice kanálu odděleně od struktury MQCNO a nastavit *ClientConnPtr* na adresu struktury definice kanálu. Nastavte *ClientConnOffset* na nulu.

  ClientConnPtr použijte pro programovací jazyky, které podporují datový typ ukazatele způsobem, který je přenositelný do různých prostředí (například programovací jazyk C).

V programovacím jazyku C můžete použít proměnnou makra MQCD_CLIENT_CONN_DEFAULT k poskytnutí počátečních hodnot pro strukturu, které jsou vhodnější pro použití ve volání MQCONNX, než počáteční hodnoty poskytované MQCD_DEFAULT.

Bez ohledu na techniku, kterou zvolíte, můžete použít pouze jednu z možností *ClientConnOffset* a *ClientConnPtr* ; volání selže s kódem příčiny MQRC_CLIENT_CONN_ERROR, pokud jsou obě nenulové.

Po dokončení volání MQCONNX není struktura MQCD znovu odkazována.

Toto pole je ignorováno, pokud je *Version* menší než MQCNO_VERSION_2.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky, přičemž počáteční hodnota je bajtový řetězec s hodnotou typu all-null.

ConnTag (MQBYTE128) pro MQCNO na více platformách

Značka připojení je koncepčně podobná identifikátoru připojení, ale může zahrnovat více souvisejících připojení a identifikovat je jako jedinou instanci aplikace. V systému Multiplatforms je značka připojení generována správcem front v době připojení.

Další informace viz identifikátor připojení a instance aplikace.

Vygenerované značky připojení jsou `semi` čitelné pro člověka. To znamená, že mohou být zobrazeny a filtrovány v prostředí MQSC jako řetězce v lokální znakové sadě. Připojení, která produkt IBM MQ zná jako související, jsou automaticky přiřazena ke stejné značce připojení. Toto přiřazení je zvláště důležité pro vyvažování aplikací.

Vygenerovaná značka připojení je viditelná třemi způsoby:

- Je-li ve výstupní struktuře MQCNO ve volání MQCONNX zadána hodnota MQCNO_GENERATE_CONN_TAG .
- Ve výstupu příkazu DISPLAY CONN (nebo programových ekvivalentů).
- Ve výstupu příkazu DISPLAY APSTATUS (nebo ekvivalentů).

Značka přestane být platná, když aplikace ukončí nebo vydá volání MQDISC.

Související odkazy

“ConnTag (MQBYTE128) pro MQCNO na IBM MQ for z/OS” na stránce 333
Značka připojení je koncepčně podobná identifikátoru připojení, ale může zahrnovat více souvisejících připojení a identifikovat je jako jedinou instanci aplikace. V systému IBM MQ for z/OSje značka připojení vstupním polem poskytovaným aplikací a používaným ve spojení s volbami MQCNO_ * _CONN_TAG k serializaci připojení z dané instance aplikace.

z/OS *ConnTag (MQBYTE128) pro MQCNO na IBM MQ for z/OS*

Značka připojení je koncepčně podobná identifikátoru připojení, ale může zahrnovat více souvisejících připojení a identifikovat je jako jedinou instanci aplikace. V systému IBM MQ for z/OSje značka připojení vstupním polem poskytovaným aplikací a používaným ve spojení s volbami MQCNO_ * _CONN_TAG k serializaci připojení z dané instance aplikace.

Pokud existuje více instancí aplikace, které mají být současně připojeny, musí každá z nich dodat jedinečnou hodnotu pro toto pole. Další podrobnosti viz popisy těchto voleb značek připojení .

Notes:

- V systému IBM MQ for z/OSneexistuje žádný způsob, jak administrativně určit značku připojení přidruženou k aplikaci za běhu.
- Hodnoty značek připojení začínající na MQ velkými, malými nebo smíšenými písmeny v kódu ASCII nebo EBCDIC jsou vyhrazeny pro použití produkty IBM . Nepoužívejte hodnoty značek připojení začínající těmito písmeny.

Pokud nevyžadujete žádnou značku, použijte následující speciální hodnotu:

MQCT_NONE
Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQCT_NONE_ARRAY; tato konstanta má stejnou hodnotu jako MQCT_NONE, ale je polem znaků místo řetězce.

Pole ConnTag se používá při připojování ke správci front z/OS .

Délka tohoto pole je dána hodnotou MQ_CONN_TAG_LENGTH. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQCNO_VERSION_3.

Multi Informace o použití značky připojení v systému IBM MQ for Multiplatformsnaleznete v tématu "ConnTag (MQBYTE128) pro MQCNO na více platformách" na stránce 333 .

SSLConfigPtr (PMQSCO) pro MQCNO

SSLConfigPtr je vstupní pole. Jeho počáteční hodnotou je ukazatel Null v programovacích jazycích, které podporují ukazatele, a řetězec bajtů s hodnotou all-null, jinak.

Volby *SSLConfigPtr* a *SSLConfigOffset* používejte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client a protokol kanálu je TCP/IP. Není-li aplikace spuštěna jako klient IBM MQ nebo není-li kanálovým protokolem protokol TCP/IP, jsou *SSLConfigPtr* a *SSLConfigOffset* ignorovány.

Zadáním *SSLConfigPtr* nebo *SSLConfigOffset*plus *ClientConnPtr* nebo *ClientConnOffset*může aplikace řídit použití TLS pro připojení klienta. Jsou-li tímto způsobem zadány informace o protokolu TLS, budou proměnné prostředí MQSSLKEYR a MQSSLCRYP ignorovány; veškeré informace související s protokolem TLS v tabulce CCDT (Client Channel Definition Table) budou rovněž ignorovány.

Informace TLS lze zadat pouze na:

- první volání MQCONNX procesu klienta, nebo
- Následné volání MQCONNX po dokončení všech předchozích připojení TLS ke správci front pomocí MQDISC.

Jedná se o jediný stav, ve kterém lze inicializovat prostředí TLS pro celý proces. Pokud je zadáno volání MQCONNX určující informace o protokolu TLS, když prostředí TLS již existuje, jsou informace o protokolu TLS ve volání ignorovány a připojení je provedeno pomocí existujícího prostředí TLS; volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_SSL_ALREADY_INICIALIZOVANÉ v tomto případě.

Strukturu MQSCO můžete poskytnout stejným způsobem jako strukturu MQCD, a to buď zadáním adresy v souboru *SSLConfigPtr*, nebo zadáním odchylky v souboru *SSLConfigOffset* ; Podrobnosti o tom, jak to provést, naleznete v popisu souboru *ClientConnPtr* . Avšak můžete použít pouze jeden z *SSLConfigPtr* a *SSLConfigOffset* ; volání selže s kódem příčiny MQRC_SSL_CONFIG_ERROR. pokud jsou obě nenulové.

Po dokončení volání MQCONNX není struktura MQSCO znovu odkazována.

Toto pole je ignorováno, pokud je hodnota *Version* menší než hodnota MQCNO_VERSION_4.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

SSLConfigOffset (MQLONG) pro MQCNO

SSLConfigOffset je posun struktury MQSCO v bajtech od začátku struktury MQCNO. Posun může být kladný nebo záporný. Toto pole je vstupní pole s počáteční hodnotou 0.

Parametr *SSLConfigOffset* používejte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client. Chcete-li získat informace o tom, jak používat toto pole, prohlédněte si popis pole *SSLConfigPtr* .

Toto pole je ignorováno, pokud je hodnota *Version* menší než hodnota MQCNO_VERSION_4.

ConnectionId (MQBYTE24) pro MQCNO

ConnectionId je jedinečný 24bajtový identifikátor, který produktu IBM MQ umožňuje spolehlivou identifikaci aplikace. Aplikace může tento identifikátor použít pro korelaci ve voláních PUT a GET. Tento výstupní parametr má počáteční hodnotu 24 nulových bajtů ve všech programovacích jazycích.

Správce front přiřadí všem připojením jedinečné ID, avšak budou vytvořena. Pokud MQCONNX vytvoří připojení s verzí 5 MQCNO, aplikace může určit ConnectionId z vráceného MQCNO. Je zaručeno, že přiřazený identifikátor bude jedinečný mezi všemi ostatními identifikátory, které produkt IBM MQ generuje, například CorrelId, MsgIDa GroupId.

ConnectionId slouží k identifikaci dlouho běžících pracovních jednotek pomocí příkazu PCF Inquire Connection nebo příkazu MQSC DISPLAY CONN. Hodnota ConnectionId používaná příkazy MQSC (CONN) je odvozena od ConnectionId vráceného zde. Příkazy PCF Inquire a Stop Connection mohou používat identifikátor ConnectionId , který je zde vrácen bez úprav.

Pomocí parametru ConnectionId můžete vynutit ukončení dlouho běžící jednotky zadáním ConnectionId pomocí příkazu PCF Stop Connection nebo příkazu MQSC STOP CONN. Další informace o použití těchto příkazů viz Zastavit připojení a STOP CONN .

Toto pole není vráceno, pokud je verze menší než MQCNO_VERSION_5.

Délka tohoto pole je dána hodnotou MQ_CONNECTION_ID_LENGTH.

SecurityParmsOffset (MQLONG) pro MQCNO

SecurityParmsOffset je posun struktury MQCSP v bajtech od začátku struktury MQCNO. Posun může být kladný nebo záporný. Toto pole je vstupní pole s počáteční hodnotou 0.

Toto pole je ignorováno, pokud je *Verze* menší než MQCNO_VERSION_5.

Struktura MQCSP je definována v souboru “MQCSP-parametry zabezpečení” na stránce 337.

SecurityParmsPtr (PMQCSP) pro MQCNO

SecurityParmsPtr je adresa struktury MQCSP, která se používá k určení ID uživatele a hesla pro ověření autorizační službou. Toto pole je vstupní pole a jeho počáteční hodnota je ukazatel s hodnotou null nebo bajty s hodnotou null.

Toto pole je ignorováno, pokud je *Verze* menší než MQCNO_VERSION_5.

Struktura MQCSP je definována v souboru “MQCSP-parametry zabezpečení” na stránce 337.

Vyhrazeno (MQBYTE4) pro MQCNO

Vyhrazené pole pro vyplnění struktury na 64bitovou hranici. Počáteční hodnota pole je binární nula pro délku pole.

Toto pole je ignorováno, pokud je hodnota `Version` menší než MQCNO_VERSION_6.

CCDTUrlLength (MQLONG) pro MQCNO

CCDTUrlLength je délka řetězce identifikovaného buď CCDTUrlPtr , nebo CCDTUrlOffset , který obsahuje URL , která identifikuje umístění tabulky kanálu připojení klienta, která se má použít pro připojení. Počáteční hodnota pole je nula.

Parametr `CCDTUrlLength` používejte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client.

Jedná se o programovou alternativu k nastavení proměnných prostředí MQCHLLIB a MQCHLTAB .

Není-li aplikace spuštěna jako klient, je parametr `CCDTUrlLength` ignorován.

Toto pole je ignorováno, pokud je hodnota `Version` menší než MQCNO_VERSION_6.

CCDTUrlPtr (PMQCHAR) pro MQCNO

CCDTUrlPtr je volitelný ukazatel na řetězec, který obsahuje URLpro identifikaci umístění tabulky kanálů připojení klienta, která se má použít pro připojení. Toto pole je vstupní pole s počáteční hodnotou

ukazatele Null v programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou Null.

Parametr `CCDTUrlPtr` používejte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client.

Důležité: Můžete použít pouze jednu z možností `CCDTUrlPtr` a `CCDTUrlOffset`. Volání selže s kódem příčiny MQRC_CCDT_URL_ERROR, pokud jsou obě pole nenulová.

Jedná se o programovou alternativu k nastavení proměnných prostředí MQCHLLIB a MQCHLTAB .

Není-li aplikace spuštěna jako klient, je parametr `CCDTUrlPtr` ignorován.

Toto pole je ignorováno, pokud je hodnota `Version` menší než MQCNO_VERSION_6.

CCDTUrlOffset (MQLONG) pro MQCNO

CCDTUrlOffset je posun v bajtech od začátku struktury MQCNO k řetězci, který obsahuje URL identifikující umístění tabulky kanálů připojení klienta, která se má použít pro připojení. Posun může být kladný nebo záporný a počáteční hodnota pole je nula.

Parametr `CCDTUrlOffset` používejte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client.

Důležité: Můžete použít pouze jednu z možností `CCDTUrlPtr` a `CCDTUrlOffset`. Volání selže s kódem příčiny MQRC_CCDT_URL_ERROR, pokud jsou obě pole nenulová.

Jedná se o programovou alternativu k nastavení proměnných prostředí MQCHLLIB a MQCHLTAB .

Není-li aplikace spuštěna jako klient, je parametr `CCDTUrlOffset` ignorován.

Toto pole je ignorováno, pokud je hodnota `Version` menší než MQCNO_VERSION_6.

ApplName (MQCHAR28) pro MQCNO

Název nastavený aplikací pro identifikaci připojení ke správci front. Počáteční hodnota pole je MQAN_NONE_ARRAY (prázdné znaky).

Toto pole je ignorováno, pokud je hodnota `Version` menší než MQCNO_VERSION_7, nebo pokud je hodnota nastavena na mezery.

z/OS Toto pole nelze nastavit v systému z/OS. Pokud se o to pokusíte, obdržíte zpět kód příčiny MQRC_CNO_ERROR.

Reserved2 (MQBYTE4) pro MQCNO

Vyhrazené pole pro vyplnění struktury na 64bitovou hranici. Počáteční hodnota pole je binární nula pro délku pole.

Toto pole je ignorováno, pokud je `Version` menší než MQCNO_VERSION_7.

V 9.3.0 *BalanceParmsOffset (MQLONG) pro MQCNO*

Umístění paměti pro strukturu typu MQBNO, která obsahuje informace o chování vyvažování aplikace. Struktura je zcela ignorována, pokud se aplikace nepřipojuje přes kanál klienta.

Toto pole je ignorováno, pokud je `Version` menší než MQCNO_VERSION_8.

Další informace viz MQBNO .

Zadáte-li toto pole, nemůžete zadat pole “BalanceParmsPtr (MQPTR) pro MQCNO” na stránce 336 . Pokusíte-li se zadat obě pole, obdržíte zprávu MQRC_CNO_ERROR. Vzhledem k tomu, že toto pole je relevantní pouze pro připojení klienta, bude výsledkem zadání tohoto pole pro jakýkoli jiný typ připojení také hodnota MQRC_CNO_ERROR.

V 9.3.0 *BalanceParmsPtr (MQPTR) pro MQCNO*

Ukazatel na umístění paměti pro strukturu typu MQBNO, která obsahuje informace o chování aplikace při vyrovnávání. Struktura je zcela ignorována, pokud se aplikace nepřipojuje přes kanál klienta.

Toto pole je ignorováno, pokud je `Version` menší než MQCNO_VERSION_8.

Další informace viz MQBNO .

Zadáte-li toto pole, nemůžete zadat pole “BalanceParmsOffset (MQLONG) pro MQCNO” na stránce 336 . Pokusíte-li se zadat obě pole, obdržíte zprávu MQRC_CNO_ERROR. Vzhledem k tomu, že toto pole je relevantní pouze pro připojení klienta, bude výsledkem zadání tohoto pole pro jakýkoli jiný typ připojení také hodnota MQRC_CNO_ERROR.

MQCSP-parametry zabezpečení

Strukturu parametrů zabezpečení připojení IBM MQ používají aplikace k toku ověřovacích informací ve volání MQCONNX do správce front. Můžete jej také použít k poskytnutí počátečního klíče, který se používá se systémem ochrany heslem IBM MQ , který šifruje citlivá data.

Nastavte parametr *AuthenticationType* na hodnotu `MQCSP_AUTH_USER_ID_AND_PWD` , chcete-li zahrnout ID uživatele a heslo z verze 1.

Zadáte-li počáteční informace o klíči ve verzi 2, výchozí hodnota parametru *AuthenticationType* je `MQCSP_AUTH_NONE`.

V 9.3.4 Od IBM MQ 9.3.4použijte *AuthenticationType* k zahrnutí informací o tokenu ověření.

V 9.3.4 Můžete použít `MQCSP_AUTH_USER_ID_AND_PWD` nebo `MQCSP_AUTH_ID_TOKEN`, ale ne obojí.

Varování: V některých případech je heslo nebo token ověření ve struktuře MQCSP pro klientskou aplikaci odesláno po síti jako prostý text. Chcete-li se ujistit, že hesla aplikace klienta a tokeny ověření jsou odpovídajícím způsobem chráněny, prohlédněte si téma Ochrana pomocí hesla MQCSP.

Dostupnost

Struktura MQCSP je k dispozici na všech podporovaných platformách IBM MQ .

Verze

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi MQCSP, ale s počáteční hodnotou pole *Version* nastavenou na MQCSP_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole *Version* na požadované číslo verze.

Znaková sada a kódování

Data v protokolu MQCSP musí být ve znakové sadě a kódování lokálního správce front, které jsou dány atributem správce front **CodedCharSetId** a MQENC_NATIVE.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 482. Pole v MQCSP pro MQCSP

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQCSP_STRUC_ID	`'CSP¬'`
Verze (číslo verze struktury)	MQCSP_VERSION_1	1
AuthenticationType (typ ověřování)	Není	`MQCSP_AUTH_NONE`

Tabulka 482. Pole v MQCSP pro MQCSP (pokračování)

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
Reserved1 (požadováno pro zarovnání ukazatele na IBM i)	Není	Prázdný řetězec nebo mezery
CSPUserIdPtr (adresa ID uživatele)	Není	Ukazatel Null nebo bajty s hodnotou Null
CSPUserIdOffset (posunutí ID uživatele)	Není	0
CSPUserIdDélka (délka ID uživatele)	Není	0
Reserved2 (požadováno pro zarovnání ukazatele na IBM i)	Není	Prázdný řetězec nebo mezery
CSPPasswordPtr (adresa hesla)	Není	Ukazatel Null nebo bajty s hodnotou Null
CSPPasswordOffset (posunutí hesla)	Není	0
CSPPasswordLength (délka hesla)	Není	0
Poznámka: Zbývající pole se ignorují, pokud je *Version* menší než MQCSP_VERSION_2.		
V 9.3.0 V 9.3.0 Reserved3 (požadováno pro zarovnání ukazatele na IBM i)	Není	Prázdný řetězec nebo mezery
V 9.3.0 V 9.3.0 InitialKeyPtr	Není	Nulový ukazatel nebo mezery
V 9.3.0 V 9.3.0 InitialKeyOffset (posun počátečního klíče pro systém ochrany heslem)	Není	0
V 9.3.0 V 9.3.0 InitialKeyDélka (délka počátečního klíče pro systém ochrany heslem)	Není	0
Poznámka: Zbývající pole se ignorují, pokud je *Version* menší než MQCSP_VERSION_3.		
V 9.3.4 Reserved4 (požadováno pro zarovnání ukazatele na IBM i)	Není	Prázdný řetězec nebo mezery
V 9.3.4 TokenPtr (adresa tokenu ověření)	Není	Nulový ukazatel nebo mezery
V 9.3.4 TokenKeyTokenKey (posun tokenu ověření)	Není	0
V 9.3.4 TokenLength (délka tokenu ověření)	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQCSP_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQCSP MyCSP = {MQCSP_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQCSP

V 9.3.0 V 9.3.0

```
typedef struct tagMQCSP MQCSP;
struct tagMQCSP {
  MQCHAR4    StrucId;            /* Structure identifier */
  MQLONG     Version;            /* Structure version number */
  MQLONG     AuthenticationType; /* Type of authentication */
  MQBYTE4    Reserved1;          /* Required for IBM i pointer alignment */
  MQPTR      CSPUserIdPtr;       /* Address of user ID */
  MQLONG     CSPUserIdOffset;    /* Offset of user ID */
  MQLONG     CSPUserIdLength;    /* Length of user ID */
  MQBYTE8    Reserved2;          /* Required for IBM i pointer alignment */
  MQPTR      CSPPasswordPtr;     /* Address of password */
  MQLONG     CSPPasswordOffset;  /* Offset of password */
  MQLONG     CSPPasswordLength;  /* Length of password */
/* Ver:1 */

  MQBYTE8    Reserved3;          /* Required for IBM i pointer alignment */
  MQPTR      InitialKeyPtr;      /* Address of initial key */
  MQLONG     InitialKeyOffset;   /* Offset of initial key */
  MQLONG     InitialKeyLength;   /* Length of initial key */
/* Ver:2 */
  };
```

V 9.3.4

```
typedef struct tagMQCSP MQCSP;
struct tagMQCSP {
  MQCHAR4    StrucId;            /* Structure identifier */
  MQLONG     Version;            /* Structure version number */
  MQLONG     AuthenticationType; /* Type of authentication */
  MQBYTE4    Reserved1;          /* Required for IBM i pointer alignment */
  MQPTR      CSPUserIdPtr;       /* Address of user ID */
  MQLONG     CSPUserIdOffset;    /* Offset of user ID */
  MQLONG     CSPUserIdLength;    /* Length of user ID */
  MQBYTE8    Reserved2;          /* Required for IBM i pointer alignment */
  MQPTR      CSPPasswordPtr;     /* Address of password */
  MQLONG     CSPPasswordOffset;  /* Offset of password */
  MQLONG     CSPPasswordLength;  /* Length of password */
/* Ver:1 */

  MQBYTE8    Reserved3;          /* Required for IBM i pointer alignment */
  MQPTR      InitialKeyPtr;      /* Address of initial key */
  MQLONG     InitialKeyOffset;   /* Offset of initial key */
  MQLONG     InitialKeyLength;   /* Length of initial key */
/* Ver:2 */

  MQBYTE8    Reserved4;          /* Required for IBM i pointer alignment */
  MQPTR      TokenPtr;           /* Address of token */
  MQLONG     TokenOffset;        /* Offset of token */
  MQLONG     TokenLength;        /* Length of token */
/* Ver:3 */
  };
```

Deklarace jazyka COBOL pro MQCSP

V 9.3.0 V 9.3.0

```
**   MQCSP structure
  10 MQCSP.
**    Structure identifier
   15 MQCSP-STRUCID           PIC X(4).
**    Structure version number
   15 MQCSP-VERSION           PIC S9(9) BINARY.
**    Type of authentication
   15 MQCSP-AUTHENTICATIONTYPE PIC S9(9) BINARY.
**    Required for IBM i pointer alignment
   15 MQCSP-RESERVED1         PIC X(4).
**    Address of user ID
   15 MQCSP-CSPUSERIDPTR      POINTER.
**    Offset of user ID
   15 MQCSP-CSPUSERIDOFFSET   PIC S9(9) BINARY.
**    Length of user ID
   15 MQCSP-CSPUSERIDLENGTH   PIC S9(9) BINARY.
**    Required for IBM i pointer alignment
```

```
   15 MQCSP-RESERVED2         PIC X(4).
**    Address of password
   15 MQCSP-CSPPASSWORDPTR    POINTER.
**    Offset of password
   15 MQCSP-CSPPASSWORDOFFSET PIC S9(9) BINARY.
**    Length of password
   15 MQCSP-CSPPASSWORDLENGTH PIC S9(9) BINARY.
** Ver:1 **

**    Reserved
   15 MQCSP-RESERVED3         PIC X(8).
**    Address of initial key
   15 MQCSP-INITIALKEYPTR     POINTER.
**    Offset of initial key
   15 MQCSP-INITIALKEYOFFSET  PIC S9(9) BINARY.
**    Length of initial key
   15 MQCSP-INITIALKEYLENGTH  PIC S9(9) BINARY.
** Ver:2 **
```

V 9.3.4

```
**   MQCSP structure
  10 MQCSP.
**    Structure identifier
   15 MQCSP-STRUCID           PIC X(4).
**    Structure version number
   15 MQCSP-VERSION           PIC S9(9) BINARY.
**    Type of authentication
   15 MQCSP-AUTHENTICATIONTYPE PIC S9(9) BINARY.
**    Required for IBM i pointer alignment
   15 MQCSP-RESERVED1         PIC X(4).
**    Address of user ID
   15 MQCSP-CSPUSERIDPTR      POINTER.
**    Offset of user ID
   15 MQCSP-CSPUSERIDOFFSET   PIC S9(9) BINARY.
**    Length of user ID
   15 MQCSP-CSPUSERIDLENGTH   PIC S9(9) BINARY.
**    Required for IBM i pointer alignment
   15 MQCSP-RESERVED2         PIC X(4).
**    Address of password
   15 MQCSP-CSPPASSWORDPTR    POINTER.
**    Offset of password
   15 MQCSP-CSPPASSWORDOFFSET PIC S9(9) BINARY.
**    Length of password
   15 MQCSP-CSPPASSWORDLENGTH PIC S9(9) BINARY.
** Ver:1 **

**    Reserved
   15 MQCSP-RESERVED3         PIC X(8).
**    Address of initial key
   15 MQCSP-INITIALKEYPTR     POINTER.
**    Offset of initial key
   15 MQCSP-INITIALKEYOFFSET  PIC S9(9) BINARY.
**    Length of initial key
   15 MQCSP-INITIALKEYLENGTH  PIC S9(9) BINARY.
** Ver:2 **

**    Reserved
   15 MQCSP-RESERVED4         PIC X(8).
**    Address of token
   15 MQCSP-TOKENPTR     POINTER.
**    Offset of token
   15 MQCSP-TOKENOFFSET  PIC S9(9) BINARY.
**    Length of token
   15 MQCSP-TOKENLENGTH  PIC S9(9) BINARY.
** Ver:3 **
```

Deklarace PL/I pro MQCSP

V 9.3.0 V 9.3.0

```
dcl
 1 MQCSP based,
  3 StrucId           char(4),       /* Structure identifier */
  3 Version           fixed bin(31), /* Structure version number */
  3 AuthenticationType fixed bin(31),  /* Type of authentication */
  3 Reserved1         char(4),       /* Required for IBM i pointer
                                        alignment */
  3 CSPUserIdPtr      pointer,       /* Address of user ID */
  3 CSPUserIdOffset   fixed bin(31), /* Offset of user ID */
```

```
  3 CSPUserIdLength   fixed bin(31), /* Length of user ID */
  3 Reserved2         char(8),       /* Required for IBM i pointer
                                        alignment */
  3 CSPPasswordPtr    pointer,       /* Address of password */
  3 CSPPasswordOffset fixed bin(31), /* Offset of user ID */
  3 CSPPasswordLength fixed bin(31), /* Length of user ID */
/* Version 1 */

  3 Reserved3         char(8),       /* Reserved */
  3 InitialKeyPtr     pointer,       /* Address of initial key */
  3 InitialKeyOffset  fixed bin(31), /* Offset of initial key */
  3 InitialKeyLength  fixed bin(31); /* Length of initial key */
/* Version 2 */
```

V 9.3.4

```
dcl
 1 MQCSP based,
  3 StrucId           char(4),       /* Structure identifier */
  3 Version           fixed bin(31), /* Structure version number */
  3 AuthenticationType fixed bin(31),  /* Type of authentication */
  3 Reserved1         char(4),       /* Required for IBM i pointer
                                        alignment */
  3 CSPUserIdPtr      pointer,       /* Address of user ID */
  3 CSPUserIdOffset   fixed bin(31), /* Offset of user ID */
  3 CSPUserIdLength   fixed bin(31), /* Length of user ID */
  3 Reserved2         char(8),       /* Required for IBM i pointer
                                        alignment */
  3 CSPPasswordPtr    pointer,       /* Address of password */
  3 CSPPasswordOffset fixed bin(31), /* Offset of user ID */
  3 CSPPasswordLength fixed bin(31), /* Length of user ID */
/* Version 1 */

  3 Reserved3         char(8),       /* Reserved */
  3 InitialKeyPtr     pointer,       /* Address of initial key */
  3 InitialKeyOffset  fixed bin(31), /* Offset of initial key */
  3 InitialKeyLength  fixed bin(31); /* Length of initial key */
/* Version 2 */

  3 Reserved4         char(8),       /* Reserved */
  3 TokenPtr          pointer,       /* Address of Token */
  3 TokenOffset       fixed bin(31), /* Offset of Token */
  3 TokenLength       fixed bin(31); /* Length of Token */
/* Version 3 */
```

Deklarace jazyka Visual Basic pro MQCSP

```
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
```

Související pojmy

Práce s tokeny ověření

StrucId (MQCHAR4) pro MQCSP.

Jedná se o identifikátor struktury struktury parametrů zabezpečení. Vždy se jedná o vstupní pole. Jeho hodnota je MQCSP_STRUC_ID.

Hodnota musí být:

MQCSP_STRUC_ID
Identifikátor struktury parametrů zabezpečení.

Pro programovací jazyk C je definována také konstanta MQCSP_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQCSP_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQCSP.
Číslo verze struktury MQCSP.

Hodnota musí být:

MQCSP_VERSION_1
Struktura parametrů zabezpečení Version-1 . Ve verzi 1 můžete do struktury MQCSP zahrnout ID uživatele a heslo pro ověření se správcem front.

V 9.3.0 V 9.3.0 **MQCSP_VERSION_2**
Struktura parametrů zabezpečení Version-2 . Ve verzi 2 můžete zahrnout ID uživatele a heslo pro ověření se správcem front a uvést počáteční klíč, který se používá k ochraně hesel.

V 9.3.4 **MQCSP_VERSION_3**
Version-3 struktura parametrů zabezpečení. Ve verzi 3 můžete do struktury MQCSP zahrnout buď ID uživatele a heslo, nebo token ověření pro ověření se správcem front. Můžete také uvést počáteční klíč, který se používá k ochraně hesel.

Následující konstanta určuje číslo verze aktuální verze:

MQCSP_CURRENT_VERSION
Aktuální verze struktury parametrů zabezpečení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCSP_VERSION_3.

AuthenticationType (MQLONG) pro MQCSP.
AuthenticationType je vstupní pole. Počáteční hodnota je MQCSP_AUTH_NONE.

Jedná se o typ ověřování, které má být provedeno. Platné jsou tyto hodnoty:

MQCSP_AUTH_NONE
Nepoužívejte pole ID uživatele a heslo nebo token ověření .

MQCSP_AUTH_USER_ID_AND_PWD
Proveďte ověření pomocí ID uživatele a hesla ve struktuře MQCSP.

V 9.3.4 **MQCSP_AUTH_ID_TOKEN**
Ověřit pomocí tokenu ověření ve struktuře MQCSP.

Výchozí hodnota je MQCSP_AUTH_NONE. Při výchozím nastavení se neprovádí žádná ochrana heslem.

Pokud vyžadujete ověření, musíte nastavit **MQCSP**.**AuthenticationType** do MQCSP_AUTH_USER_ID_AND_PWD nebo MQCSP_AUTH_ID_TOKEN.

Další informace naleznete v tématu Ochrana heslem MQCSP.

Související pojmy
Práce s tokeny ověření

Reserved1 (MQBYTE4) pro MQCSP.
Vyhrazené pole požadované pro zarovnání ukazatele na IBM i.

Počáteční hodnota tohoto pole je null.

CSPUserIdPtr (MQPTR) pro MQCSP.
Adresa pro ID uživatele, které se má použít při ověření.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou typu all-null. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQCNO_VERSION_5.

Toto pole může obsahovat ID uživatele operačního systému, je-li v poli CONNAUTH správce front uvedena hodnota **AUTHTYPE** *IDPWOS* .

V systému Windows se může jednat o úplné ID uživatele domény.

Toto pole může obsahovat ID uživatele LDAP, je-li v poli CONNAUTH správce front uvedeno **AUTHTYPE** *IDPWLDAP* .

CSPUserIdOffset (MQLONG) pro MQCSP.
Posun v bajtech pro ID uživatele, který se má použít při ověření. Posun může být kladný nebo záporný.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSPUserIdDélka (MQLONG) pro MQCSP.
Délka ID uživatele, které má být použito při ověřování.

Maximální délka ID uživatele závisí na platformě, viz ID uživatele. V 9.3.0 V 9.3.0 Pokud je délka ID uživatele větší než maximální povolená délka, požadavek na ověření selže s MQRC_CSP_ERROR. Ve starších verzích systému IBM MQbyla vrácena chyba MQRC_NOT_AUTHORIZED.

Toto pole je vstupní pole. Počáteční hodnota tohoto pole je 0.

Reserved2 (MQBYTE8) pro MQCSP.
Vyhrazené pole požadované pro zarovnání ukazatele na IBM i.

Počáteční hodnota tohoto pole je null.

CSPPasswordPtr (MQPTR) pro MQCSP.
Adresa pro heslo, které se má použít při ověření.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou Null. Toto pole je ignorováno, pokud je hodnota `Version` menší než MQCNO_VERSION_5.

Toto pole může obsahovat prázdné heslo, které je odmítnuto kontrolou hesla operačního systému nebo LDAP, v závislosti na nastavení, ale není odmítnuto produktem IBM MQ před jeho předáním do metody ověření.

CSPPasswordOffset (MQLONG) pro MQCSP.
Toto je posun v bajtech pro heslo, které se má použít při ověření. Posun může být kladný nebo záporný.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSPPasswordLength (MQLONG) pro MQCSP.
Délka hesla, které se má použít při ověření.

Maximální délka hesla je MQ_CSP_PASSWORD_LENGTH, což je 256 znaků. V 9.3.0 V 9.3.0 Pokud je délka hesla větší než maximální povolená délka, požadavek na ověření selže s MQRC_CSP_ERROR. Ve starších verzích systému IBM MQbyla vrácena chyba MQRC_NOT_AUTHORIZED.

Toto pole je vstupní pole. Počáteční hodnota tohoto pole je 0.

V 9.3.0 V 9.3.0 *Reserved3 (MQBYTE8) pro MQCSP.*
Vyhrazené pole požadované pro zarovnání ukazatele na IBM i.

Počáteční hodnota tohoto pole je null.

V 9.3.0 Multi V 9.3.0 *InitialKeyPtr (MQPTR) pro MQCSP.*
Adresa pro počáteční klíč pro systém ochrany heslem.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou Null. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQCSP_VERSION_2.

Toto pole je relevantní pouze pro IBM MQ MQI clients spuštění na systémech IBM i, AIX, Linux, and Windows .

Produkt IBM MQ MQI clients může zadat šifrované hodnoty pro některá pole pomocí systému IBM MQ pro ochranu heslem. Pokud jste použili počáteční klíč k zašifrování hesla pro úložiště klíčů určené ve struktuře MQCSO, ujistěte se, že jste zahrnuli počáteční klíčová pole do MQCSP pro stejnou klientskou aplikaci.

IBM MQ Klienti MQI mohou pro některá pole zadat zašifrované hodnoty pomocí systému ochrany heslem IBM MQ . Pokud jste použili počáteční klíč k zašifrování hesla pro úložiště klíčů určené ve struktuře MQCSO, ujistěte se, že jste zahrnuli počáteční klíčová pole do struktury MQCSP pro stejnou klientskou aplikaci.

Počáteční klíč je používán šifrovacím algoritmem k šifrování a dešifrování těchto hodnot. Pokud je počáteční klíč zadán, když jsou hodnoty těchto polí šifrovány pomocí obslužného programu **runmqicred** , musí klient při připojení ke správci front zadat stejný počáteční klíč.

Počáteční klíč určený pomocí tohoto pole přepíše jakýkoli počáteční klíč určený pomocí proměnné prostředí *MQS_MQI_KEYFILE* nebo vlastnosti *MQIInitialKey* v sekci Zabezpečení konfiguračního souboru klienta.

K určení počátečního klíče můžete použít buď *InitialKeyOffset* , nebo *InitialKeyPtr* , ale ne obojí.

Související úlohy

Zadání počátečního klíče pro klienta MQI IBM MQ v systému AIX, Linux a Windows

Ochrana hesel v konfiguračních souborech komponenty IBM MQ

Související odkazy

runmqicred (chránit hesla klienta IBM MQ)

“KeyRepoPasswordPtr (MQPTR) pro MQSCO” na stránce 568
Jedná se o adresu hesla úložiště klíčů TLS v bajtech.

“InitialKeyOffset (MQLONG) pro MQCSP.” na stránce 344
Posun v bajtech pro počáteční klíč pro systém ochrany heslem od spuštění struktury MQCSP. Posun může být kladný nebo záporný.

V 9.3.0 Multi V 9.3.0 *InitialKeyOffset (MQLONG) pro MQCSP.*

Posun v bajtech pro počáteční klíč pro systém ochrany heslem od spuštění struktury MQCSP. Posun může být kladný nebo záporný.

K určení počátečního klíče můžete použít buď *InitialKeyOffset* , nebo *InitialKeyPtr* , ale ne obojí. Další informace viz popis pole *InitialKeyPtr* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQCSP_VERSION_2.

Související úlohy

Ochrana hesel v konfiguračních souborech komponenty IBM MQ

Zadání počátečního klíče pro klienta MQI IBM MQ v systému AIX, Linux a Windows

Související odkazy

“InitialKeyPtr (MQPTR) pro MQCSP.” na stránce 343
Adresa pro počáteční klíč pro systém ochrany heslem.

V 9.3.0 Multi V 9.3.0 *InitialKeyDélka (MQLONG) pro MQCSP.*

Délka počátečního klíče pro systém ochrany heslem.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQCSP_VERSION_2.

Související úlohy

Ochrana hesel v konfiguračních souborech komponenty IBM MQ

Zadání počátečního klíče pro klienta MQI IBM MQ v systému AIX, Linux a Windows

V 9.3.4 *Reserved4 (MQBYTE8) pro MQCSP.*

Vyhrazené pole požadované pro zarovnání ukazatele na IBM i.

Počáteční hodnota tohoto pole je null.

V 9.3.4 *TokenPtr (MQPTR) pro MQCSP.*

Adresa tokenu ověření, který se používá pro ověření se správcem front.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou Null. Toto pole je ignorováno, pokud je *Version* menší než MQCSP_VERSION_3.

Toto pole je relevantní pro IBM MQ MQI clients připojování ke správcům front systému IBM MQ , kteří jsou spuštění v systémech AIX nebo Linux .

K určení tokenu ověření můžete použít buď *TokenOffset* , nebo *TokenPtr* , ale ne obojí.

Další informace naleznete v tématu Použití tokenů ověření v aplikaci.

Související pojmy

Práce s tokeny ověření

Související odkazy

“TokenOffset (MQLONG) pro MQCSP.” na stránce 345
Toto je posun v bajtech pro token ověření od spuštění struktury MQCSP. Posun může být kladný nebo záporný.

V 9.3.4 *TokenOffset (MQLONG) pro MQCSP.*

Toto je posun v bajtech pro token ověření od spuštění struktury MQCSP. Posun může být kladný nebo záporný.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je *Version* menší než MQCSP_VERSION_3.

K určení tokenu můžete použít buď *TokenOffset* , nebo *TokenPtr* , ale ne obojí. Další informace viz popis pole *TokenPtr* .

Související pojmy

Práce s tokeny ověření

Související úlohy

Použití tokenů ověření v aplikaci

Související odkazy

“TokenPtr (MQPTR) pro MQCSP.” na stránce 345
Adresa tokenu ověření, který se používá pro ověření se správcem front.

V 9.3.4 *TokenLength (MQLONG) pro MQCSP.*

Jedná se o délku tokenu ověřování používaného pro ověřování se správcem front.

Maximální délka tokenu ověření je MQ_CSP_TOKEN_LENGTH, což je 8192 bajtů. Pokud je *TokenLength* větší než maximální povolená délka, požadavek na ověření selže s MQRC_CSP_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je *Version* menší než MQCSP_VERSION_3.

Další informace naleznete v tématu Použití tokenů ověření v aplikaci.

Související pojmy

Práce s tokeny ověření

MQCTLO-Struktura voleb zpětného volání řízení

Struktura MQCTLO se používá k určení voleb souvisejících s funkcí zpětného volání řízení. Struktura je vstupní a výstupní parametr volání MQCTL.

Dostupnost

Struktura MQCTLO je k dispozici na následujících platformách:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows
- 

z/OS

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Verze

Aktuální verze MQCTLO je MQCTLO_VERSION_1.

Znaková sada a kódování

Data v objektu MQCTLO musí být ve znakové sadě zadané atributem správce front **CodedCharSetId** a kódování lokálního správce front zadané proměnnou MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 483. Pole v MQCTLO

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucID (identifikátor struktury)	MQCTLO_STRUC_ID	'CTLO'
Verze (číslo verze struktury)	MQCTLO_VERSION_1	1
Volby (volby)	MQCTLO_NONE	Hodnoty null
Volby (vyhrazené pole)	Vyhrazené pole	
ConnectionArea (pole pro použití funkce zpětného volání)	Není	Ukazatel Null nebo bajty s hodnotou Null

Notes:

1. V programovacím jazyku C se jedná o proměnnou makra.MQCTLO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCTLO MyCTLO = {MQCTLO_DEFAULT};
```

Deklarace jazyka

Deklarace C pro MQCTLO

```
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 */
 };
```

Deklarace jazyka COBOL pro objekt MQCTLO

```
** 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
```

Deklarace PL/I pro MQCTLO

```
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 */
```

StrucId (MQCHAR4) pro MQCTLO

Toto je identifikátor struktury struktury voleb řízení. Vždy se jedná o vstupní pole. Jeho hodnota je MQCTLO_STRUC_ID.

Hodnota musí být:

MQCTLO_STRUC_ID
Identifikátor pro strukturu voleb řízení.

Pro programovací jazyk C je definována také konstanta MQCTLO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQCTLO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQCTLO

Struktura voleb řízení-pole Verze

Toto je číslo verze struktury; hodnota musí být:

MQCTLO_VERSION_1
Version-1 Struktura voleb řízení.

Následující konstanta určuje číslo verze aktuální verze:

MQCTLO_CURRENT_VERSION
Aktuální verze struktury voleb řízení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCTLO_VERSION_1.

Volby (MQLONG) pro MQCTLO

Struktura voleb řízení-pole Volby

Volby, které řídí akci MQCTL.

MQCTLO_FAIL_IF_QUIESCING
Vynutí selhání volání MQCTL, pokud je správce front nebo připojení ve stavu uvedení do klidového stavu.

Zadejte volbu MQGMO_FAIL_IF_QUIESCING ve volbách MQGMO předaných pro volání MQCB, abyste způsobili oznámení spotřebitelům zpráv při jejich uvedení do klidového stavu.

MQCTLO_THREAD_AFFINITY
Tato volba informuje systém, že aplikace vyžaduje, aby všichni spotřebitelé zpráv pro stejné připojení byli voláni ve stejném podprocesu. Tento podproces bude použit pro všechna vyvolání spotřebitelů, dokud nebude připojení zastaveno.

Výchozí volba: Pokud nepotřebujete žádnou z popsaných voleb, použijte následující volbu:

MQCTLO_NONE
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty. MQCTLO_NONE je definován jako pomůcka pro programovou dokumentaci; není zamýšleno, aby tato volba byla použita s jinými, ale protože je její hodnota nula, nelze takové použití zjistit.

Toto je vstupní pole. Počáteční hodnota pole *Options* je MQCTLO_NONE.

Vyhrazeno (MQLONG) pro MQCTLO

Toto je vyhrazené pole. Hodnota musí být nula.

ConnectionArea (MQPTR) pro MQCTLO

Struktura voleb řízení-pole ConnectionArea

Jedná se o pole, které je k dispozici pro použití funkcí zpětného volání.

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a předává se beze změny do pole ConnectionArea ve struktuře MQCBC, což je vstupní parametr pro zpětné volání.

Toto pole je ignorováno pro všechny jiné operace než MQOP_START a MQOP_START_WAIT.

Toto je vstupní a výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

Záhlaví MQDH-Distribution

Struktura MQDH popisuje další data, která jsou přítomna ve zprávě v případě, že se jedná o zprávu distribučního seznamu uloženou v přenosové frontě. Zpráva distribučního seznamu je zpráva, která je odeslána do více cílových front. Další data se skládají ze struktury MQDH následované polem záznamů MQOR a polem záznamů MQPMR.Tuto strukturu používají specializované aplikace, které vkládají zprávy přímo do přenosových front nebo odebírají zprávy z přenosových front (například agenti kanálů zpráv). Aplikace, které chtějí vkládat zprávy do distribučních seznamů, nesmí tuto strukturu používat. Místo toho musí použít strukturu MQOD k definování cílů v distribučním seznamu a strukturu MQPMO k určení vlastností zprávy nebo k přijetí informací o zprávách odesílaných do jednotlivých míst určení.

Dostupnost

Struktura MQDH je k dispozici na následujících platformách:

- AIX
- IBM i
- Linux

- Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Název formátu

MQFMT_DIST_HEADER

Znaková sada a kódování

Data v MQDH musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE.

Nastavte znakovou sadu a kódování MQDH do polí *CodedCharSetId* a *Encoding* v:

- MQMD (pokud je struktura MQDH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQDH (všechny ostatní případy).

Pouľití

Když aplikace vloží zprávu do distribučního seznamu a některá nebo všechna místa určení jsou vzdálená, správce front před data zprávy aplikace vloží struktury MQXQH a MQDH a umístí zprávu do příslušné přenosové fronty. Data se proto vyskytují v následující posloupnosti, když je zpráva v přenosové frontě:

- Struktura MQXQH
- Struktura MQDH plus pole záznamů MQOR a MQPMR
- Data zprávy aplikace

V závislosti na cílech může správce front generovat více než jednu takovou zprávu a umístit ji do různých přenosových front. V tomto případě struktury MQDH v těchto zprávách identifikují různé podmnožiny cílů definovaných v distribučním seznamu otevřeném aplikací.

Aplikace, která vkládá zprávu distribučního seznamu přímo do přenosové fronty, musí odpovídat dříve popsané posloupnosti a musí zajistit správnost struktury MQDH. Pokud je struktura MQDH neplatná, správce front může selhat při volání MQPUT nebo MQPUT1 s kódem příčiny MQRC_DH_ERROR.

Zprávy ve frontě můžete ukládat ve formě rozdělovníku pouze v případě, že jste definovali frontu jako schopnou podporovat zprávy rozdělovníku. Viz atribut fronty **DistLists** popsaný v části “Atributy pro fronty” na stránce 836. Pokud aplikace vloží zprávu distribučního seznamu přímo do fronty, která nepodporuje distribuční seznamy, správce front rozdělí zprávu distribučního seznamu na jednotlivé zprávy a umístí je do fronty.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 484. Pole v MQDH pro MQDH

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQDH_STRUC_ID	'DH¬¬'
Verze (číslo verze struktury)	MQDH_VERSION_1	1
StrucLength (délka struktury MQDH plus následující záznamy)	Není	0
Kódování (číselné kódování dat, která následují za polem záznamů MQPMR)	Není	0

Tabulka 484. Pole v MQDH pro MQDH (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
CodedCharSetId (identifikátor znakové sady dat, která následují za polem záznamů MQPMR)	MQCCSI_UNDEFINED	0
Formát (název formátu dat, která následují za polem záznamů MQPMR)	MQFMT_NONE	Mezery
Příznaky (obecné příznaky)	MQDHF_NONE	0
PutMsgRecFields (příznaky označující, která pole MQPMR jsou přítomna)	MQPMRF_NONE	0
RecsPresent (počet přítomných záznamů objektů)	Není	0
ObjectRecOffset (posun prvního záznamu objektu od začátku MQDH)	Není	0
PutMsgRecOffset (posun prvního záznamu vkládané zprávy od začátku MQDH)	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQDH_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQDH MyDH = {MQDH_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQDH

```
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 */
};
```

Deklarace jazyka COBOL pro MQDH

```
**   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.
```

Deklarace PL/I pro MQDH

```
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 */
```

Vizuální základní deklarace pro MQDH

```
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
```

StrucId (MQCHAR4) pro MQDH

Jedná se o identifikátor struktury struktury záhlaví distribuce. Vždy se jedná o vstupní pole. Jeho hodnota je MQDH_STRUC_ID.

Hodnota musí být:

MQDH_STRUC_ID
Identifikátor pro strukturu záhlaví distribuce.

Pro programovací jazyk C je definována také konstanta MQDH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQDH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQDH

Hodnota musí být:

MQDH_VERSION_1
Číslo verze pro strukturu záhlaví distribuce.

Následující konstanta určuje číslo verze aktuální verze:

MQDH_CURRENT_VERSION
Aktuální verze struktury záhlaví distribuce.

Počáteční hodnota tohoto pole je MQDH_VERSION_1.

StrucLength (MQLONG) pro MQDH

Jedná se o počet bajtů od začátku struktury MQDH po začátek dat zprávy následujících po polích záznamů MQOR a MQPMR. Data se vyskytují v následujícím pořadí:

- Struktura MQDH
- Pole záznamů MQOR
- Pole záznamů MQPMR
- Data zprávy

Pole záznamů MQOR a MQPMR jsou adresována offsety obsaženými ve struktuře MQDH. Pokud tyto odchylky vedou k nepoužitým bajtům mezi jednou nebo více strukturami MQDH, poli záznamů a daty zpráv, musí být tyto nepoužívané bajty zahrnuty do hodnoty *StrucLength*, ale obsah těchto bajtů není správcem front zachován. Pole záznamů MQPMR je platné před polem záznamů MQOR.

Počáteční hodnota tohoto pole je 0.

Kódování (MQLONG) pro MQDH

Toto je číselné kódování dat, která následují za poli záznamů MQOR a MQPMR; neplatí pro číselná data v samotné struktuře MQDH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

CodedCharSetId (MQLONG) pro MQDH

Jedná se o identifikátor znakové sady dat, která následují za poli záznamů MQOR a MQPMR; nevztahuje se na znaková data v samotné struktuře MQDH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Můžete použít následující speciální hodnotu:

MQCCSI_INHERIT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Pokud nedojde k žádné chybě, volání MQGET nevrátí hodnotu MQCCSI_INHERIT.

Nemůžete použít MQCCSI_INHERIT, pokud hodnota pole *PutApplType* v MQMD je MQAT_BROKER.

Tato hodnota je podporována v následujících prostředích:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

  a pro klienty IBM MQ připojené k těmto systémům.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8) pro MQDH

Jedná se o název formátu dat, která následují za poli záznamů MQOD a MQPMR (podle toho, co nastane později).

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *Format* v MQMD.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQDH

Můžete zadat následující příznak:

MQDHF_NEW_MSG_IDS
Vygenerujte nový identifikátor zprávy pro každé místo určení v rozdělovníku. Nastavte tuto volbu pouze v případě, že nejsou přítomny žádné záznamy vložených zpráv, nebo pokud jsou tyto záznamy přítomny, ale neobsahují pole *MsgId* .

Použití tohoto příznaku odkládá generování identifikátorů zpráv až do okamžiku, kdy je zpráva rozdělovníku nakonec rozdělena na jednotlivé zprávy. Tím se minimalizuje množství řídicích informací, které musí proudit se zprávou distribučního seznamu.

Když aplikace vloží zprávu do distribučního seznamu, správce front nastaví MQDHF_NEW_MSG_IDS v MQDH, který vygeneruje, když jsou splněny oba následující příkazy:

- Aplikace neposkytla žádné záznamy vložených zpráv, nebo poskytnuté záznamy neobsahují pole *MsgId* .
- Pole *MsgId* v MQMD je MQMI_NONE nebo pole *Options* v MQPMO obsahuje MQPMO_NEW_MSG_ID

Pokud nejsou potřeba žádné příznaky, zadejte následující:

MQDHF_NONE
Nebyly zadány žádné příznaky. MQDHF_NONE je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby tato konstanta byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Počáteční hodnota tohoto pole je MQDHF_NONE.

PutMsgRecFields (MQLONG) pro MQDH

Můžete uvést žádný nebo více následujících příznaků:

MQPMRF_MSG_ID
Pole identifikátoru zprávy je přítomno.

MQPMRF_CORREL_ID
Pole identifikátoru korelace je přítomno.

MQPMRF_GROUP_ID
Pole identifikátoru skupiny je přítomno.

MQPMRF_ZPĚTNÁ vazba
Pole zpětné vazby je přítomno.

MQPMRF_ACCOUNTING_TOKEN
Pole tokenu evidence je přítomno.

Pokud nejsou přítomna žádná pole MQPMR, uveďte následující:

MQPMRF_NONE
Nejsou přítomna žádná pole záznamu vložená zpráva. MQPMRF_NONE je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby tato konstanta byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Počáteční hodnota tohoto pole je MQPMRF_NONE.

RecsPresent (MQLONG) pro MQDH

Jedná se o počet míst určení. Rozdělovník musí vždy obsahovat alespoň jeden cíl, takže *RecsPresent* musí být vždy větší než nula.

Počáteční hodnota tohoto pole je 0.

ObjectRecOffset (MQLONG) pro MQDH

Tím se získá posun v bajtech prvního záznamu v poli záznamů objektů MQOR, který obsahuje názvy cílových front. V tomto poli jsou záznamy *RecsPresent* . Tyto záznamy (plus všechny bajty přeskočené mezi prvním záznamem objektu a předchozím polem) jsou zahrnuty do délky dané polem *StrucLength* .

Rozdělovník musí vždy obsahovat alespoň jeden cíl, takže *ObjectRecOffset* musí být vždy větší než nula.

Počáteční hodnota tohoto pole je 0.

PutMsgRecOffset (MQLONG) pro MQDH

Tím se získá posun v bajtech prvního záznamu v poli záznamů vložených zpráv MQPMR, které obsahují vlastnosti zprávy. Je-li přítomen, v tomto poli jsou záznamy *RecsPresent* . Tyto záznamy (plus všechny bajty přeskočené mezi prvním záznamem vkládané zprávy a předchozím polem) jsou zahrnuty do délky dané polem *StrucLength* .

Záznamy vložených zpráv jsou volitelné; pokud nejsou poskytnuty žádné záznamy, *PutMsgRecOffset* je nula a *PutMsgRecFields* má hodnotu MQPMRF_NONE.

Počáteční hodnota tohoto pole je 0.

MQDLH-Záhlaví nedoručených zpráv

Struktura MQDLH popisuje informace, které jsou předponou dat zprávy aplikace pro zprávy ve frontě nedoručených zpráv (nedoručených zpráv). Zpráva může být doručena do fronty nedoručených zpráv buď proto, že ji správce front nebo agent kanálu zpráv přesměrovali do fronty, nebo proto, že aplikace vložila zprávu přímo do fronty.

Název formátu

MQFMT_DEAD_LETTER_HEADER

Znaková sada a kódování

Pole ve struktuře MQDLH jsou ve znakové sadě a kódování dané poli *CodedCharSetId* a *Encoding* . Ty jsou určeny ve struktuře záhlaví, která předchází MQDLH, nebo ve struktuře MQMD, pokud je MQDLH na začátku dat zprávy aplikace.

Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech front.

Používáte-li třídy IBM MQ pro Java/JMSa kódová stránka definovaná v deskriptoru MQMD není podporována virtuálním počítačem Java , je MQDLH zapsán ve znakové sadě UTF-8 .

Použití

Aplikace, které vkládají zprávy přímo do fronty nedoručených zpráv, musí před data zprávy vložit strukturu MQDLH a inicializovat pole s odpovídajícími hodnotami. Správce front však nevyžaduje přítomnost struktury MQDLH nebo zadání platných hodnot pro pole.

Pokud je zpráva příliš dlouhá pro vložení do fronty nedoručených zpráv, aplikace musí provést jednu z následujících možností:

- Oříznětе data zprávy tak, aby se vešla do fronty nedoručených zpráv.
- Zaznamenejte zprávu do pomocné paměti a umístěte zprávu o výjimce do fronty nedoručených zpráv, která to označuje.
- Zahoďte zprávu a vraťte chybu jejímu původci. Jedná-li se (nebo může-li být) o kritickou zprávu, proveďte to pouze v případě, že je známo, že původce stále má kopii zprávy; například zprávu přijatou agentem kanálu zpráv z komunikačního kanálu.

Která z předchozích akcí je vhodná (pokud existuje), závisí na návrhu aplikace.

Správce front provádí speciální zpracování, když je zpráva, která je segmentem, vložena se strukturou MQDLH do popředí; další podrobnosti viz popis struktury MQMDE.

Vkládání zpráv do fronty nedoručených zpráv

Při vložení zprávy do fronty nedoručených zpráv musí být struktura MQMD použitá pro volání MQPUT nebo MQPUT1 identická s MQMD přidruženým ke zprávě (obvykle MQMD vrácené voláním MQGET), s výjimkou následujícího:

- Nastavte pole *CodedCharSetId* a *Encoding* na jakoukoli znakovou sadu a kódování použité pro pole ve struktuře MQDLH.
- Nastavte pole *Format* na hodnotu MQFMT_DEAD_LETTER_HEADER, abyste označili, že data začínají strukturou MQDLH.
- Nastavte kontextová pole (*AccountingToken*, *ApplIdentityData*, *ApplOriginData*, *PutApplName*, *PutApplType*, *PutDate*, *PutTime*, *UserIdentifier*) pomocí kontextové volby odpovídající okolnostem:
  - Aplikace vkládající do fronty nedoručených zpráv zprávu, která nesouvisí s žádnou předchozí zprávou, musí použít volbu MQPMO_DEFAULT_CONTEXT; to způsobí, že správce front nastaví všechna pole kontextu v deskriptoru zprávy na výchozí hodnoty.
  - Serverová aplikace, která vkládá do fronty nedoručených zpráv zprávu, kterou právě přijala, musí použít volbu MQPMO_PASS_ALL_CONTEXT, aby zachovala původní informace o kontextu.
  - Serverová aplikace vkládající do fronty nedoručených zpráv *odpověď* na zprávu, kterou právě přijala, musí použít volbu MQPMO_PASS_IDENTITY_CONTEXT; tím se zachová informace o identitě, ale nastaví informace o původu na informace o aplikaci serveru.
  - Agent kanálu zpráv, který vkládá do fronty nedoručených zpráv zprávu, kterou přijal od svého komunikačního kanálu, musí použít volbu MQPMO_SET_ALL_CONTEXT k zachování původních informací o kontextu.

V samotné struktuře MQDLH nastavte pole takto:

- Nastavte pole *CodedCharSetId*, *Encoding*a *Format* na hodnoty, které popisují data následující za strukturou MQDLH, obvykle hodnoty z původního deskriptoru zprávy.
- Nastavte pole kontextu *PutApplType*, *PutApplName*, *PutDate*a *PutTime* na hodnoty odpovídající aplikaci, která vkládá zprávu do fronty nedoručených zpráv; tyto hodnoty nesouvisejí s původní zprávou.
- Podle potřeby nastavte další pole.

Ujistěte se, že všechna pole mají platné hodnoty a že znaková pole jsou vyplněna mezerami na definovanou délku pole; neukončujte znaková data předčasně pomocí znaku null, protože správce front nepřevádí znaky null a následné znaky na mezery ve struktuře MQDLH.

Získávání zpráv z fronty nedoručených zpráv

Aplikace, které získají zprávy z fronty nedoručených zpráv, musí ověřit, že zprávy začínají strukturou MQDLH. Aplikace může určit, zda je přítomna struktura MQDLH, kontrolou pole *Format* v deskriptoru zprávy MQMD; pokud má pole hodnotu MQFMT_DEAD_LETTER_HEADER, data zprávy začínají strukturou MQDLH. Uvědomte si také, že zprávy, které aplikace získají z fronty nedoručených zpráv, mohou být zkráceny, pokud byly původně příliš dlouhé pro frontu.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 485. Pole v MQDLH pro MQDLH

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	ID_STRUC_MQDLH_STRUC_ID	'DLH¬'
Verze (číslo verze struktury)	MQDLH_VERSION_1	1
Příčina (zpráva příčiny dorazila do fronty nedoručených zpráv)	MQRC_NONE	0
DestQName (název původní cílové fronty)	Není	Prázdný řetězec nebo mezery
DestQMgrNázev (název původního správce cílových front)	Není	Prázdný řetězec nebo mezery
Kódování (číselné kódování dat, která následují za MQDLH)	Není	0
CodedCharSetId (identifikátor znakové sady dat, která následují za MQDLH)	MQCCSI_UNDEFINED	0
Formát (název formátu dat, která následují za MQDLH)	MQFMT_NONE	Mezery
PutApplTyp (typ aplikace, která vložila zprávu do fronty nedoručených zpráv)	Není	0
PutApplNázev (název aplikace, která vložila zprávu do fronty nedoručených zpráv)	Není	Prázdný řetězec nebo mezery
PutDate (datum, kdy byla zpráva vložena do fronty nedoručených zpráv)	Není	Prázdný řetězec nebo mezery
PutTime (čas, kdy byla zpráva vložena do fronty nedoručených zpráv)	Není	Prázdný řetězec nebo mezery

Tabulka 485. Pole v MQDLH pro MQDLH (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQDLH_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQDLH MyDLH = {MQDLH_DEFAULT};
```

Deklarace jazyka

Deklarace C pro MQDLH

```
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 */
};
```

Deklarace jazyka COBOL pro MQDLH

```
**   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).
```

Deklarace PL/I pro MQDLH

```
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 */
```

Deklarace High Level Assembler pro MQDLH

```
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)
```

Deklarace jazyka Visual Basic pro MQDLH

```
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
```

StrucId (MQCHAR4) pro MQDLH

Toto je identifikátor struktury struktury záhlaví nedoručeného dopisu. Vždy se jedná o vstupní pole. Jeho hodnota je MQDLH_STRUC_ID.

Hodnota musí být:

ID_STRUC_MQDLH_STRUC_ID
Identifikátor pro strukturu záhlaví nedoručného dopisu.

Pro programovací jazyk C je definována také konstanta MQDLH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQDLH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQDLH

Verze je číslo verze struktury.

Hodnota musí být:

MQDLH_VERSION_1
Číslo verze pro strukturu záhlaví nedoručených dopisů.

Následující konstanta určuje číslo verze aktuální verze:

MQDLH_CURRENT_VERSION
Aktuální verze struktury záhlaví nedoručených dopisů.

Počáteční hodnota tohoto pole je MQDLH_VERSION_1.

Příčina (MQLONG) pro MQDLH

V poli Příčina je uvedena příčina, proč byla zpráva umístěna do fronty nedoručených zpráv, a nikoli do původní cílové fronty.

To identifikuje příčinu, proč byla zpráva umístěna do fronty nedoručených zpráv místo do původní cílové fronty. Měla by být jednou z hodnot MQFB_ * nebo MQRC_ * (například MQRC_Q_FULL). Podrobnosti o běžných hodnotách MQFB_ *, které se mohou vyskytnout, naleznete v popisu pole *Feedback* v souboru “MQMD-Deskriptor zpráv” na stránce 424 .

Pokud je hodnota v rozsahu MQFB_IMS_FIRST až MQFB_IMS_LAST, lze skutečný kód chyby IMS určit odečtením hodnoty MQFB_IMS_ERROR od hodnoty pole *Reason* .

Některé hodnoty MQFB_ * se vyskytují pouze v tomto poli. Týkají se zpráv úložiště, zpráv spouštěče nebo zpráv přenosové fronty, které byly přeneseny do fronty nedoručených zpráv. Patří mezi ně:

MQFB_APPL_CANNOT_BE_STARTED (X'00000109')
Aplikace, která zpracovává zprávu spouštěče, nemůže spustit aplikaci uvedenou v poli *ApplId* zprávy spouštěče (viz “MQTM-zpráva spouštěče” na stránce 601).

V systému z/OSje transakce CKTI CICS příkladem aplikace, která zpracovává zprávy spouštěče.

MQFB_APPL_TYPE_ERROR (X'0000010B')
Aplikace, která zpracovává zprávu spouštěče, nemůže spustit aplikaci, protože pole *ApplType* zprávy spouštěče není platné (viz “MQTM-zpráva spouštěče” na stránce 601).

V systému z/OSje transakce CKTI CICS příkladem aplikace, která zpracovává zprávy spouštěče.

MQFB_BIND_OPEN_CLUSRCVR_DEL (X'00000119')
Zpráva byla na SYSTEM.CLUSTER.TRANSMIT.QUEUE je určena pro frontu klastru, která byla otevřena s volbou MQOO_BIND_ON_OPEN, ale kanál příjemce vzdáleného klastru, který má být použit pro přenos zprávy do cílové fronty, byl před odesláním zprávy odstraněn. Vzhledem k tomu, že byl zadán parametr MQOO_BIND_ON_OPEN, lze k přenosu zprávy použít pouze kanál vybraný při otevření fronty. Vzhledem k tomu, že tento kanál již není k dispozici, je zpráva umístěna do fronty nedoručených zpráv.

MQFB_NOT_A_REPOSITORY_MSG (X'00000118')
Zpráva není zprávou úložiště.

MQFB_STOPPED_BY_CHAD_EXIT (X'00000115')
Zpráva byla zastavena uživatelskou procedurou automatické definice kanálu.

MQFB_STOPPED_BY_MSG_EXIT (X'0000010D')
Zpráva byla zastavena uživatelskou procedurou pro zprávy kanálu.

MQFB_TM_ERROR (X'0000010A')
Pole *Format* v MQMD určuje MQFMT_TRIGGER, ale zpráva nezačíná platnou strukturou MQTM. Například mnemonický oko-catcher *StrucId* nemusí být platný, *Version* nemusí být rozpoznán nebo délka zprávy spouštěče může být nedostatečná pro to, aby obsahovala strukturu MQTM.

V systému z/OSje transakce CKTI CICS příkladem aplikace, která zpracovává zprávy spouštěče a může generovat tento kód zpětné vazby.

MQFB_XMIT_Q_MSG_ERROR (X'0000010F')
Agent kanálu zpráv zjistil, že zpráva v přenosové frontě není ve správném formátu. Agent kanálu zpráv vloží zprávu do fronty nedoručených zpráv pomocí tohoto kódu zpětné vazby.

Jednou z běžných příčin je, že zpráva byla vložena přímo do přenosové fronty, takže zpráva nemá očekávané záhlaví XQH. Zprávy by měly být vloženy do přenosové fronty prostřednictvím vzdálené fronty, pokud aplikace nesestaví záhlaví MQXQH.

Počáteční hodnota tohoto pole je MQRC_NONE.

DestQName (MQCHAR48) pro MQDLH
DestQName je název fronty zpráv, která byla původním cílem zprávy.

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

DestQMgrNázev (MQCHAR48) pro MQDLH
DestQMgrNázev je název správce front, který byl původním cílem zprávy.

Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

Kódování (MQLONG) pro MQDLH
Kódování je číselné kódování dat, která se řídí strukturou MQDLH (obvykle data z původní zprávy); nevztahuje se na číselná data v samotné struktuře MQDLH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

CodedCharSetId (MQLONG) pro MQDLH
CodedCharSetId je identifikátor znakové sady dat, která procházejí strukturou MQDLH (obvykle data z původní zprávy); neplatí pro znaková data v samotné struktuře MQDLH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

MQCCSI_INHERIT
Znaková data v datech následujících za touto strukturou jsou ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota MQCCSI_INHERIT vrácena voláním MQGET.

Nemůžete použít MQCCSI_INHERIT, pokud hodnota pole *PutApplType* v MQMD je MQAT_BROKER.

Tato hodnota je podporována v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

a pro klienty IBM MQ připojené k těmto systémům.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8) pro MQDLH
Formát je název formátu dat, která se řídí strukturou MQDLH (obvykle data z původní zprávy).

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro kódování pole *Format* v deskriptoru MQMD.

Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

PutApplTyp (MQLONG) pro MQDLH
PutApplTyp je typ aplikace, která vložila zprávu do fronty nedoručených zpráv (nedoručených zpráv).

Toto pole má stejný význam jako pole *PutApplType* v deskriptoru zprávy MQMD (podrobnosti viz “MQMD-Deskriptor zpráv” na stránce 424).

Pokud správce front přesměruje zprávu do fronty nedoručených zpráv, má parametr *PutApplType* hodnotu MQAT_QMGR.

Počáteční hodnota tohoto pole je 0.

PutApplNázev (MQCHAR28) pro MQDLH
PutApplNázev je název aplikace, která vložila zprávu do fronty nedoručených zpráv.

Formát názvu závisí na poli *PutApplType* . Formát může měnit vydání od vydání. Viz popis pole *PutApplName* v souboru “MQMD-Deskriptor zpráv” na stránce 424.

Pokud správce front přesměruje zprávu do fronty nedoručených zpráv, obsahuje produkt *PutApplName* prvních 28 znaků názvu správce front, které jsou v případě potřeby doplněny mezerami.

Délka tohoto pole je dána hodnotou MQ_PUT_APPL_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 28 prázdných znaků v jiných programovacích jazycích.

PutDate (MQCHAR8) pro MQDLH

PutDate je datum, kdy byla zpráva vložena do fronty nedoručených zpráv (nedoručených zpráv).

Formát použitý pro datum, kdy je toto pole generováno správcem front, je:

- RRRRMMDD

kde znaky představují:

yyyy
rok (čtyři číslice)

MM
měsíc v roce (01 až 12)

DD
den v měsíci (01 až 31)

Greenwichský střední čas (GMT) se používá pro pole *PutDate* a *PutTime* pod podmínkou, že systémové hodiny jsou přesně nastaveny na GMT.

Délka tohoto pole je dána hodnotou MQ_PUT_DATE_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a osm prázdných znaků v jiných programovacích jazycích.

PutTime (MQCHAR8) pro MQDLH

PutTime je čas, kdy byla zpráva vložena do fronty nedoručených zpráv (nedoručených zpráv).

Formát použitý pro čas, kdy je toto pole generováno správcem front, je:

- HHMMSSTH

kde znaky představují:

HH
hodiny (00 až 23)

MM
minut (00 až 59)

SS
sekundy (00 až 59; viz poznámka)

T
desetiny sekundy (0 až 9)

H
setiny sekundy (0 až 9)

Poznámka: Pokud jsou systémové hodiny synchronizovány s velmi přesným časovým standardem, je možné ve vzácných případech vrátit 60 nebo 61 sekund v souboru *PutTime*. K tomu dochází, když jsou do globálního časového standardu vloženy přestupné sekundy.

Greenwichský střední čas (GMT) se používá pro pole *PutDate* a *PutTime* pod podmínkou, že systémové hodiny jsou přesně nastaveny na GMT.

Délka tohoto pole je dána hodnotou MQ_PUT_TIME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a osm prázdných znaků v jiných programovacích jazycích.

MQDMHO-Volby odstranění popisovače zprávy

Struktura **MQDMHO** umožňuje aplikacím určit volby, které řídí způsob odstranění manipulátorů zpráv. Struktura je vstupní parametr volání **MQDLTMH** .

Znaková sada a kódování

Data v souboru **MQDMHO** musí být ve znakové sadě aplikace a kódování aplikace (**MQENC_NATIVE**).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 486. Pole v MQDMHO

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQDMHO_STRUC_ID	'DMHO'
Verze (číslo verze struktury)	MQDMHO_VERSION_1	1
Volby (volby)	MQDMHO_NONE	0

Notes:

1. V programovacím jazyku C se jedná o proměnnou makra.MQDMHO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQDMHO MyDMHO = {MQDMHO_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQDMHO

```
typedef struct tagMQDMHO;
struct tagMQDMHO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQDLTMH */
};
```

Deklarace jazyka COBOL pro objekt MQDMHO

```
**   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.
```

Deklarace PL/I pro MQDMHO

```
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 */
```

Deklarace High Level Assembler pro MQDMHO

```
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)
```

StrucId (MQCHAR4) pro MQDMHO

Jedná se o identifikátor struktury struktury voleb popisovače zprávy odstranění. Vždy se jedná o vstupní pole. Jeho hodnota je MQDMHO_STRUC_ID.

Hodnota musí být:

MQDMHO_STRUC_ID

Identifikátor pro strukturu voleb popisovače zprávy odstranění.

Pro programovací jazyk C je definována také konstanta **MQDMHO_STRUC_ID_ARRAY** . Má stejnou hodnotu jako **MQDMHO_STRUC_ID**, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQDMHO

Toto je číslo verze struktury; hodnota musí být:

MQDMHO_VERSION_1
Version-1 odstraňte strukturu voleb popisovače zprávy.

Následující konstanta určuje číslo verze aktuální verze:

MQDMHO_CURRENT_VERSION
Aktuální verze struktury voleb popisovače zprávy odstranění.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQDMHO_VERSION_1**.

Volby (MQLONG) pro MQDMHO

Hodnota musí být:

MQDMHO_NONE

Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQDMHO_NONE**.

MQDMPO-Volby vlastnosti odstranění zprávy

Struktura MQDMPO umožňuje aplikacím určit volby, které řídí způsob odstraňování vlastností zpráv. Struktura je vstupní parametr volání MQDLTMP.

Znaková sada a kódování

Data v MQDMPO musí být ve znakové sadě aplikace a kódování aplikace (MQENC_NATIVE).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 487. Pole v MQDPMO		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	ID_STRUC_MQDMPO_ID	'DMPO'
Verze (číslo verze struktury)	MQDMPO_VERSION_1	1
Volby (volby řídící akci MQDMPO)	Volby, které řídí akci MQDLTMP	MQDMPO_NONE

Notes:

1. V programovacím jazyku C se jedná o proměnnou makra.MQDMPO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQDMPO MyDMPO = {MQDMPO_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQDMPO

```
typedef struct tagMQDMPO MQDMPO;
struct tagMQDMPO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQDLTMP */
};
```

Deklarace jazyka COBOL pro MQDMPO

```
**   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.
```

Prohlášení PL/I pro MQDMPO

```
Dcl
 1 MQDMPO 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 */
```

Deklarace High Level Assembler pro MQDMPO

```
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)
```

StrucId (MQCHAR4) pro MQDMPO
Jedná se o identifikátor struktury struktury voleb vlastnosti odstranění zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQDMPO_STRUC_ID.

Hodnota musí být:

ID_STRUC_MQDMPO_ID
Identifikátor pro strukturu voleb vlastností zprávy odstranění.

Pro programovací jazyk C je definována také konstanta MQDMPO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQDMPO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQDMPO
Struktura voleb vlastností zprávy odstranění-pole Verze

Toto je číslo verze struktury. Hodnota musí být:

MQDMPO_VERSION_1
Číslo verze pro strukturu voleb vlastností zprávy odstranění.

Následující konstanta určuje číslo verze aktuální verze:

MQDMPO_CURRENT_VERSION
Aktuální verze struktury voleb vlastností zprávy odstranění.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQDMPO_VERSION_1.

Volby (MQLONG) pro MQDMPO
Struktura voleb vlastností zprávy odstranění-pole Volby

Volby umístění: Následující volby se vztahují k relativnímu umístění vlastnosti v porovnání s kurzorem vlastnosti.

MQDMPO_DEL_FIRST
Odstraní první vlastnost, která odpovídá zadanému názvu.

MQDMPO_DEL_PROP_UNDER_CURSOR
Odstraní vlastnost, na kterou ukazuje kurzor vlastnosti; to je vlastnost, která byla naposledy zjišťována pomocí volby MQIMPO_INQ_FIRST nebo MQIMPO_INQ_NEXT.

Kurzor vlastnosti se resetuje při opětovném použití popisovače zprávy. Resetuje se také v případě, že je v poli *MsgHandle* ve struktuře MQGMO ve volání MQGET nebo ve struktuře MQPMO ve volání MQPUT určen manipulátor zprávy.

Pokud je tato volba použita v případě, že kurzor vlastnosti dosud nebyl vytvořen, volání se nezdaří s kódem dokončení MQCC_FAILED a důvodem MQRC_PROPERTY_NOT_AVAILABLE. Pokud již byla vlastnost, na kterou ukazuje kurzor vlastnosti, odstraněna, volání také selže s kódem dokončení MQCC_FAILED a důvodem MQRC_PROPERTY_NOT_AVAILABLE.

Pokud není požadována žádná z voleb thees, lze použít následující volbu:

MQDMPO_NONE
Nejsou uvedeny žádné volby.

Toto pole je vždy vstupní pole. Počáteční hodnota tohoto pole je MQDMPO_DEL_FIRST.

MQEPH-Vložené záhlaví PCF

Struktura MQEPH popisuje další data, která jsou přítomna ve zprávě, když je tato zpráva ve formátu programovatelného příkazu (PCF). Pole *PCFHeader* definuje parametry PCF, které se řídí touto strukturou, a to vám umožňuje sledovat data zprávy PCF s jinými záhlavími.

Název formátu

MQFMT_EMBEDDED_PCF

Znaková sada a kódování

Data v MQEPH musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE.

Nastavte znakovou sadu a kódování MQEPH do polí *CodedCharSetId* a *Encoding* v deskriptoru MQMD (pokud je struktura MQEPH na začátku dat zprávy) nebo do struktury záhlaví, která předchází struktuře MQEPH (všechny ostatní případy).

Použití

Struktury MQEPH nelze použít k odesílání příkazů příkazovému serveru nebo jinému serveru akceptujícího PCF správce front.

Podobně příkazový server ani jiný správce front PCF-akceptující server negenerují odpovědi ani události obsahující struktury MQEPH.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 488. Pole v MQEPH pro MQEPH

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQEPH_STRUC_ID	'EPH¬'
Verze (číslo verze struktury)	MQEPH_VERSION_1	1
StrucLength (délka struktury MQEPH plus struktury MQCFH a parametrů, které ji následují)	MQEPH_STRUC_LENGTH_FIXED	68
Kódování (číselné kódování dat, která následují za poslední strukturou parametrů PCF)	Není	0
CodedCharSetId (identifikátor znakové sady dat, která následují za poslední strukturou parametrů PCF)	MQCCSI_UNDEFINED	0
Formát (název formátu dat, která následují za poslední strukturou parametrů PCF)	MQFMT_NONE	Mezery
Příznaky (příznaky)	MQEPH_NONE	0
PCFHeader (záhlaví PCF (programovatelný formát příkazu))	Názvy a hodnoty definované v souboru Tabulka 489 na stránce 371	0

Tabulka 488. Pole v MQEPH pro MQEPH (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQEPH_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQEPH MyEPH = {MQEPH_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQEPH

```
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 */
 };
```

Deklarace jazyka COBOL pro MQEPH

```
**   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.
```

Deklarace PL/I pro MQEPH

```
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 */
```

Deklarace High Level Assembler pro MQEPH

```
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)
```

Vizuální základní deklarace pro MQEPH

```
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
```

StrucId (MQCHAR4) pro MQEPH

Jedná se o identifikátor struktury vložené struktury záhlaví PCF. Vždy se jedná o vstupní pole. Jeho hodnota je MQEPH_STRUC_ID.

Hodnota musí být:

MQEPH_STRUC_ID
Identifikátor pro vloženou strukturu záhlaví PCF.

Pro programovací jazyk C je definována také konstanta MQEPH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQEPH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQEPH

Hodnota musí být:

MQEPH_VERSION_1
Číslo verze pro vloženou strukturu záhlaví PCF.

Následující konstanta určuje číslo verze aktuální verze:

MQCFH_VERSION_3
Aktuální verze vložené struktury záhlaví PCF.

Počáteční hodnota tohoto pole je MQEPH_VERSION_1.

StrucLength (MQLONG) pro MQEPH

Jedná se o množství dat před další strukturou záhlaví. Zahrnuje:

- Délka záhlaví MQEPH
- Délka všech parametrů PCF následujících za záhlavím
- Jakákoli prázdná výplň za těmito parametry

Hodnota StrucLength musí být násobkem 4.

Část struktury s pevnou délkou je definována parametrem MQEPH_STRUC_LENGTH_FIXED.

Počáteční hodnota tohoto pole je 68.

Kódování (MQLONG) pro MQEPH

Jedná se o číselné kódování dat, která se řídí strukturou MQEPH a přidruženými parametry PCF; nevztahuje se na znaková data v samotné struktuře MQEPH.

Počáteční hodnota tohoto pole je 0.

CodedCharSetId (MQLONG) pro MQEPH

Jedná se o identifikátor znakové sady dat, která se řídí strukturou MQEPH a přidruženými parametry PCF; nevztahuje se na znaková data v samotné struktuře MQEPH.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8) pro MQEPH

Jedná se o název formátu dat, která následují za strukturou MQEPH a přidruženými parametry PCF.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQEPH

K dispozici jsou tyto hodnoty:

MQEPH_NONE
Nebyly zadány žádné příznaky. MQEPH_NONE je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby tato konstanta byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

MQEPH_CCSID_EMBEDDED
Znaková sada parametrů obsahujících znaková data je v každé struktuře určena jednotlivě v poli CodedCharSetId . Znaková sada polí StrucId a Formát je definována polem CodedCharSetId ve struktuře záhlaví, která předchází struktuře MQEPH, nebo polem CodedCharSetId v deskriptoru MQMD, pokud je MQEPH na začátku zprávy.

Počáteční hodnota tohoto pole je MQEPH_NONE.

PCFHeader (MQCFH) pro MQEPH

Toto je záhlaví formátu programovatelného příkazu (PCF) definující parametry PCF, které se řídí strukturou MQEPH. To vám umožní sledovat data zprávy PCF s dalšími záhlavími.

Záhlaví PCF je na počátku definováno s následujícími hodnotami:

Tabulka 489. Pole v MQCFH

Název pole	**Název konstanty**	**Hodnota konstanty**
Type	MQCFT_NONE	0
StrucLength	MQCFH_STRUC_LENGTH	36
Version	MQCFH_VERSION_3	3
StrucLength	Není	0
Command	MQCMD_NONE	0
MsgSeqNumber	Není	1
Control	MQCFC_LAST	1
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
ParameterCount	Není	0

Aplikace musí změnit Type z MQCFT_NONE na platný typ struktury pro použití, které vytváří z vloženého záhlaví PCF.

MQGMO-Volby získání zprávy

Struktura MQGMO umožňuje aplikaci řídit způsob odebírání zpráv z front. Struktura je vstupní/výstupní parametr volání MQGET.

Verze

Aktuální verze MQGMO je MQGMO_VERSION_4. Určitá pole jsou k dispozici pouze v určitých verzích MQGMO. Potřebujete-li portovat aplikace mezi několika prostředími, musíte se ujistit, že verze produktu MQGMO je konzistentní ve všech prostředích. Pole, která existují pouze v konkrétních verzích struktury,

jsou jako taková identifikována v produktu “MQGMO-Volby získání zprávy” na stránce 371 a v popisech polí.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi MQGMO, která je podporována prostředím, ale s počáteční hodnotou pole *Version* nastavenou na MQGMO_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , nastavte pole *Version* na číslo verze požadované verze.

Znaková sada a kódování

Data v MQGMO musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 490. Pole v MQGMO pro MQGMO

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQGMO_STRUC_ID	'GMO¬'
Verze (číslo verze struktury)	MQGMO_VERSION_1	1
MQGMO-pole Volby (volby, které řídí akci MQGET)	MQGMO_NO_WAIT	0
WaitInterval (interval čekání)	Není	0
Signal1 (signál)	Není	Nulový ukazatel na z/OS ; 0 jinak
Signal2 (identifikátor signálu)	Není	0
ResolvedQName (přeložený název cílové fronty)	Není	Prázdný řetězec nebo mezery
Poznámka: Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než hodnota MQGMO_VERSION_2.		
MatchOptions (volby řídící kritéria výběru použitá pro MQGET)	MQMO_MATCH_MSG_ID + MQMO_MATCH_CORREL_ID	3
GroupStatus (příznak označující, zda je načtená zpráva ve skupině)	MQGS_NOT_IN_GROUP	'¬'
SegmentStatus (příznak označující, zda načtená zpráva je segmentem logické zprávy)	MQSS_NOT_A_SEGMENT	'¬'
Segmentace (příznak označující, zda je pro načtenou zprávu povolena další segmentace)	MQSEG_INHIBITED	'¬'
Reserved1 (vyhrazeno)	Není	'¬'
Poznámka: Zbývající pole jsou ignorována, pokud je *Version* menší než MQGMO_VERSION_3.		
MsgToken (token zprávy)	MQMTOK_NONE	Hodnoty null

Tabulka 490. Pole v MQGMO pro MQGMO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
ReturnedLength (délka vrácených dat zprávy v bajtech)	MQRL_UNDEFINED	-1
Poznámka: Zbývající pole se ignorují, pokud je *Version* menší než MQGMO_VERSION_4.		
Reserved2 (vyhrazeno)	Není	'¬'
MsgHandle (popisovač zprávy, který má být naplněn vlastnostmi zprávy načítané z fronty)	MQHM_NONE	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQGMO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQGMO MyGMO = {MQGMO_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQGMO

```
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 */
};
```

Poznámka: V systému z/OSje pole *Signal1* deklarováno jako PMQLONG.

Deklarace jazyka COBOL pro MQGMO

```
**   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.
```

Poznámka: V systému z/OSje pole *Signal1* deklarováno jako POINTER.

Prohlášení PL/I pro MQGMO

```
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 */
```

Poznámka: V systému z/OSje pole *Signal1* deklarováno jako pointer.

Deklarace High Level Assembler pro MQGMO

```
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)
```

Deklarace High Level Assembler pro MQGMO

```
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
```

PROPCTL volby kanálu pro MQGMO

Pomocí atributu kanálu **PROPCTL** můžete řídit, které vlastnosti zpráv budou zahrnuty do zprávy odeslané ze správce front systému IBM MQ 9.3 partnerskému správci front ze starší verze produktu IBM MQ.

Tabulka 491. Nastavení atributu vlastnosti zprávy kanálu

PROPCTL	Popis
all	Tuto volbu použijte v případě, že aplikace připojené ke správci front partnera z dřívější verze mohou zpracovat vlastnosti umístěné ve zprávě aplikací IBM MQ 9.3 . Všechny vlastnosti jsou odesílány do partnerského správce front kromě všech dvojic název-hodnota umístěných v souboru MQRFH2. Musíte zvážit dva problémy s návrhem aplikace: 1. Aplikace připojená ke správci front partnera musí být schopna zpracovat zprávy obsahující záhlaví MQRFH2 generovaná ve správci front IBM MQ 9.3 . 2. Aplikace připojená ke správci front partnera musí zpracovat nové vlastnosti zprávy, které jsou správně označeny příznakem MQPD_SUPPORT_REQUIRED . S nastavenou volbou kanálu ALL mohou aplikace JMS spolupracovat mezi produktem IBM MQ 9.3 a dřívější verzí s použitím kanálu. Nové aplikace IBM MQ 9.3 používající vlastnosti zpráv mohou spolupracovat s aplikacemi ze starší verze v závislosti na způsobu, jakým aplikace starší verze zpracovává záhlaví MQRFH2 .
COMPAT	Pomocí této volby můžete v některých případech odeslat vlastnosti zprávy aplikacím připojeným ke správci front partnera dřívější verze, nikoli však všem. Vlastnosti zprávy jsou odeslány pouze v případě, že jsou splněny dvě podmínky: 1. Žádná vlastnost nesmí být označena jako vyžadující zpracování vlastnosti zprávy. 2. Alespoň jedna z vlastností zprávy musí být ve "vyhrazené" složce; viz Poznámka. S nastavenou volbou kanálu COMPAT mohou aplikace JMS spolupracovat mezi produktem IBM MQ 9.3 a dřívější verzí pomocí kanálu. Kanál není k dispozici pro všechny aplikace používající vlastnosti zprávy, pouze pro ty aplikace, které používají vyhrazené složky. Pravidla týkající se toho, zda je zpráva nebo vlastnost odeslána, jsou: 1. Pokud má zpráva vlastnosti, ale žádná z vlastností není přidružena ke složce "reserved" , nebudou odeslány žádné vlastnosti zprávy. 2. Pokud byla ve "vyhrazené" složce vlastností vytvořena nějaká vlastnost zprávy, budou odeslány všechny vlastnosti zprávy přidružené ke zprávě. Nicméně: a. Pokud je některá z vlastností zprávy označena jako vyžadovaná podpora, MQPD_SUPPORT_REQUIRED nebo MQPD_SUPPORT_REQUIRED_IF_LOCAL, celá zpráva bude odmítnuta. Je vrácena, vyřazena nebo odeslána do fronty nedoručených zpráv podle hodnoty voleb sestavy. b. Pokud nejsou žádné vlastnosti zprávy označeny jako povinné pro podporu, nemusí být individuální vlastnost odeslána. Je-li některé z polí deskriptoru vlastností zprávy nastaveno na jiné než výchozí hodnoty, není individuální vlastnost odeslána. Zpráva je stále odeslána. Příkladem jiné než výchozí hodnoty pole deskriptoru vlastnosti je MQPD_USER_CONTEXT. **Poznámka:** Názvy "vyhrazených" složek začínají znaky mcd., jms., usr.nebo mqext.. Tyto složky jsou vytvořeny pro aplikace, které používají rozhraní JMS . V produktu IBM MQ 9.3 jsou všechny dvojice název-hodnota, které jsou umístěny v těchto složkách, považovány za vlastnosti zprávy. Vlastnosti zprávy se odesílají v záhlaví MQRFH2 , kromě všech dvojic název-hodnota umístěných v záhlaví MQRFH2 . Jakékoli dvojice název-hodnota umístěné v záhlaví MQRFH2 se odešlou, dokud nebude zpráva odmítnuta.

Tabulka 491. Nastavení atributu vlastnosti zprávy kanálu (pokračování)	
PROPCTL	**Popis**
ŽÁDNÉ	Pomocí této volby můžete zabránit odesílání vlastností zpráv aplikacím připojeným ke správci front partnera předchozí verze. Soubor MQRFH2 , který obsahuje dvojice název-hodnota a vlastnosti zprávy, je stále odeslán, ale pouze s dvojicemi název-hodnota. S nastavenou volbou kanálu NONE je zpráva JMS odeslána jako JMSTextMessage nebo JMSBytesMessage bez vlastností zprávy JMS . Je-li možné, aby aplikace starší verze ignorovala všechny vlastnosti nastavené v aplikaci IBM MQ 9.3 , může s ní spolupracovat.

PROPCTL volby fronty pro MQGMO

Pomocí atributu fronty **PROPCTL** můžete řídit, jak jsou vlastnosti zprávy vráceny aplikaci, která volá **MQGET** bez nastavení voleb vlastností zprávy **MQGMO** .

Tabulka 492. Nastavení atributu vlastnosti zprávy fronty	
PROPCTL	**Popis**
all	Použijte volbu ALL , aby různé aplikace, které čtou zprávu ze stejné fronty, mohly zpracovat zprávu různými způsoby. • Aplikace, migrovaná beze změny ze starší verze, může pokračovat v přímém čtení MQRFH2 . Vlastnosti jsou přímo přístupné v záhlaví MQRFH2 . Musíte upravit aplikaci tak, aby zpracovávala všechny nové vlastnosti a nové atributy vlastností. Je možné, že aplikace může být ovlivněna změnami v rozvržení a počtem záhlaví MQRFH2 . Některé atributy složky mohou být odebrány nebo produkt IBM MQ ohlásí chybu v rozvržení záhlaví MQRFH2 , kterou v dřívější verzi ignoroval. • Nová nebo změněná aplikace může použít vlastnost zprávy MQI k dotazování na vlastnosti zprávy a k přímému čtení dvojic název-hodnota v záhlaví MQRFH2 . Všechny vlastnosti ve zprávě jsou vráceny aplikaci. • Pokud aplikace volá MQCRTMH k vytvoření popisovače zprávy, musí se dotázat na vlastnosti zprávy pomocí MQINQMP. Dvojice název-hodnota, které nejsou vlastnostmi zprávy, zůstávají v souboru MQRFH2, který je zbaven všech vlastností zprávy. • Pokud aplikace nevytvoří popisovač zprávy, všechny vlastnosti zprávy a dvojice název-hodnota zůstanou v souboru MQRFH2. Volba ALL má tento efekt pouze v případě, že přijímající aplikace nenastavila volbu MQGMO_PROPERTIES nebo ji nastavila na hodnotu MQGMO_PROPERTIES_AS_Q_DEF.

Tabulka 492. Nastavení atributu vlastnosti zprávy fronty (pokračování)

PROPCTL	**Popis**
COMPAT (výchozí)	Výchozí volba je COMPAT . Není-li parametr GMO_PROPERTIES_* nastaven, jako v nezměněné aplikaci ze starší verze se předpokládá COMPAT . Při použití výchozí volby COMPAT funguje aplikace starší verze, která explicitně nevytvořila MQRFH2, beze změny na systému IBM MQ 9.3. Tuto volbu použijte, pokud jste napsali aplikaci MQI starší verze pro čtení zpráv produktu JMS . • Vlastnosti JMS uložené v záhlaví MQRFH2 jsou vráceny aplikaci v záhlaví MQRFH2 ve složkách s názvy začínajícími na mcd., jms., usr. nebo mqext. • Pokud má zpráva složky JMS a aplikace IBM MQ 9.3 přidá do zprávy nové složky vlastností, tyto vlastnosti se také vrátí v souboru MQRFH2. V důsledku toho musíte upravit aplikaci tak, aby zpracovávala všechny nové vlastnosti a nové atributy vlastností. Je možné, že nezměněná aplikace může být ovlivněna změnami v rozvržení a počtem záhlaví MQRFH2 . Může najít, že některé atributy složky byly odebrány, nebo že produkt IBM MQ najde chyby v rozvržení záhlaví MQRFH2 , které v dřívější verzi ignoroval. **Poznámka:** V tomto scénáři je chování aplikace stejné bez ohledu na to, zda je připojena ke starší verzi nebo ke správci front IBM MQ 9.3 . Je-li atribut kanálu **PROPCTL** nastaven na hodnotu COMPAT nebo ALL , budou všechny nové vlastnosti zprávy odeslány ve zprávě správci front partnera dřívější verze. • Pokud zpráva není zprávou JMS , ale obsahuje další vlastnosti, nejsou tyto vlastnosti vráceny aplikaci v záhlaví MQRFH2 .[1] • Tato volba také v mnoha případech povoluje dřívější verze aplikací, které explicitně vytvářejí MQRFH2 , aby fungovaly správně. Například program MQI, který vytváří MQRFH2 obsahující JMS vlastnosti zprávy, nadále pracuje správně. Pokud je zpráva vytvořena bez vlastností zprávy JMS , ale s některými dalšími složkami MQRFH2 , jsou tyto složky vráceny do aplikace. Pouze v případě, že složky jsou složky vlastností zpráv, jsou tyto specifické složky odebrány z adresáře MQRFH2. Složky vlastností zpráv jsou identifikovány tím, že mají nový atribut složky content='properties', nebo se jedná o složky s názvy uvedenými v části Název složky definovaných vlastností nebo Název složky neseskupených vlastností. • Pokud aplikace volá MQCRTMH k vytvoření popisovače zprávy, musí se dotázat na vlastnosti zprávy pomocí MQINQMP. Vlastnosti zprávy jsou odebrány ze záhlaví MQRFH2 . Dvojice název-hodnota, které nejsou vlastnostmi zprávy, zůstávají v souboru MQRFH2. • Pokud aplikace volá MQCRTMH k vytvoření popisovače zprávy, může se dotazovat na všechny vlastnosti zprávy bez ohledu na to, zda má zpráva složky JMS . • Pokud aplikace nevytvoří popisovač zprávy, všechny vlastnosti zprávy a dvojice název-hodnota zůstanou v souboru MQRFH2. Pokud zpráva obsahuje nové složky uživatelských vlastností, můžete odvodit, že zpráva byla vytvořena novou nebo změněnou aplikací IBM MQ 9.3 . Má-li přijímající aplikace zpracovat tyto nové vlastnosti přímo v produktu MQRFH2, musíte upravit aplikaci tak, aby používala volbu ALL . Je-li nastavena výchozí volba COMPAT , neupravená aplikace pokračuje ve zpracování zbytku souboru MQRFH2bez vlastností IBM MQ 9.3 . Účelem rozhraní PROPCTL je podporovat staré aplikace, které čtou složky MQRFH2 , a nové a změněné aplikace, které používají rozhraní vlastností zpráv. Cílem je, aby nové aplikace používaly rozhraní vlastností zpráv pro všechny vlastnosti uživatelských zpráv a aby se zabránilo přímému čtení a zápisu záhlaví MQRFH2 . COMPAT má tento efekt pouze v případě, že přijímající aplikace nenastavila volbu MQGMO_PROPERTIES nebo ji nastavila na hodnotu MQGMO_PROPERTIES_AS_Q_DEF.

Tabulka 492. Nastavení atributu vlastnosti zprávy fronty (pokračování)	
PROPCTL	**Popis**
Vynutit	Volba FORCE umístí všechny vlastnosti zpráv do záhlaví MQRFH2 . Všechny vlastnosti zprávy a dvojice název-hodnota v záhlavích MQRFH2 zůstávají ve zprávě. Vlastnosti zprávy nejsou odebrány ze serveru MQRFH2a jsou zpřístupněny prostřednictvím popisovače zprávy. Výsledkem výběru volby FORCE je umožnit nově migrované aplikaci číst vlastnosti zprávy ze záhlaví MQRFH2 . Předpokládejme, že jste upravili aplikaci tak, aby zpracovávala vlastnosti zprávy IBM MQ 9.3 , ale zároveň jste zachovali její schopnost pracovat přímo se záhlavími MQRFH2 , jako dříve. Můžete se rozhodnout, kdy přepnout aplikaci na použití vlastností zprávy, a to tak, že na začátku nastavíte atribut fronty PROPCTL na hodnotu FORCE. Nastavte atribut fronty **PROPCTL** na jinou hodnotu, až budete připraveni začít používat vlastnosti zprávy. Pokud se nová funkce v aplikaci nechová tak, jak jste očekávali, nastavte volbu **PROPCTL** zpět na FORCE. Volba FORCE má tento účinek pouze v případě, že přijímající aplikace nenastavila volbu MQGMO_PROPERTIES nebo ji nastavila na hodnotu MQGMO_PROPERTIES_AS_Q_DEF.
ŽÁDNÉ	Použijte volbu NONE , aby existující aplikace mohla zpracovat zprávu, ignorovat všechny vlastnosti zprávy a nová nebo změněná aplikace se mohla dotazovat na vlastnosti zprávy. • Pokud aplikace volá MQCRTMH k vytvoření popisovače zprávy, musí se dotázat na vlastnosti zprávy pomocí MQINQMP. Dvojice název-hodnota, které nejsou vlastnostmi zprávy, zůstávají v souboru MQRFH2, který je zbaven všech vlastností zprávy. • Pokud aplikace nevytvoří manipulátor zprávy, všechny vlastnosti zprávy se odeberou z MQRFH2. Dvojice název-hodnota v záhlavích MQRFH2 zůstávají ve zprávě. Volba NONE má tento efekt pouze v případě, že přijímající aplikace nenastavila volbu MQGMO_PROPERTIES nebo ji nastavila na hodnotu MQGMO_PROPERTIES_AS_Q_DEF.
V6COMPAT	Tuto volbu použijte, chcete-li přijmout MQRFH2 ve stejném formátu, jako byl odeslán. Pokud odesílající aplikace nebo správce front vytvoří další vlastnosti zprávy, jsou tyto vlastnosti vráceny v popisovači zprávy. Tato volba musí být nastavena pro odesílací i přijímací fronty i pro všechny vedlejší přenosové fronty. Potlačí všechny ostatní volby PROPCTL nastavené v definicích front v cestě k rozlišení názvů front. Volbu V6COMPAT použijte pouze za výjimečných okolností. Pokud například provádíte migraci aplikací ze starší verze na verzi IBM MQ 9.3, je tato volba cenná, protože zachovává chování starší verze. Tato volba pravděpodobně ovlivní propustnost zpráv. Je také obtížnější spravovat; musíte se ujistit, že volba je nastavena na odesílatele, příjemce a zasahování do přenosových front. V6COMPAT má tento efekt pouze v případě, že přijímající aplikace nenastavila volbu MQGMO_PROPERTIES nebo ji nastavila na MQGMO_PROPERTIES_AS_Q_DEF.

Další informace o vlastnostech zprávy a dvojicích název-hodnota viz “NameValueData (MQCHARn) pro MQRFH2” na stránce 535.

Volby vlastností zprávy pro MQGMO

Pomocí voleb vlastností zprávy **MQGMO** můžete řídit, jak jsou vlastnosti zprávy vraceny do aplikace.

[1] Existence specifických složek vlastností vytvořených pomocí IBM MQ classes for JMS označuje zprávu JMS . Složky vlastností jsou mcd., jms., usr.nebo mqext.

Tabulka 493. Nastavení volby vlastnosti zprávy MQGMO	
MQGMO Volba	**Popis**
MQGMO_PROPERTIES_AS_Q_DEF	IBM MQ aplikace, které čtou ze stejné fronty a nenastavují GMO_PROPERTIES_*, přijímají vlastnosti zprávy odlišně. Aplikace systému IBM MQ , které nevytvářejí manipulátor zpráv, jsou řízeny atributem fronty **PROPCTL** . Aplikace IBM MQ může v produktu MQRFH2zvolit příjem vlastností zprávy nebo vytvořit popisovač zprávy a dotázat se na vlastnosti zprávy. Pokud aplikace vytvoří popisovač zprávy, vlastnosti se odeberou z MQRFH2. • Nová nebo změněná aplikace IBM MQ , která nenastavuje GMO_PROPERTIES_* nebo ji nastavuje na MQGMO_PROPERTIES_AS_Q_DEF , může zvolit dotazování na vlastnosti zprávy. Musí nastavit MQCRTMH pro vytvoření popisovače zprávy a vlastností zprávy dotazu pomocí volání MQINQMP MQI. • Pokud nová nebo změněná aplikace nevytvoří manipulátor zprávy, musí číst všechny vlastnosti zprávy, které obdrží přímo ze záhlaví MQRFH2 . • Je-li atribut fronty **PROPCTL** nastaven na hodnotu FORCE, nejsou v popisovači zprávy vráceny žádné vlastnosti. Všechny vlastnosti jsou vráceny v záhlavích MQRFH2 . • Pokud je atribut fronty **PROPCTL** nastaven na hodnotu NONEnebo COMPAT, aplikace IBM MQ , která vytvoří popisovač zprávy, obdrží všechny vlastnosti zprávy.
MQGMO_PROPERTIES_IN_HANDLE	Vynutit, aby aplikace používala vlastnosti zprávy. Pomocí této volby zjistíte, zda se upravené aplikaci nepodaří vytvořit popisovač zprávy. Aplikace se možná pokouší číst vlastnosti zprávy přímo z MQRFH2, spíše než volat MQINQMP.
MQGMO_NO_PROPERTIES	• Všechny vlastnosti jsou odebrány. Vlastnosti generované správcem front, například vlastnosti JMS , jsou odebrány. • Vlastnosti jsou odebrány i v případě, že je vytvořen manipulátor zprávy. Dvojice název-hodnota v jiných složkách MQRFH2 jsou k dispozici v datech zprávy.
MQGMO_PROPERTIES_FORCE_MQRFH2	Vlastnosti jsou vráceny v záhlavích MQRFH2 , a to i v případě, že je vytvořen manipulátor zprávy. • Funkce MQINQMP nevrací žádné vlastnosti zprávy, a to ani v případě, že je vytvořen manipulátor zprávy. Hodnota MQRC_PROPERTY_NOT_AVAILABLE je vrácena v případě, že je zjišťována vlastnost.

Tabulka 493. Nastavení volby vlastnosti zprávy MQGMO (pokračování)	
MQGMO Volba	**Popis**
MQGMO_PROPERTIES_COMPATIBILITY	Pokud zpráva pochází od klienta JMS , vlastnosti JMS jsou vráceny v záhlavích MQRFH2 . Nové nebo upravené aplikace IBM MQ , které vytvářejí popisovač zprávy, se chovají odlišně. • Všechny vlastnosti ve všech složkách vlastností zpráv jsou vráceny, pokud zpráva obsahuje složku mcd., jms., usr.nebo mqext . • Pokud zpráva obsahuje složky vlastností, ale ne složku mcd., jms., usr.nebo mqext , nejsou v souboru MQRFH2vráceny žádné vlastnosti zprávy. • Pokud nová nebo upravená aplikace IBM MQ vytvoří popisovač zprávy, vlastnosti zprávy dotazu pomocí volání MQINQMP MQI. Všechny vlastnosti zprávy jsou odebrány z MQRFH2. • Pokud nová nebo upravená aplikace IBM MQ vytvoří popisovač zprávy, lze se dotazovat na všechny vlastnosti ve zprávě. I když zpráva neobsahuje složku mcd., jms., usr.nebo mqext , všechny vlastnosti zprávy jsou dotazovatelné.

Související odkazy
PROPCTL
2471 (09A7) (RC2471): MQRC_PROPERTY_NOT_AVAILABLE

StrucId (MQCHAR4) pro MQGMO
Jedná se o identifikátor struktury struktury voleb získání zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQGMO_STRUC_ID.

Hodnota musí být:

MQGMO_STRUC_ID
Identifikátor pro strukturu voleb získání zprávy.

Pro programovací jazyk C je definována také konstanta MQGMO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQGMO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQGMO
Verze je číslo verze struktury.

Hodnota musí být jedna z následujících:

MQGMO_VERSION_1
Version-1 struktura voleb get-message.

Tato verze je podporována ve všech prostředích.

MQGMO_VERSION_2
Version-2 struktura voleb get-message.

Tato verze je podporována ve všech prostředích.

MQGMO_VERSION_3
Version-3 struktura voleb get-message.

Tato verze je podporována ve všech prostředích.

MQGMO_VERSION_4
Version-4 struktura voleb get-message.

Tato verze je podporována ve všech prostředích.

Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MQGMO_CURRENT_VERSION
Aktuální verze struktury voleb get-message.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQGMO_VERSION_1.

Volby (MQLONG) pro MQGMO

Volby **MQGMO** řídí akci MQGET. Můžete uvést nula nebo více voleb. Pokud potřebujete více než jednu volitelnou hodnotu:

- Přidejte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo
- Zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Kombinace voleb, které nejsou platné, jsou uvedeny; všechny ostatní kombinace jsou platné.

Volby čekání

Následující volby se týkají čekání na doručení zpráv do fronty:

MQGMO_WAIT
Aplikace čeká na doručení vhodné zprávy. Maximální doba, po kterou aplikace čeká, je uvedena v souboru *WaitInterval*.

Důležité: Neexistuje žádné čekání nebo prodleva, pokud je okamžitě k dispozici vhodná zpráva.

Pokud je zablokováno MQGET požadavků nebo MQGET požadavků bude zablokováno během čekání, čekání se zruší. Volání se dokončí s MQCC_FAILED a kódem příčiny MQRC_GET_INHIBITEDbez ohledu na to, zda jsou ve frontě vhodné zprávy.

Můžete použít MQGMO_WAIT s volbami MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT .

Pokud několik aplikací čeká ve stejné sdílené frontě, následující pravidla vyberou, která aplikace se aktivuje při doručení vhodné zprávy:

Tabulka 494. Pravidla pro aktivaci volání MQGET ve sdílené frontě.

Počet volání MQGET čekajících na aktivaci		**Výsledek**
S volbou BROWSE	**Bez volby BROWSE**[2]	
Není	Jedna a více	Je aktivováno jedno volání MQGET bez volby BROWSE .
Jedna a více	Není	Všechna volání MQGET s volbou BROWSE jsou aktivována.
Jedna a více	Jedna a více	Je aktivováno jedno volání MQGET bez volby BROWSE . Počet volání MQGET s aktivovanou volbou BROWSE je nepředvídatelný.

Pokud ve stejné frontě čeká více než jedno volání MQGET bez volby BROWSE , aktivuje se pouze jedno. Správce front se pokusí přidělit prioritu čekajícím voláním v následujícím pořadí:

1. Specifické požadavky get-wait, které mohou být uspokojeny pouze určitými zprávami, například těmi, které mají specifickou hodnotu MsgId nebo CorrelId (nebo obojí).
2. Obecné požadavky get-wait, které mohou být uspokojeny libovolnou zprávou.

Poznámka:

- V rámci první kategorie nejsou pro konkrétnější požadavky get-wait přiřazeny žádné další priority. Například požadavky, které uvádějí jak MsgId , tak CorrelId .

[2] Volání MQGET určující volbu MQGMO_LOCK je považováno za volání bez procházení.

- V rámci jedné z těchto kategorií nelze předpovědět, která aplikace je vybrána. Zejména aplikace, která čeká nejdéle, nemusí být nutně vybrána.
- Délka cesty a aspekty plánování priorit operačního systému mohou znamenat, že čekající aplikace s nižší prioritou operačního systému, než se očekávalo, načte zprávu.
- Může se také stát, že aplikace, která nečeká, načte zprávu raději než takovou, která je.

z/OS V systému z/OSplatí následující body:

- Pokud chcete, aby aplikace při čekání na doručení zprávy pokračovala v jiné práci, zvažte místo toho použití volby signálu (MQGMO_SET_SIGNAL). Volba signálu je však specifická pro dané prostředí; aplikace, které chcete portovat mezi různými prostředími, ji nesmí používat.
- Pokud existuje více než jedno volání MQGET čekající na stejnou zprávu s kombinací voleb čekání a signálu, je každé čekající volání považováno za stejné. Jedná se o chybu při zadávání MQGMO_SET_SIGNAL s MQGMO_WAIT. Je také chybou specifikovat tuto volbu s popisovačem fronty, pro kterou je signál neprovedený.
- Pokud uvedete MQGMO_WAIT nebo MQGMO_SET_SIGNAL pro frontu, která má IndexType hodnotu MQIT_MSG_TOKEN, nejsou povolena žádná kritéria výběru. To znamená, že:
  - Pokud používáte version-1 MQGMO, nastavte pole MsgId a CorrelId v MQMD určeném ve volání MQGET na MQMI_NONE a MQCI_NONE.
  - Používáte-li verzi version-2 nebo novější MQGMO, nastavte pole MatchOptions na hodnotu MQMO_NONE.
- Pro volání MQGET ve sdílené frontě a volání je požadavek na procházení nebo destruktivní získání skupinové zprávy, a ani MsgId ani CorrelId se nemají shodovat, je váš signál ECB odeslán MQEC_MSG_ARRebo po 200 milisekundách.

  K tomu dochází i v případě, že do fronty nedorazila vhodná zpráva, a to až do vypršení intervalu čekání, kdy je fronta odeslána s parametrem MQEC_WAIT_INTERVAL_VYPRŠELA. Po odeslání zprávy MQEC_MSG_ARRIDEmusíte znovu zadat druhé volání MQGET , aby se zpráva načetl, je-li k dispozici.

  Tato technika se používá k zajištění toho, že budete včas informováni o příchodu zprávy, ale může se objevit jako neočekávaná režie zpracování ve srovnání s podobnou posloupností volání v nesdílené frontě.

Parametr MQGMO_WAIT je ignorován, pokud je zadán s parametrem MQGMO_BROWSE_MSG_UNDER_CURSOR nebo MQGMO_MSG_UNDER_CURSOR ; není vyvolána žádná chyba.

MQGMO_NO_WAIT
Aplikace nečeká, pokud není k dispozici žádná vhodná zpráva. MQGMO_NO_WAIT je opak MQGMO_WAIT. MQGMO_NO_WAIT je definován jako pomůcka pro programovou dokumentaci. Jedná se o předvolbu, není-li zadán ani jeden z nich.

MQGMO_SET_SIGNAL
Tuto volbu použijte s poli Signal1 a Signal2 . Umožňuje aplikacím pokračovat s jinou prací při čekání na doručení zprávy. Také umožňuje (jsou-li k dispozici vhodné prostředky operačního systému) aplikacím čekat na zprávy přicházející do více než jedné fronty.

Poznámka: Volba MQGMO_SET_SIGNAL je specifická pro prostředí; nepoužívejte ji pro aplikace, které chcete portovat.

Za dvou okolností se volání dokončí stejným způsobem, jako kdyby tato volba nebyla uvedena:

1. Pokud aktuálně dostupná zpráva splňuje kritéria uvedená v deskriptoru zprávy.
2. Pokud je zjištěna chyba parametru nebo jiná synchronní chyba.

Není-li aktuálně k dispozici žádná zpráva splňující kritéria zadaná v deskriptoru zprávy, vrátí se řízení aplikaci bez čekání na doručení zprávy. Parametry **CompCode** a **Reason** jsou nastaveny na MQCC_WARNING a MQRC_SIGNAL_REQUEST_ACCEPTED. Další výstupní pole v deskriptoru zprávy

a výstupní parametry volání MQGET nejsou nastaveny. Jakmile přijde vhodná zpráva později, je signál doručen vysláním ECB.

Volající pak musí znovu zadat volání MQGET , aby načetl zprávu. Aplikace může čekat na tento signál pomocí funkcí poskytovaných operačním systémem.

Pokud operační systém poskytuje mechanizmus vícenásobného čekání, můžete jej použít k čekání na zprávu přicházející do kterékoli z několika front.

Je-li zadána nenulová hodnota WaitInterval , bude signál doručen po uplynutí čekacího intervalu. Správce front může také zrušit čekání. V takovém případě je signál doručen.

Více než jedno volání MQGET může nastavit signál pro stejnou zprávu. Pořadí, ve kterém jsou aplikace aktivovány, je stejné jako u MQGMO_WAIT.

Pokud na stejnou zprávu čeká více než jedno volání MQGET , je každé čekající volání považováno za stejné. Volání mohou obsahovat kombinaci možností čekání a signálu.

Za určitých podmínek může volání MQGET načíst zprávu a signál, který je výsledkem přijetí stejné zprávy, může být doručen. Když je signál doručen, aplikace musí být připravena, aby nebyla k dispozici žádná zpráva.

Popisovač fronty nemůže mít více než jeden neprovedený požadavek na signál.

Tato volba není platná s žádnou z následujících voleb:

- MQGMO_UNLOCK
- MQGMO_WAIT

Pro volání MQGET ve sdílené frontě a volání je požadavek na procházení nebo destruktivní získání skupinové zprávy a ani MsgId ani CorrelId se neshodují, je signál uživatele ECB zveřejněn MQEC_MSG_ARRIVED po 200 milisekundách.

K tomu dojde, i když do fronty nedorazila vhodná zpráva, dokud nevyprší čekací interval, když je fronta zapsána s produktem MQEC_WAIT_INTERVAL_EXPIRED. Po odeslání zprávy MQEC_MSG_ARRIVED musíte znovu zadat druhé volání MQGET , abyste zprávu načetli, je-li k dispozici.

Tato technika se používá k zajištění toho, že budete včas informováni o příchodu zprávy, ale může se objevit jako neočekávaná režie zpracování ve srovnání s podobnou posloupností volání v nesdílené frontě.

Nejedná se o efektivní metodu načítání zpráv, pokud jsou zprávy přidávány zřídka. Chcete-li se vyhnout této režii pro případ procházení, uveďte MsgId (pokud není indexováno nebo indexováno pomocí MsgId) nebo *CorrelId* (pokud je indexováno pomocí CorrelId) odpovídající volání MQGET .

z/OS Tato volba je podporována pouze na systému z/OS .

MQGMO_FAIL_IF_QUIESCING
Vynutí selhání volání MQGET , pokud je správce front ve stavu uvedení do klidového stavu.

z/OS V systému z/OStato volba také vynutí selhání volání MQGET , pokud je připojení (pro aplikaci CICS nebo IMS) v klidovém stavu.

Je-li tato volba zadána s volbou MQGMO_WAIT nebo MQGMO_SET_SIGNALa v době, kdy správce front přejde do klidového stavu, probíhá čekání nebo signál:

- Čekání je zrušeno a volání vrátí kód dokončení MQCC_FAILED s kódem příčiny MQRC_Q_MGR_QUIESCING nebo MQRC_CONNECTION_QUIESCING.
- Signál je zrušen s kódem dokončení signálu specifickým pro dané prostředí.

  z/OS V systému z/OSse signál dokončí s kódem dokončení události MQEC_Q_MGR_QUIESCING nebo MQEC_CONNECTION_QUIESCING.

Pokud není zadána hodnota MQGMO_FAIL_IF_QUIESCING a správce front nebo připojení přejde do klidového stavu, nebude čekání ani signál zrušen.

Volby synchronizačního bodu

Následující volby se týkají účasti volání MQGET v rámci pracovní jednotky:

MQGMO_SYNCPOINT
Požadavek je pracovat v rámci běžných protokolů jednotky práce. Zpráva je označena jako nedostupná pro jiné aplikace, ale je odstraněna z fronty pouze v případě, že je transakce potvrzena. Zpráva se znovu zpřístupní, pokud je jednotka práce vrácena zpět.

Můžete nechat MQGMO_SYNCPOINT a MQGMO_NO_SYNCPOINT nenastavené. V takovém případě je zahrnutí požadavku get do protokolů jednotky práce určeno prostředím, ve kterém je spuštěn správce front. Není určen prostředím, ve kterém je spuštěna aplikace.

- z/OS V systému z/OSje požadavek na získání v rámci pracovní jednotky.
- Ve všech prostředích kromě prostředí z/OSnení požadavek na získání v rámci pracovní jednotky.

Vzhledem k těmto rozdílům nesmí aplikace, kterou chcete portovat, povolit tuto volbu jako výchozí; zadejte MQGMO_SYNCPOINT nebo MQGMO_NO_SYNCPOINT explicitně.

Tato volba není platná s žádnou z následujících voleb:

- 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
Požadavek je pracovat v rámci normálních protokolů jednotky práce, ale pouze v případě, že načtená zpráva je trvalá. Trvalá zpráva má hodnotu MQPER_PERSISTENT v poli Persistence v poli MQMD.

- Pokud je zpráva trvalá, správce front zpracuje volání, jako by aplikace určila MQGMO_SYNCPOINT.
- Pokud zpráva není trvalá, správce front zpracuje volání, jako by aplikace zadala hodnotu MQGMO_NO_SYNCPOINT.

Tato volba není platná s žádnou z následujících voleb:

- 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

Tato volba je podporována v následujících prostředích:

- AIX AIX
- IBM i IBM i
- Linux Linux
- z/OS z/OS

a pro systém IBM MQ MQI clients připojený k těmto systémům.

MQGMO_NO_SYNCPOINT

Požadavek je pracovat mimo normální protokoly jednotky práce. Pokud obdržíte zprávu bez volby procházení, je okamžitě odstraněna z fronty. Zprávu nelze znovu zpřístupnit zálohováním pracovní jednotky.

Tato volba se předpokládá, pokud uvedete MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT.

Můžete nechat MQGMO_SYNCPOINT a MQGMO_NO_SYNCPOINT nenastavené. V takovém případě je zahrnutí požadavku get do protokolů jednotky práce určeno prostředím, ve kterém je spuštěn správce front. Není určen prostředím, ve kterém je spuštěna aplikace.

- z/OS V systému z/OSje požadavek na získání v rámci pracovní jednotky.
- Ve všech prostředích kromě prostředí z/OSnení požadavek na získání v rámci pracovní jednotky.

Vzhledem k těmto rozdílům nesmí aplikace, kterou chcete portovat, povolit tuto volbu jako výchozí; uveďte buď MQGMO_SYNCPOINT , nebo MQGMO_NO_SYNCPOINT explicitně.

Tato volba není platná s žádnou z následujících voleb:

- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT

z/OS **MQGMO_MARK_SKIP_BACKOUT**

Vraťte pracovní jednotku zpět, aniž byste ve frontě obnovili zprávu, která byla označena touto volbou.

Tato volba je podporována pouze na systému z/OS.

Je-li zadána tato volba, musí být zadána také hodnota MQGMO_SYNCPOINT . Parametr MQGMO_MARK_SKIP_BACKOUT není platný s žádnou z následujících voleb:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT
- MQGMO_LOCK
- MQGMO_NO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

Poznámka: V systémech IMS a CICSbudete možná muset po zálohování pracovní jednotky obsahující zprávu označenou jako MQGMO_MARK_SKIP_BACKOUTvydat volání extran IBM MQ . Před potvrzením nové pracovní jednotky obsahující označenou zprávu musíte zadat volání IBM MQ . Volání může být libovolné volání IBM MQ , které se vám líbí.

1. Pokud jste v systému IMSnepoužili opravu IMS APAR PN60855 a spouštíte aplikaci IMS MPP nebo BMP.
2. V systému CICS, pokud spouštíte jakoukoli aplikaci.

V obou případech zadejte jakékoli volání IBM MQ před potvrzením nové pracovní jednotky obsahující zálohovanou zprávu.

Poznámka: V rámci pracovní jednotky může být pouze jeden požadavek na získání označený jako vynechání, stejně jako žádný nebo několik neoznačených požadavků na získání.

Pokud dojde k odvolání aplikace z pracovní jednotky, zpráva, která byla načtena pomocí produktu MQGMO_MARK_SKIP_BACKOUT , se neobnoví do předchozího stavu. Ostatní aktualizace prostředků jsou vráceny zpět. Se zprávou se zachází, jako by byla načtena v nové pracovní jednotce spuštěné požadavkem na vrácení. Zpráva se načte bez volby MQGMO_MARK_SKIP_BACKOUT .

Parametr MQGMO_MARK_SKIP_BACKOUT je užitečný, pokud se po změně některých prostředků ukáže, že transakci nelze úspěšně dokončit. Pokud tuto volbu vynecháte, zálohování pracovní jednotky obnoví zprávu ve frontě. Stejná posloupnost událostí se vyskytne znovu, když je zpráva znovu načtena.

Pokud však při původním volání funkce MQGET zadáte hodnotu MQGMO_MARK_SKIP_BACKOUT , bude zálohováním jednotky práce dojde k odvolání aktualizací ostatních prostředků. Se zprávou se zachází, jako by byla načtena pod novou pracovní jednotkou. Aplikace může provádět odpovídající ošetření chyb. Může odeslat zprávu sestavy odesílateli původní zprávy nebo umístit původní zprávu do fronty nedoručených zpráv. Poté může potvrdit novou pracovní jednotku. Potvrzení nové pracovní jednotky odebere zprávu trvale z původní fronty.

MQGMO_MARK_SKIP_BACKOUT označuje jednu fyzickou zprávu. Pokud zpráva patří do skupiny zpráv, ostatní zprávy ve skupině nejsou označeny. Podobně, pokud je označená zpráva segmentem logické zprávy, ostatní segmenty v logické zprávě nejsou označeny.

Jakoukoli zprávu ve skupině lze označit, ale pokud jsou zprávy načteny pomocí produktu MQGMO_LOGICAL_ORDER, je výhodné označit první zprávu ve skupině. Pokud je jednotka práce vrácena zpět, první (označená) zpráva se přesune do nové jednotky práce. Druhá a pozdější zpráva ve skupině jsou znovu uvedeny do fronty. Zprávy zanechané ve frontě nemohou být načteny jinou aplikací pomocí MQGMO_LOGICAL_ORDER. První zpráva ve skupině již není ve frontě. Avšak aplikace, která zálohovala jednotku práce, může načíst druhou a pozdější zprávu do nové jednotky práce pomocí volby MQGMO_LOGICAL_ORDER . První zpráva již byla načtena.

Občas budete muset vycouvat z nové jednotky práce. Například protože fronta nedoručených zpráv je plná a zpráva nesmí být vyřazena. Zálohování nové pracovní jednotky obnoví zprávu v původní frontě, což zabrání ztrátě zprávy. Avšak v této situaci nemůže zpracování pokračovat. Po zálohování nové pracovní jednotky musí aplikace informovat operátora nebo administrátora, že došlo k neopravitelné chybě, a poté ji dokončit.

Funkce MQGMO_MARK_SKIP_BACKOUT funguje pouze v případě, že je pracovní jednotka obsahující požadavek na získání přerušena aplikací, která ji zálohuje. Pokud je pracovní jednotka obsahující požadavek na získání vrácena zpět, protože transakce nebo systém selhaly, MQGMO_MARK_SKIP_BACKOUT se ignoruje. Všechny zprávy načtené pomocí této volby jsou znovu zařazeny do fronty stejným způsobem jako zprávy načtené bez této volby.

Volby procházení

Následující volby se týkají procházení zpráv ve frontě:

MQGMO_BROWSE_FIRST
Při otevření fronty s volbou MQOO_BROWSE je vytvořen kurzor procházení, který je logicky umístěn před první zprávou ve frontě. Poté můžete použít volání MQGET uvádějící volbu MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXTnebo MQGMO_BROWSE_MSG_UNDER_CURSOR k nedestruktivnímu načtení zpráv z fronty. Kurzor procházení označuje pozici v rámci zpráv ve frontě, ze které další volání MQGET s voláním funkce MQGMO_BROWSE_NEXT vyhledává vhodnou zprávu.

Parametr MQGMO_BROWSE_FIRST není platný s žádnou z následujících voleb:

- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

Také se jedná o chybu, pokud nebyla fronta otevřena pro procházení.

Volání MQGET s MQGMO_BROWSE_FIRST ignoruje předchozí pozici kurzoru procházení. Načte se první zpráva ve frontě, která splňuje podmínky uvedené v deskriptoru zprávy. Zpráva zůstává ve frontě a kurzor procházení je umístěn na této zprávě.

Po tomto volání je kurzor procházení umístěn na vrácené zprávě. Zpráva může být odebrána z fronty před vydáním dalšího volání MQGET s MQGMO_BROWSE_NEXT . V tomto případě zůstane kurzor procházení na pozici ve frontě, kde byla zpráva obsazena, i když je tato pozice nyní prázdná.

Použijte volbu MQGMO_MSG_UNDER_CURSOR s neprocházením volání MQGET , abyste odebrali zprávu z fronty.

Kurzor procházení není přesunut voláním funkce MQGET bez procházení, a to ani v případě, že používáte stejný popisovač *Hobj* . Nepřesunuje se ani pomocí volání procházení MQGET , které vrací kód dokončení MQCC_FAILEDnebo kód příčiny MQRC_TRUNCATED_MSG_FAILED.

Uveďte volbu MQGMO_LOCK s touto volbou, chcete-li zamknout procházenou zprávu.

Můžete uvést MQGMO_BROWSE_FIRST s jakoukoli platnou kombinací voleb MQGMO_* a MQMO_* , které řídí zpracování zpráv ve skupinách a segmentech logických zpráv.

Zadáte-li hodnotu MQGMO_LOGICAL_ORDER, budou zprávy procházeny v logickém pořadí. Pokud tuto volbu vynecháte, budou zprávy procházeny ve fyzickém pořadí. Zadáte-li MQGMO_BROWSE_FIRST, můžete přepínat mezi logickým a fyzickým pořadem. Následná MQGET volání používající MQGMO_BROWSE_NEXT procházejte frontu ve stejném pořadí jako poslední volání, které určilo MQGMO_BROWSE_FIRST pro popisovač fronty.

Správce front uchovává dvě sady informací o skupinách a segmentech pro volání MQGET . Informace o skupině a segmentu pro volání procházení jsou uchovány odděleně od informací pro volání, která odebírají zprávy z fronty. Zadáte-li MQGMO_BROWSE_FIRST, správce front ignoruje informace o skupině a segmentu pro procházení. Prochází frontu, jako by neexistovala žádná aktuální skupina a žádná aktuální logická zpráva. Pokud je volání MQGET úspěšné, kód dokončení MQCC_OK nebo MQCC_WARNING, informace o skupině a segmentu pro procházení se nastaví na hodnotu vrácené zprávy. Pokud volání selže, informace o skupině a segmentu zůstanou stejné jako před voláním.

MQGMO_BROWSE_NEXT
Přesuňte kurzor procházení na další zprávu ve frontě, která splňuje kritéria výběru uvedená ve volání MQGET . Zpráva je vrácena aplikaci, ale zůstává ve frontě.

Parametr MQGMO_BROWSE_NEXT není platný s žádnou z následujících voleb:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

Také se jedná o chybu, pokud nebyla fronta otevřena pro procházení.

MQGMO_BROWSE_NEXT se chová stejně jako MQGMO_BROWSE_FIRST, jedná-li se o první volání pro procházení fronty po otevření fronty pro procházení.

Zpráva pod kurzorem může být odebrána z fronty před vydáním dalšího volání MQGET s MQGMO_BROWSE_NEXT . Kurzor procházení logicky zůstává na pozici ve frontě, kterou zpráva zabírala, i když je tato pozice nyní prázdná.

Zprávy jsou uloženy ve frontě jedním ze dvou způsobů:

- FIFO v rámci priority (MQMDS_PRIORITY), nebo
- FIFO bez ohledu na prioritu (MQMDS_FIFO)

Atribut fronty **MsgDeliverySequence** označuje, která metoda se použije (podrobnosti viz “Atributy pro fronty” na stránce 836).

Fronta může mít hodnotu MsgDeliverySequence MQMDS_PRIORITY. Do fronty dorazí zpráva, která má vyšší prioritu než ta, na kterou momentálně ukazuje kurzor procházení. V takovém případě

nebyla během aktuálního procházení fronty s použitím parametru MQGMO_BROWSE_NEXTnalezena zpráva s vyšší prioritou. Lze jej nalézt pouze po resetování kurzoru procházení pomocí MQGMO_BROWSE_FIRSTnebo opětovným otevřením fronty.

Volbu MQGMO_MSG_UNDER_CURSOR lze v případě potřeby použít s voláním bez procházení MQGET k odebrání zprávy z fronty.

Kurzor procházení není přesunut pomocí volání MQGET bez procházení pomocí stejného popisovače Hobj .

Zadejte volbu MQGMO_LOCK s touto volbou, chcete-li zamknout procházenou zprávu.

Můžete uvést MQGMO_BROWSE_NEXT s jakoukoli platnou kombinací voleb MQGMO_* a MQMO_* , které řídí zpracování zpráv ve skupinách a segmentech logických zpráv.

Zadáte-li hodnotu MQGMO_LOGICAL_ORDER, budou zprávy procházeny v logickém pořadí. Pokud tuto volbu vynecháte, budou zprávy procházeny ve fyzickém pořadí. Zadáte-li MQGMO_BROWSE_FIRST, můžete přepínat mezi logickým a fyzickým pořadem. Následná MQGET volání používající MQGMO_BROWSE_NEXT procházejte frontu ve stejném pořadí jako poslední volání, které určilo MQGMO_BROWSE_FIRST pro popisovač fronty. Volání selže s kódem příčiny MQRC_INCONSISTENT_BROWSE , pokud není tato podmínka splněna.

Poznámka: Při používání volání MQGET k procházení za koncem skupiny zpráv věnujte zvláštní pozornost, pokud není zadán parametr MQGMO_LOGICAL_ORDER . Předpokládejme například, že poslední zpráva ve skupině předchází první zprávě ve skupině ve frontě. Použití MQGMO_BROWSE_NEXT k procházení za koncem skupiny a určení MQMO_MATCH_MSG_SEQ_NUMBER s MsgSeqNumber nastaveným na 1 vrátí první zprávu ve skupině, která již byla procházena. Tento výsledek se může vyskytnout okamžitě, nebo počet volání MQGET později, pokud existují skupiny, které zasahují. Totéž platí pro logickou zprávu, která není ve skupině.

Informace o skupině a segmentu pro volání procházení jsou uchovány odděleně od informací pro volání, která odebírají zprávy z fronty.

MQGMO_BROWSE_MSG_UNDER_CURSOR
Načtěte zprávu, na kterou ukazuje kurzor procházení, nedestruktivně, bez ohledu na volby MQMO_* uvedené v poli MatchOptions v MQGMO.

Parametr MQGMO_BROWSE_MSG_UNDER_CURSOR není platný s žádnou z následujících voleb:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

Také se jedná o chybu, pokud nebyla fronta otevřena pro procházení.

Zpráva, na kterou ukazuje kurzor procházení, je ta, která byla naposledy načtena pomocí volby MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT . Volání se nezdaří, pokud pro tuto frontu nebylo od jejího otevření zadáno žádné z těchto volání. Volání také selže, pokud byla zpráva, která byla pod kurzorem procházení, od té doby destruktivně načtena.

Pozice kurzoru procházení se tímto voláním nezmění.

Volbu MQGMO_MSG_UNDER_CURSOR lze použít s voláním non-browse MQGET k odebrání zprávy z fronty.

Kurzor procházení není přesunut voláním funkce MQGET bez procházení, a to ani v případě, že používáte stejný popisovač Hobj . Nepřesunuje se ani pomocí volání procházení MQGET , které vrací kód dokončení MQCC_FAILEDnebo kód příčiny MQRC_TRUNCATED_MSG_FAILED.

Je-li parametr MQGMO_BROWSE_MSG_UNDER_CURSOR zadán s parametrem MQGMO_LOCK:

- Pokud již existuje zamčená zpráva, musí být ta, která je pod kurzorem, aby se vrátila bez odemčení a zamykání znovu. Zpráva zůstává uzamčena.
- Pokud neexistuje žádná zamčená zpráva a pod kurzorem procházení se nachází zpráva, je uzamčena a vrácena do aplikace. Pokud pod kurzorem procházení není žádná zpráva, volání se nezdaří.

Je-li parametr MQGMO_BROWSE_MSG_UNDER_CURSOR zadán bez parametru MQGMO_LOCK:

- Pokud je již zpráva uzamčena, musí být ta, která je pod kurzorem. Zpráva se vrátí do aplikace a pak se odemkne. Vzhledem k tomu, že zpráva je nyní odemknuta, neexistuje žádná záruka, že ji lze znovu procházet nebo načíst destruktivně stejnou aplikací. Je možné, že byla načtena destruktivně jinou aplikací, která získala zprávy z fronty.
- Pokud neexistuje žádná zamknutá zpráva a pod kurzorem procházení je zpráva vrácena aplikaci. Pokud pod kurzorem procházení není žádná zpráva, volání selže.

Je-li v parametru MQGMO_COMPLETE_MSG zadána hodnota MQGMO_BROWSE_MSG_UNDER_CURSOR, musí kurzor procházení identifikovat zprávu, jejíž pole Offset v deskriptoru MQMD má hodnotu nula. Není-li tato podmínka splněna, volání se nezdaří s kódem příčiny MQRC_INVALID_MSG_UNDER_CURSOR.

Informace o skupině a segmentu pro volání procházení jsou uchovány odděleně od informací pro volání, která odebírají zprávy z fronty.

MQGMO_MSG_UNDER_CURSOR
Načtěte zprávu, na kterou ukazuje kurzor procházení, bez ohledu na volby MQMO_* uvedené v poli MatchOptions v produktu MQGMO. Zpráva je odebrána z fronty.

Zpráva, na kterou ukazuje kurzor procházení, je ta, která byla naposledy načtena pomocí volby MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT .

Je-li v parametru MQGMO_COMPLETE_MSG zadána hodnota MQGMO_MSG_UNDER_CURSOR, musí kurzor procházení identifikovat zprávu, jejíž pole Offset v deskriptoru MQMD má hodnotu nula. Není-li tato podmínka splněna, volání se nezdaří s kódem příčiny MQRC_INVALID_MSG_UNDER_CURSOR.

Tato volba není platná s žádnou z následujících voleb:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT
- MQGMO_UNLOCK

Také se jedná o chybu, pokud nebyla fronta otevřena pro procházení i pro vstup. Pokud kurzor procházení aktuálně neukazuje na zprávu, kterou lze načíst, volání MQGET vrátí chybu.

MQGMO_MARK_BROWSE_HANDLE
Zpráva, která je vrácena úspěšným MQGETnebo identifikována vráceným MsgToken, je označena. Značka je specifická pro popisovač objektu použitý ve volání.

Zpráva není odebrána z fronty.

Parametr MQGMO_MARK_BROWSE_HANDLE je platný pouze v případě, že je zadána jedna z následujících voleb:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT

Parametr MQGMO_MARK_BROWSE_HANDLE není platný s žádnou z následujících voleb:

- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK

- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

Zpráva zůstává v tomto stavu, dokud nedojde k jedné z následujících událostí:

- Příslušná rukojeť objektu je uzavřena, buď normálně, nebo jinak.
- Zpráva není pro tento popisovač označena voláním funkce MQGET s volbou MQGMO_UNMARK_BROWSE_HANDLE.
- Zpráva je vrácena z volání na destruktivní MQGET, které je dokončeno s MQCC_OK nebo MQCC_WARNING. Stav zprávy zůstane změněn, i když je MQGET později odvolán.
- Platnost zprávy vyprší.

MQGMO_MARK_BROWSE_CO_OP
Zpráva vrácená úspěšným produktem MQGETnebo identifikovaná vráceným produktem *MsgToken*je označena pro všechny manipulátory ve spolupracující sadě.

Kooperativní značka úrovně je navíc k jakékoli značce úrovně popisovače, která by mohla být nastavena.

Zpráva není odebrána z fronty.

Parametr MQGMO_MARK_BROWSE_CO_OP je platný pouze v případě, že použitý popisovač objektu byl vrácen voláním metody MQOPEN , která byla zadána MQOO_CO_OP. Musíte také uvést jednu z následujících voleb MQGMO :

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT

Tato volba není platná s žádnou z následujících voleb:

- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK
- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

Pokud je zpráva již označena a volba MQGMO_UNMARKED_BROWSE_MSG není uvedena, volání se nezdaří s kódem příčiny MQCC_FAILED a kódem příčiny MQRC_MSG_MARKED_BROWSE_CO_OP.

Zpráva zůstává v tomto stavu, dokud nedojde k jedné z následujících událostí:

- Všechny manipulátory objektů ve spolupracující sadě jsou uzavřeny.
- Zpráva není označena pro spolupracující prohlížeče voláním funkce MQGET s volbou MQGMO_UNMARK_BROWSE_CO_OP.
- Správce front automaticky zruší označení zprávy.
- Zpráva je vrácena z volání na neprocházení MQGET. Stav zprávy zůstane změněn, i když je MQGET později odvolán.
- Platnost zprávy vyprší.

MQGMO_UNMARKED_BROWSE_MSG
Volání MQGET , které uvádí MQGMO_UNMARKED_BROWSE_MSG , vrátí zprávu, která je považována za neoznačenou pro svůj popisovač. Nevrátí zprávu, pokud byla zpráva označena pro svůj popisovač. Také nevrátí zprávu, pokud byla fronta otevřena voláním funkce MQOPENs volbou MQOO_CO_OPa zpráva byla označena členem spolupracující sady.

Tato volba není platná s žádnou z následujících voleb:

- MQGMO_ALL_MSGS_AVAILABLE

- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK
- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

MQGMO_UNMARK_BROWSE_CO_OP
Po volání MQGET , které uvádí tuto volbu, již není zpráva brána v úvahu pro žádné otevřené popisovače v sadě spolupracujících popisovačů, které mají být označeny pro spolupracující sadu. Zpráva je stále považována za označenou na úrovni popisovače, pokud byla označena na úrovni popisovače před tímto voláním.

Použití parametru MQGMO_UNMARK_BROWSE_CO_OP je platné pouze s popisovačem vráceným úspěšným voláním funkce MQOPEN s volbou MQOO_CO_OP. MQGET uspěje i v případě, že zpráva není považována za označenou spolupracující sadou popisovačů.

Volba MQGMO_UNMARK_BROWSE_CO_OP není platná při volání MQGET bez procházení nebo s některou z následujících voleb:

- 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
Po volání MQGET , které uvádí tuto volbu, se vyhledaná zpráva již nepovažuje za označenou tímto popisovačem.

Volání je úspěšné i v případě, že zpráva není pro tento manipulátor označena.

Tato volba není platná při volání MQGET bez procházení nebo s některou z následujících voleb:

- 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

Volby zámku

Následující volby se týkají zamykání zpráv ve frontě:

MQGMO_LOCK
Zamkněte procházenou zprávu, aby se tato zpráva stala neviditelnou pro jakýkoli jiný manipulátor otevřený pro danou frontu. Volbu lze zadat pouze v případě, že je zadána také jedna z následujících voleb:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT

- MQGMO_BROWSE_MSG_UNDER_CURSOR

Pro každý popisovač fronty lze uzamknout pouze jednu zprávu. Zpráva může být logická nebo fyzická:

- Zadáte-li hodnotu MQGMO_COMPLETE_MSG, všechny segmenty zpráv, které tvoří logickou zprávu, budou uzamčeny pro manipulátor fronty. Všechny zprávy musí být ve frontě a musí být k dispozici pro načtení.
- Pokud vynecháte MQGMO_COMPLETE_MSG, pro manipulátor fronty se uzamkne pouze jedna fyzická zpráva. Pokud se tato zpráva stane segmentem logické zprávy, uzamčený segment zabrání ostatním aplikacím, aby pomocí produktu MQGMO_COMPLETE_MSG načetli nebo procházeli logickou zprávu.

Zamknutá zpráva je vždy ta pod kurzorem procházení. Zprávu lze odebrat z fronty pozdějším voláním funkce MQGET , které uvádí volbu MQGMO_MSG_UNDER_CURSOR . Jiná volání MQGET používající popisovač fronty mohou také odebrat zprávu (například volání, které uvádí identifikátor zprávy zamknuté zprávy).

Pokud volání vrátí kód dokončení MQCC_FAILEDnebo MQCC_WARNING s kódem příčiny MQRC_TRUNCATED_MSG_FAILED, žádná zpráva se nezamkne.

Pokud aplikace neodebere zprávu z fronty, zámek se uvolní jednou z následujících akcí:

- Zadejte další volání MQGET pro tento popisovač a zadejte buď MQGMO_BROWSE_FIRST , nebo MQGMO_BROWSE_NEXT. Zámek se uvolní, pokud se volání dokončí s MQCC_OK nebo MQCC_WARNING. Zpráva zůstane uzamčena, pokud se volání dokončí s MQCC_FAILED. Platí však následující výjimky:
  - Zpráva není odemknuta, pokud je vrácena hodnota MQCC_WARNING s MQRC_TRUNCATED_MSG_FAILED.
  - Zpráva je odemknuta, pokud je vrácena hodnota MQCC_FAILED s MQRC_NO_MSG_AVAILABLE.

  Zadáte-li také hodnotu MQGMO_LOCK, vrácená zpráva bude uzamčena. Pokud vynecháte MQGMO_LOCK, po volání se neobjeví žádná zamknutá zpráva.

  Pokud uvedete MQGMO_WAITa žádná zpráva není okamžitě k dispozici, původní zpráva se odemkne před začátkem čekání.
- Zadejte další volání MQGET pro tento manipulátor s volbou MQGMO_BROWSE_MSG_UNDER_CURSORbez volání MQGMO_LOCK. Zámek se uvolní, pokud se volání dokončí s MQCC_OK nebo MQCC_WARNING. Zpráva zůstane uzamčena, pokud se volání dokončí s MQCC_FAILED. Platí však následující výjimka:
  - Zpráva není odemknuta, pokud je vrácena hodnota MQCC_WARNING s MQRC_TRUNCATED_MSG_FAILED.
- Zadejte další volání MQGET pro tento popisovač s MQGMO_UNLOCK.
- Zadání volání MQCLOSE pomocí popisovače. MQCLOSE může být implicitní, způsobené ukončením aplikace.

K určení volby MQGMO_LOCK, jiné než MQOO_BROWSE, která je potřebná k určení doprovodné volby procházení, není požadována žádná speciální volba MQOPEN .

Parametr MQGMO_LOCK není platný s žádnou z následujících voleb:

- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

MQGMO_UNLOCK
Zpráva, která má být odemknuta, musí být dříve uzamčena voláním MQGET s volbou MQGMO_LOCK . Pokud pro tento popisovač není uzamčena žádná zpráva, volání se dokončí s MQCC_WARNING a MQRC_NO_MSG_LOCKED.

Parametry **MsgDesc**, **BufferLength**, **Buffer**a **DataLength** nejsou kontrolovány ani pozměněny, pokud zadáte MQGMO_UNLOCK. V souboru *Buffer*není vrácena žádná zpráva.

Pro zadání MQGMO_UNLOCK není požadována žádná speciální volba otevření (ačkoli MQOO_BROWSE je potřeba pro vydání požadavku na zámek na prvním místě).

Tato volba není platná s žádnými volbami kromě následujících:

- MQGMO_NO_WAIT
- MQGMO_NO_SYNCPOINT

Obě tyto volby se předpokládají bez ohledu na to, zda jsou zadány, či nikoli.

Volby dat zpráv

Následující volby se vztahují ke zpracování dat zprávy při čtení zprávy z fronty:

MQGMO_ACCEPT_TRUNCATED_MSG
Pokud je vyrovnávací paměť zpráv příliš malá na to, aby mohla obsahovat celou zprávu, povolte volání MQGET , aby vyrovnávací paměť vyplňovalo. MQGET vyplní vyrovnávací paměť co nejvíce zpráv. Vydá varovný kód dokončení a dokončí zpracování. To znamená, že:

- Při procházení zpráv je kurzor procházení rozšířen na vrácenou zprávu.
- Při odebírání zpráv je vrácená zpráva odebrána z fronty.
- Kód příčiny MQRC_TRUNCATED_MSG_ACCEPTED je vrácen, pokud nedojde k žádné jiné chybě.

Bez této volby je vyrovnávací paměť stále naplněna tolik zprávy, kolik může zadržet. Je vydán varovný kód dokončení, ale zpracování není dokončeno. To znamená, že:

- Při procházení zpráv není kurzor procházení rozšířen.
- Při odebírání zpráv není zpráva odebrána z fronty.
- Pokud nedojde k žádné jiné chybě, vrátí se kód příčiny MQRC_TRUNCATED_MSG_FAILED .

MQGMO_CONVERT
Tato volba převede data aplikace ve zprávě tak, aby byla v souladu s hodnotami CodedCharSetId a Encoding uvedenými v parametru **MsgDesc** volání MQGET . Data se převedou před jejich zkopírováním do parametru **Buffer** .

Pole Format zadané při vložení zprávy předpokládá proces převodu, aby identifikoval povahu dat ve zprávě. Data zprávy jsou převedena správcem front pro vestavěné formáty a uživatelskou procedurou pro jiné formáty. Podrobnosti o uživatelské proceduře pro převod dat viz “Uživatelská procedura převodu dat” na stránce 905 .

- Pokud je převod úspěšný, pole CodedCharSetId a Encoding uvedená v parametru **MsgDesc** se při návratu z volání MQGET nezmění.
- Pokud se nezdaří pouze převod, data zprávy se vrátí nepřevedená. Pole CodedCharSetId a Encoding v souboru MsgDesc jsou nastavena na hodnoty pro nepřevedenou zprávu. V tomto případě je kód dokončení MQCC_WARNING .

V obou případech tato pole popisují identifikátor znakové sady a kódování dat zprávy vrácených v parametru **Buffer** .

Seznam názvů formátů, pro které správce front provádí převod, naleznete v poli *Format* popsaném v tématu “MQMD-Deskriptor zpráv” na stránce 424 .

Volby skupin a segmentů

Následující volby se týkají zpracování zpráv ve skupinách a segmentech logických zpráv. Před popisy voleb jsou uvedeny některé definice důležitých pojmů:

Fyzická zpráva
Fyzická zpráva je nejmenší jednotka informací, kterou lze umístit do fronty nebo z ní odebrat. Často odpovídá informacím uvedeným nebo načteným v jediném volání MQPUT, MQPUT1nebo MQGET . Každá fyzická zpráva má svůj vlastní deskriptor zprávy MQMD. Fyzické zprávy se obvykle rozlišují podle

různých hodnot pro identifikátor zprávy, což je pole MsgId v databázi MQMD. Správce front nevynucuje různé hodnoty.

Logická zpráva
Logická zpráva je jedna jednotka informací o aplikaci. Při absenci systémových omezení je logická zpráva stejná jako fyzická zpráva. Pokud jsou logické zprávy velké, systémová omezení mohou způsobit, že je vhodné nebo nezbytné rozdělit logickou zprávu do dvou nebo více fyzických zpráv, nazývaných segmenty.

Logická zpráva, která byla segmentována, se skládá ze dvou nebo více fyzických zpráv, které mají stejný nenulový identifikátor skupiny GroupId v poli MQMD. Mají stejné pořadové číslo zprávy, pole MsgSeqNumber v poli MQMD. Segmenty se rozlišují podle různých hodnot pro posunutí segmentu, pole Offset v produktu MQMD. Posunutí segmentu je posunutí dat ve fyzické zprávě od začátku dat v logické zprávě. Protože každý segment je fyzickou zprávou, segmenty v logické zprávě mají obvykle různé identifikátory zpráv.

Logická zpráva, která nebyla segmentována, ale pro kterou byla segmentace povolena odesílající aplikací, má také nenulový identifikátor skupiny. V tomto případě existuje pouze jedna fyzická zpráva s tímto identifikátorem skupiny, pokud logická zpráva nepatří do skupiny zpráv. Logické zprávy, pro které byla segmentace blokována odesílající aplikací, mají identifikátor skupiny s hodnotou null, MQGI_NONE, pokud logická zpráva nepatří do skupiny zpráv.

Skupina zpráv
Skupina zpráv je sada jedné nebo více logických zpráv, které mají stejný nenulový identifikátor skupiny. Logické zprávy ve skupině jsou odlišeny různými hodnotami pro pořadové číslo zprávy. Pořadové číslo je celé číslo v rozsahu 1 až n, kde n je počet logických zpráv ve skupině. Je-li jedna nebo více logických zpráv segmentováno, je ve skupině více než n fyzických zpráv.

MQGMO_LOGICAL_ORDER
MQGMO_LOGICAL_ORDER řídí pořadí, ve kterém jsou zprávy vraceny po sobě jdoucími MQGET voláními pro popisovač fronty. Volba musí být uvedena pro každé volání.

Pokud je parametr MQGMO_LOGICAL_ORDER uveden pro následná volání MQGET pro stejný popisovač fronty, zprávy ve skupinách se vrátí v pořadí podle jejich pořadových čísel zpráv. Segmenty logických zpráv jsou vráceny v pořadí daném jejich offsety segmentů. Toto pořadí se může lišit od pořadí, ve kterém se tyto zprávy a segmenty vyskytují ve frontě.

Poznámka: Uvedení parametru MQGMO_LOGICAL_ORDER nemá žádné nepříznivé důsledky na zprávy, které nepatří do skupin a nejsou segmenty. Ve skutečnosti se s takovými zprávami zachází tak, jako by každá z nich patřila do skupiny zpráv, která se skládá pouze z jedné zprávy. Je bezpečné uvést MQGMO_LOGICAL_ORDER , když načítáte zprávy z front, které obsahují směs zpráv ve skupinách, segmentech zpráv a nesegmentovaných zprávách, které nejsou ve skupinách.

Chcete-li vrátit zprávy v požadovaném pořadí, správce front uchová informace o skupině a segmentu mezi následnými voláními MQGET . Informace o skupině a segmentu identifikují aktuální skupinu zpráv a aktuální logickou zprávu pro popisovač fronty. Také identifikuje aktuální pozici ve skupině a logickou zprávu, a zda jsou zprávy načítány v rámci pracovní jednotky. Vzhledem k tomu, že správce front tyto informace uchovává, nemusí aplikace před každým voláním funkce MQGET nastavit informace o skupině a segmentu. Konkrétně to znamená, že aplikace nemusí nastavit pole GroupId, MsgSeqNumbera Offset v MQMD. Aplikace však musí při každém volání správně nastavit volbu MQGMO_SYNCPOINT nebo MQGMO_NO_SYNCPOINT .

Když je fronta otevřena, neexistuje žádná aktuální skupina zpráv a žádná aktuální logická zpráva. Skupina zpráv se stane aktuální skupinou zpráv, když volání MQGET vrátí zprávu, která má příznak MQMF_MSG_IN_GROUP . Při zadání parametru MQGMO_LOGICAL_ORDER při následných voláních zůstává tato skupina aktuální skupinou, dokud se nevrátí zpráva, která má:

- MQMF_LAST_MSG_IN_GROUP bez MQMF_SEGMENT (to znamená, že poslední logická zpráva ve skupině není segmentovaná), nebo
- MQMF_LAST_MSG_IN_GROUP s MQMF_LAST_SEGMENT (to znamená, že vrácená zpráva je posledním segmentem poslední logické zprávy ve skupině).

Když je taková zpráva vrácena, skupina zpráv je ukončena a po úspěšném dokončení volání MQGET již není aktuální skupina. Podobným způsobem se logická zpráva stane aktuální logickou zprávou, když volání funkce MQGET vrátí zprávu s příznakem MQMF_SEGMENT . Logická zpráva je ukončena, když je vrácena zpráva, která má příznak MQMF_LAST_SEGMENT .

Nejsou-li uvedena žádná kritéria výběru, následná volání MQGET vrátí zprávy pro první skupinu zpráv ve frontě ve správném pořadí. Pak vrátí zprávy pro druhou skupinu zpráv a tak dále, dokud nebudou k dispozici žádné další zprávy. Je možné vybrat konkrétní vrácené skupiny zpráv uvedením jedné nebo více následujících voleb v poli MatchOptions :

- MQMO_MATCH_MSG_ID
- MQMO_MATCH_CORREL_ID
- MQMO_MATCH_GROUP_ID

Tyto volby jsou však platné pouze v případě, že neexistuje žádná aktuální skupina zpráv nebo logická zpráva. Další podrobnosti viz pole MatchOptions popsané v části “MQGMO-Volby získání zprávy” na stránce 371 .

Tabulka 495 na stránce 396 zobrazuje hodnoty polí MsgId, CorrelId, GroupId, MsgSeqNumbera Offset , které správce front hledá při pokusu o nalezení zprávy, která má být vrácena při volání MQGET . Pravidla platí jak pro odebrání zpráv z fronty, tak pro procházení zpráv ve frontě. V tabulce buď znamená Ano, nebo Ne:

LOG ORD
Označuje, zda je ve volání uvedena volba MQGMO_LOGICAL_ORDER .

Cur grp
Označuje, zda před voláním existuje aktuální skupina zpráv.

Cur log msg
Označuje, zda před voláním existuje aktuální logická zpráva.

Ostatní sloupce
Zobrazit hodnoty, které správce front hledá. Předchozí označuje hodnotu vrácenou pro pole v předchozí zprávě pro popisovač fronty.

Tabulka 495. Volby MQGET související se zprávami ve skupinách a segmentech logických zpráv

Volby, které zadáte	**Stav skupiny a zprávy protokolu před voláním**		**Hodnoty, které správce front hledá**				
LOG ORD	**Cur grp**	**Cur log msg**	*MsgId*	*CorrelId*	*GroupId*	*MsgSeqNumber*	*Offset*
Ano	Ne	Ne	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*	1	0
Ano	Ne	Ano	Libovolný identifikátor zprávy	Libovolný identifikátor korelace	Předchozí identifikátor skupiny	1	Předchozí offset + předchozí délka segmentu
Ano	Ano	Ne	Libovolný identifikátor zprávy	Libovolný identifikátor korelace	Předchozí identifikátor skupiny	Předchozí pořadové číslo + 1	0

Tabulka 495. Volby MQGET související se zprávami ve skupinách a segmentech logických zpráv (pokračování)							
Volby, které zadáte	**Stav skupiny a zprávy protokolu před voláním**		**Hodnoty, které správce front hledá**				
Ano	Ano	Ano	Libovolný identifikátor zprávy	Libovolný identifikátor korelace	Předchozí identifikátor skupiny	Předchozí pořadové číslo	Předchozí offset + předchozí délka segmentu
Ne	buď	buď	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*

Pokud je ve frontě přítomno více skupin zpráv a jsou vhodné pro návrat, vrátí se skupiny v pořadí určeném pozicí ve frontě prvního segmentu první logické zprávy v každé skupině. To znamená, že fyzické zprávy, které mají pořadová čísla 1, a posuny 0, určují pořadí, ve kterém jsou vráceny vhodné skupiny.

Volba MQGMO_LOGICAL_ORDER ovlivňuje jednotky práce takto:

- Pokud je první logická zpráva nebo segment ve skupině načten v rámci pracovní jednotky, všechny ostatní logické zprávy a segmenty ve skupině musí být načteny v rámci pracovní jednotky, pokud je použit stejný manipulátor fronty. Nemusí však být načteny v rámci stejné pracovní jednotky. To umožňuje rozdělit skupinu zpráv skládající se z mnoha fyzických zpráv na dvě nebo více po sobě jdoucích jednotek práce pro popisovač fronty.
- Není-li první logická zpráva nebo segment ve skupině načten v rámci pracovní jednotky a je-li použit stejný manipulátor fronty, nelze v rámci pracovní jednotky načíst žádné další logické zprávy a segmenty ve skupině.

Nejsou-li tyto podmínky splněny, volání MQGET selže s kódem příčiny MQRC_INCONSISTENT_UOW.

Je-li zadána volba MQGMO_LOGICAL_ORDER , nesmí být hodnota MQGMO zadaná ve volání MQGET menší než hodnota MQGMO_VERSION_2a hodnota MQMD nesmí být menší než hodnota MQMD_VERSION_2. Není-li tato podmínka splněna, volání selže s kódem příčiny MQRC_WRONG_GMO_VERSION nebo MQRC_WRONG_MD_VERSION, podle potřeby.

Není-li pro následná volání MQGET pro popisovač fronty zadána hodnota MQGMO_LOGICAL_ORDER , budou zprávy vráceny bez ohledu na to, zda patří do skupin zpráv, nebo zda se jedná o segmenty logických zpráv. To znamená, že zprávy nebo segmenty z konkrétní skupiny nebo logické zprávy mohou být vráceny mimo pořadí, nebo mohou být promíchány se zprávami nebo segmenty z jiných skupin nebo logických zpráv, nebo se zprávami, které nejsou ve skupinách a nejsou segmenty. V této situaci jsou konkrétní zprávy, které jsou vráceny následnými voláními MQGET , řízeny volbami MQMO_* uvedenými na těchto voláních (podrobnosti o těchto volbách viz pole *MatchOptions* popsané v části “MQGMO-Volby získání zprávy” na stránce 371).

Jedná se o techniku, kterou lze použít k restartování skupiny zpráv nebo logické zprávy uprostřed po selhání systému. Když se systém restartuje, aplikace může nastavit pole GroupId, MsgSeqNumber, Offseta MatchOptions na odpovídající hodnoty a poté zadat volání MQGET se sadou MQGMO_SYNCPOINT nebo MQGMO_NO_SYNCPOINT , ale bez uvedení MQGMO_LOGICAL_ORDER. Je-li toto volání úspěšné, správce front zachová informace o skupině a segmentu a následná volání MQGET používající tento manipulátor fronty mohou určit hodnotu MQGMO_LOGICAL_ORDER jako normální.

Informace o skupině a segmentu, které správce front uchovává pro volání MQGET , jsou odděleny od informací o skupině a segmentu, které uchovává pro volání MQPUT. Kromě toho správce front uchovává samostatné informace pro:

- Volání produktu MQGET , která odebírají zprávy z fronty.
- Volání MQGET , která procházejí zprávy ve frontě.

Pro jakýkoli daný popisovač fronty může aplikace kombinovat volání MQGET , která uvádějí MQGMO_LOGICAL_ORDER s MQGET voláními, která neuvádějí. Všimněte si však následujících bodů:

- Vynecháte-li MQGMO_LOGICAL_ORDER, každé úspěšné volání MQGET způsobí, že správce front nastaví informace o uložené skupině a segmentu na hodnoty odpovídající vrácené zprávě. Tato volba nahradí existující informace o skupině a segmentu uchovávané správcem front pro manipulátor fronty. Změní se pouze informace odpovídající akci volání (procházet nebo odebrat).
- Pokud vynecháte MQGMO_LOGICAL_ORDER, volání neselže, pokud existuje aktuální skupina zpráv nebo logická zpráva; volání může být úspěšné s kódem dokončení MQCC_WARNING . Tabulka 496 na stránce 398 zobrazuje různé případy, které mohou nastat. V těchto případech, pokud kód dokončení není MQCC_OK, kód příčiny je jeden z následujících (podle potřeby):
  - MQRC_INCOMPLETE_GROUP
  - MQRC_INCOMPLETE_MSG
  - MQRC_INCONSISTENT_UOW

  Poznámka: Správce front nekontroluje informace o skupině a segmentu při procházení fronty nebo při zavírání fronty, která byla otevřena pro procházení, ale nikoli pro vstup; v těchto případech je kód dokončení vždy MQCC_OK (bez dalších chyb).

Tabulka 496. Výsledek, když volání MQGET nebo MQCLOSE není konzistentní s informacemi o skupině a segmentu

Aktuální volání je	Předchozí volání bylo MQGET s MQGMO_LOGICAL_ORDER	Předchozí volání bylo MQGET bez MQGMO_LOGICAL_ORDER
Produkt MQGET s produktem MQGMO_LOGICAL_ORDER	MQCC_FAILED	MQCC_FAILED
MQGET bez MQGMO_LOGICAL_ORDER	MQCC_WARNING	MQCC_OK
MQCLOSE s neukončenou skupinou nebo logickou zprávou	MQCC_WARNING	MQCC_OK

Aplikacím, které chtějí načíst zprávy a segmenty v logickém pořadí, se doporučuje zadat MQGMO_LOGICAL_ORDER, protože se jedná o nejjednodušší volbu, kterou lze použít. Tato volba zbavuje aplikaci potřeby spravovat informace o skupině a segmentu, protože správce front tyto informace spravuje. Specializované aplikace však mohou vyžadovat větší kontrolu, než jakou poskytuje volba MQGMO_LOGICAL_ORDER , a toho lze dosáhnout neurčením této volby. Aplikace se pak musí ujistit, že pole MsgId, CorrelId, GroupId, MsgSeqNumbera Offset v MQMDa volby MQMO_* v MatchOptions v MQGMOjsou správně nastaveny před každým voláním MQGET .

Například aplikace, která chce předávat přijaté fyzické zprávy bez ohledu na to, zda jsou tyto zprávy ve skupinách nebo segmentech logických zpráv, nesmí uvádět MQGMO_LOGICAL_ORDER. V komplexní síti s více cestami mezi odesílajícími a přijímajícími správci front mohou fyzické zprávy přicházet mimo pořadí. Zadáním parametru MQGMO_LOGICAL_ORDERani odpovídajícího parametru MQPMO_LOGICAL_ORDER ve volání MQPUT může předávající aplikace načíst a předat každou fyzickou zprávu ihned po jejím přijetí, aniž by musela čekat na další zprávu v logickém pořadí, aby byla doručena.

Můžete uvést MQGMO_LOGICAL_ORDER s jakoukoli jinou volbou MQGMO_* a s různými volbami MQMO_* za příslušných okolností (viz předchozí část).

- z/OS V systému z/OSje tato volba podporována pro soukromé a sdílené fronty, ale fronta musí mít typ indexu MQIT_GROUP_ID. Pro sdílené fronty musí být objekt CFSTRUCT, na který se fronta mapuje, na úrovni CFLEVEL (3) nebo vyšší.
- Tato volba je podporována pro všechny lokální fronty na následujících platformách:

– 

AIX

– 

Linux

– 

IBM i

– 

Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

MQGMO_COMPLETE_MSG
Volání MQGET může vrátit pouze úplnou logickou zprávu. Je-li logická zpráva segmentována, správce front znovu sestaví segmenty a vrátí aplikaci úplnou logickou zprávu. Skutečnost, že logická zpráva byla segmentována, není pro aplikaci, která ji načítá, zřejmá.

Poznámka: Jedná se o jedinou volbu, která způsobí, že správce front znovu sestaví segmenty zpráv. Není-li uveden, segmenty jsou vráceny aplikaci jednotlivě, pokud jsou přítomny ve frontě (a splňují ostatní kritéria výběru uvedená ve volání MQGET). Aplikace, které nechtějí přijímat jednotlivé segmenty, musí vždy uvádět MQGMO_COMPLETE_MSG.

Chcete-li použít tuto volbu, musí aplikace poskytnout vyrovnávací paměť, která je dostatečně velká, aby pojmula celou zprávu, nebo musí zadat volbu MQGMO_ACCEPT_TRUNCATED_MSG .

Pokud fronta obsahuje segmentované zprávy s chybějícími některými segmenty (například proto, že byly zpožděny v síti a dosud nebyly přijaty), zadání parametru MQGMO_COMPLETE_MSG zabrání načtení segmentů, které patří k neúplným logickým zprávám. Tyto segmenty zpráv však stále přispívají k hodnotě atributu fronty **CurrentQDepth** ; to znamená, že nemusí existovat žádné načítatelné logické zprávy, i když je hodnota *CurrentQDepth* větší než nula.

V případě trvalých zpráv může správce front znovu sestavit segmenty pouze v rámci pracovní jednotky:

- Pokud volání MQGET pracuje v rámci uživatelem definované jednotky práce, použije se tato jednotka práce. Dojde-li během procesu opětovného sestavení k selhání volání, správce front ve frontě obnoví všechny segmenty, které byly během opětovného sestavení odebrány. Selhání však nebrání úspěšnému potvrzení transakce.
- Pokud volání pracuje mimo uživatelem definovanou jednotku práce a neexistuje žádná uživatelem definovaná jednotka práce, správce front vytvoří jednotku práce po dobu trvání volání. Je-li volání úspěšné, správce front potvrdí transakci automaticky (aplikace to nemusí provést). Pokud volání selže, správce front odvolá transakci.
- Pokud volání pracuje mimo uživatelem definovanou jednotku práce, ale existuje uživatelem definovaná jednotka práce, nemůže správce front znovu sestavit. Pokud zpráva nevyžaduje opětovné sestavení, volání může být stále úspěšné. Pokud však zpráva vyžaduje opětovné sestavení, volání selže s kódem příčiny MQRC_UOW_NOT_AVAILABLE.

Pro přechodné zprávy správce front nevyžaduje, aby byla k dispozici jednotka práce pro opětovné sestavení.

Každá fyzická zpráva, která je segmentem, má svůj vlastní deskriptor zprávy. Pro segmenty tvořící jedinou logickou zprávu je většina polí v deskriptoru zprávy stejná pro všechny segmenty v logické zprávě; obvykle se liší pouze pole MsgId, Offseta MsgFlags v jednotlivých segmentech logické zprávy. Pokud je však segment umístěn do fronty nedoručených zpráv ve zprostředkujícím správci front, obslužná rutina DLQ načte zprávu s uvedením volby MQGMO_CONVERT , což může vést ke změně znakové sady nebo kódování segmentu. Pokud obslužná rutina DLQ úspěšně odešle segment po cestě, může mít segment znakovou sadu nebo kódování, které se liší od ostatních segmentů v logické zprávě, když segment dorazí do cílového správce front.

Logickou zprávu sestávající ze segmentů, v nichž se pole CodedCharSetId a Encoding liší, nemůže správce front znovu sestavit do jediné logické zprávy. Místo toho správce front znovu sestaví a vrátí prvních několik po sobě jdoucích segmentů na začátku logické zprávy, které mají stejné identifikátory znakové sady a kódování, a volání MQGET se dokončí s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_INCONSISTENT_CCSIDS nebo MQRC_INCONSISTENT_ENCODINGS, podle potřeby.

K tomu dochází bez ohledu na to, zda je zadán parametr MQGMO_CONVERT . Chcete-li načíst zbývající segmenty, musí aplikace znovu zadat volání MQGET bez volby MQGMO_COMPLETE_MSG a načíst segmenty jeden po druhém. MQGMO_LOGICAL_ORDER lze použít k načtení zbývajících segmentů v pořadí.

Aplikace, která vkládá segmenty, může také nastavit jiná pole v deskriptoru zprávy na hodnoty, které se v jednotlivých segmentech liší. Pokud však přijímající aplikace používá k načtení logické zprávy příkaz MQGMO_COMPLETE_MSG , nemá to žádnou výhodu. Když správce front znovu sestaví logickou zprávu, vrátí v deskriptoru zprávy hodnoty z deskriptoru zprávy pro první segment; jedinou výjimkou je pole MsgFlags , které správce front nastaví tak, aby označilo, že znovu sestavená zpráva je jediným segmentem.

Je-li pro zprávu sestavy zadána hodnota MQGMO_COMPLETE_MSG , provede správce front speciální zpracování. Správce front zkontroluje frontu, aby zjistil, zda se ve frontě nacházejí všechny zprávy sestavy daného typu týkající se různých segmentů v logické zprávě. Pokud ano, lze je načíst jako jednu zprávu zadáním MQGMO_COMPLETE_MSG. Aby to bylo možné, zprávy sestavy musí být generovány správcem front nebo agentem MCA, který podporuje segmentaci, nebo musí původní aplikace požadovat alespoň 100 bajtů dat zprávy (tj. musí být zadány příslušné volby MQRO_*_WITH_DATA nebo MQRO_*_WITH_FULL_DATA). Pokud je pro segment přítomno méně dat aplikace, chybějící bajty jsou nahrazeny hodnotami null ve vrácené zprávě sestavy.

Je-li parametr MQGMO_COMPLETE_MSG zadán spolu s parametrem MQGMO_MSG_UNDER_CURSOR nebo MQGMO_BROWSE_MSG_UNDER_CURSOR, musí být kurzor procházení umístěn na zprávě, jejíž pole *Offset* v poli MQMD má hodnotu 0. Není-li tato podmínka splněna, volání se nezdaří s kódem příčiny MQRC_INVALID_MSG_UNDER_CURSOR.

MQGMO_COMPLETE_MSG implikuje MQGMO_ALL_SEGMENTS_AVAILABLE, které proto nemusí být specifikovány.

Volbu MQGMO_COMPLETE_MSG lze zadat s jakoukoli jinou volbou MQGMO_* kromě volby MQGMO_SYNCPOINT_IF_PERSISTENTa s libovolnou z voleb MQMO_* kromě volby MQMO_MATCH_OFFSET.

- z/OS V systému z/OSje tato volba podporována pro soukromé a sdílené fronty, ale fronta musí mít typ indexu MQIT_GROUP_ID. Pro sdílené fronty platí, že objekt CFSTRUCT, na který má být fronta mapována, musí být na úrovni CFLEVEL (3) nebo vyšší.
- Na následujících platformách:
  - AIX AIX
  - IBM i IBM i
  - Linux Linux
  - Windows Windows

  a pro produkt IBM MQ MQI clients připojený k těmto systémům je tato volba podporována pro všechny lokální fronty.

MQGMO_ALL_MSGS_AVAILABLE
Zprávy ve skupině budou k dispozici pro načtení pouze v případě, že jsou k dispozici všechny zprávy ve skupině. Pokud fronta obsahuje skupiny zpráv s některými chybějícími zprávami (možná proto, že byly zpožděny v síti a ještě nebyly přijaty), uvedení MQGMO_ALL_MSGS_AVAILABLE zabrání načtení zpráv, které patří do neúplných skupin. Tyto zprávy však stále přispívají k hodnotě atributu fronty **CurrentQDepth** ; to znamená, že nemusí existovat žádné načítatelné skupiny zpráv, i když je hodnota CurrentQDepth větší než nula. Pokud nejsou k dispozici žádné další zprávy, které by bylo možné načíst, kód příčiny MQRC_NO_MSG_AVAILABLE se vrátí po uplynutí určeného intervalu čekání (pokud existuje).

Zpracování MQGMO_ALL_MSGS_AVAILABLE závisí na tom, zda je zadána také hodnota MQGMO_LOGICAL_ORDER :

- Jsou-li zadány obě volby, má parametr MQGMO_ALL_MSGS_AVAILABLE účinek pouze v případě, že neexistuje žádná aktuální skupina nebo logická zpráva. Pokud existuje aktuální skupina nebo logická zpráva, MQGMO_ALL_MSGS_AVAILABLE se ignoruje. To znamená, že MQGMO_ALL_MSGS_AVAILABLE může zůstat zapnutý při zpracování zpráv v logickém pořadí.
- Je-li parametr MQGMO_ALL_MSGS_AVAILABLE zadán bez parametru MQGMO_LOGICAL_ORDER, má parametr MQGMO_ALL_MSGS_AVAILABLE vždy nějaký účinek. To znamená, že volba musí být vypnuta po odebrání první zprávy ve skupině z fronty, aby bylo možné odebrat zbývající zprávy ve skupině.

Úspěšné dokončení MQGET volání uvádějící MQGMO_ALL_MSGS_AVAILABLE znamená, že v době, kdy bylo volání MQGET vydáno, byly všechny zprávy ve skupině ve frontě. Mějte však na paměti, že ostatní aplikace mohou i nadále odebírat zprávy ze skupiny (skupina není uzamčena pro aplikaci, která načte první zprávu ve skupině).

Pokud tuto volbu vynecháte, zprávy patřící do skupin lze načíst i v případě, že je skupina neúplná.

MQGMO_ALL_MSGS_AVAILABLE implikuje MQGMO_ALL_SEGMENTS_AVAILABLE, které proto nemusí být specifikovány.

Parametr MQGMO_ALL_MSGS_AVAILABLE lze zadat s jakoukoli jinou volbou MQGMO_* a s libovolnou z voleb MQMO_* .

- z/OS V systému z/OSje tato volba podporována pro soukromé a sdílené fronty, ale fronta musí mít typ indexu MQIT_GROUP_ID. Pro sdílené fronty platí, že objekt CFSTRUCT, na který má být fronta mapována, musí být na úrovni CFLEVEL (3) nebo vyšší.
- Na následujících platformách:
  - AIX AIX
  - IBM i IBM i
  - Linux Linux
  - Windows Windows

  a pro produkt IBM MQ MQI clients připojený k těmto systémům je tato volba podporována pro všechny lokální fronty.

MQGMO_ALL_SEGMENTS_AVAILABLE
Segmenty v logické zprávě budou k dispozici pro načtení pouze v případě, že jsou k dispozici všechny segmenty v logické zprávě. Pokud fronta obsahuje segmentované zprávy s chybějícími segmenty (například proto, že byly v síti zpožděny a dosud nebyly přijaty), zadání parametru MQGMO_ALL_SEGMENTS_AVAILABLE zabrání načtení segmentů, které patří k neúplným logickým zprávám. Tyto segmenty však stále přispívají k hodnotě atributu fronty **CurrentQDepth** ; to znamená, že nemusí existovat žádné načítatelné logické zprávy, i když je hodnota CurrentQDepth větší než nula. Pokud nejsou k dispozici žádné další zprávy, které by bylo možné načíst, kód příčiny MQRC_NO_MSG_AVAILABLE se vrátí po uplynutí určeného intervalu čekání (pokud existuje).

Zpracování MQGMO_ALL_SEGMENTS_AVAILABLE závisí na tom, zda je zadána také hodnota MQGMO_LOGICAL_ORDER :

- Jsou-li zadány obě volby, má parametr MQGMO_ALL_SEGMENTS_AVAILABLE účinek pouze v případě, že neexistuje žádná aktuální logická zpráva. Pokud existuje aktuální logická zpráva, MQGMO_ALL_SEGMENTS_AVAILABLE se ignoruje. To znamená, že MQGMO_ALL_SEGMENTS_AVAILABLE může zůstat zapnutý při zpracování zpráv v logickém pořadí.
- Je-li parametr MQGMO_ALL_SEGMENTS_AVAILABLE zadán bez parametru MQGMO_LOGICAL_ORDER, má parametr MQGMO_ALL_SEGMENTS_AVAILABLE vždy nějaký účinek. To znamená, že volba musí být vypnuta po odebrání prvního segmentu v logické zprávě z fronty, aby bylo možné odebrat zbývající segmenty v logické zprávě.

Není-li tato volba uvedena, segmenty zpráv lze načíst i v případě, že je logická zpráva neúplná.

Zatímco MQGMO_COMPLETE_MSG i MQGMO_ALL_SEGMENTS_AVAILABLE vyžadují, aby byly všechny segmenty k dispozici dříve, než je lze načíst, první z nich vrátí úplnou zprávu, zatímco druhá umožňuje, aby byly segmenty načteny jeden po druhém.

Je-li pro zprávu sestavy zadána hodnota MQGMO_ALL_SEGMENTS_AVAILABLE , správce front zkontroluje frontu, aby zjistil, zda pro každý segment, který tvoří úplnou logickou zprávu, existuje alespoň jedna zpráva sestavy. Pokud existuje, podmínka MQGMO_ALL_SEGMENTS_AVAILABLE je splněna. Správce front však nekontroluje *typ* přítomných zpráv sestavy, a proto může ve zprávách sestavy existovat směs typů sestav souvisejících se segmenty logické zprávy. V důsledku toho úspěch produktu MQGMO_ALL_SEGMENTS_AVAILABLE neznamená, že MQGMO_COMPLETE_MSG bude úspěšný. Je-li pro segmenty konkrétní logické zprávy přítomna směs typů sestav, musí být tyto zprávy sestavy načteny jeden po druhém.

Můžete uvést MQGMO_ALL_SEGMENTS_AVAILABLE s jakoukoli jinou volbou MQGMO_* a s jakoukoli z voleb MQMO_* .

- V systému z/OSje tato volba podporována pro soukromé a sdílené fronty, ale fronta musí mít typ indexu MQIT_GROUP_ID. Pro sdílené fronty platí, že objekt CFSTRUCT, na který má být fronta mapována, musí být na úrovni CFLEVEL (3) nebo vyšší.
- Na následujících platformách:
  - AIX
  - IBM i
  - Linux
  - Windows

  a pro produkt IBM MQ MQI clients připojený k těmto systémům je tato volba podporována pro všechny lokální fronty.

Volby vlastností

Následující volby se týkají vlastností zprávy:

MQGMO_PROPERTIES_AS_Q_DEF

Vlastnosti zprávy, s výjimkou těch, které jsou obsaženy v deskriptoru zprávy (nebo rozšíření), by měly být reprezentovány tak, jak je definováno atributem fronty **PropertyControl** . Pokud je zadána volba MsgHandle , je tato volba ignorována a vlastnosti zprávy jsou k dispozici prostřednictvím volby MsgHandle, pokud hodnota atributu fronty **PropertyControl** není MQPROP_FORCE_MQRFH2.

Tato akce je výchozí, jestliže nejsou zadány žádné volby vlastností.

MQGMO_PROPERTIES_IN_HANDLE

Vlastnosti zprávy by měly být zpřístupněny prostřednictvím MsgHandle. Není-li k dispozici žádný manipulátor zprávy, volání se nezdaří s příčinou MQRC_HMSG_ERROR.

Poznámka: Pokud je zpráva později přečtena aplikací, která nevytváří manipulátor zprávy, správce front umístí všechny vlastnosti zprávy do struktury MQRFH2 . Můžete zjistit, že přítomnost neočekávaného záhlaví MQRFH2 narušuje chování existující aplikace.

MQGMO_NO_PROPERTIES

Nebudou načteny žádné vlastnosti zprávy kromě těch, které jsou obsaženy v deskriptoru zprávy (nebo rozšíření). Je-li zadán parametr MsgHandle , bude ignorován.

MQGMO_PROPERTIES_FORCE_MQRFH2

Vlastnosti zprávy, s výjimkou vlastností obsažených v deskriptoru zprávy (nebo rozšíření), by měly být reprezentovány pomocí záhlaví MQRFH2 . To poskytuje kompatibilitu s dřívější verzí pro aplikace, které očekávají načtení vlastností, ale nelze je změnit tak, aby používaly manipulátory zpráv. Je-li zadán parametr MsgHandle , bude ignorován.

MQGMO_PROPERTIES_COMPATIBILITY
Pokud zpráva obsahuje vlastnost s předponou **"mcd."**, **"jms."**, **"usr."**nebo **"mqext."**, jsou všechny vlastnosti zprávy doručeny do aplikace v záhlaví MQRFH2 . Jinak budou všechny vlastnosti zprávy, kromě vlastností obsažených v deskriptoru (či rozšíření) zprávy, zahozeny a nebudou nadále pro aplikaci přístupné.

Výchozí volba

Pokud není požadována žádná z popsaných voleb, lze použít následující volbu:

MQGMO_NONE
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty. Produkt MQGMO_NONE pomáhá s dokumentací k programům; není určeno, aby byla tato volba použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Počáteční hodnota pole Options je MQGMO_NO_WAIT plus MQGMO_PROPERTIES_AS_Q_DEF.

WaitInterval (MQLONG) pro MQGMO

Jedná se o přibližnou dobu vyjádřenou v milisekundách, po kterou volání MQGET čeká na doručení vhodné zprávy (tj. zprávy splňující kritéria výběru uvedená v parametru **MsgDesc** volání MQGET.

Důležité: Neexistuje žádné čekání nebo prodleva, pokud je okamžitě k dispozici vhodná zpráva.

Další podrobnosti viz pole *MsgId* popsané v části “MQMD-Deskriptor zpráv” na stránce 424). Pokud po uplynutí této doby nedorazila žádná vhodná zpráva, volání se dokončí s MQCC_FAILED a kódem příčiny MQRC_NO_MSG_AVAILABLE.

V systému z/OSje doba, po kterou volání MQGET skutečně čeká, ovlivněna načítáním systému a plánováním práce a může se lišit mezi hodnotou zadanou pro *WaitInterval* a přibližně o 100 milisekund větší než *WaitInterval*.

WaitInterval se používá ve spojení s volbou MQGMO_WAIT nebo MQGMO_SET_SIGNAL. Pokud není zadán ani jeden z nich, bude ignorován. Je-li zadána jedna z těchto hodnot, musí být hodnota *WaitInterval* větší nebo rovna nule nebo musí být zadána následující speciální hodnota:

MQWI_UNLIMITED
Neomezený interval čekání.

Počáteční hodnota tohoto pole je 0.

Signal1 (MQLONG) pro MQGMO

Toto je vstupní pole, které se používá pouze ve spojení s volbou MQGMO_SET_SIGNAL; identifikuje signál, který má být doručen, když je zpráva k dispozici.

Poznámka: Datový typ a využití tohoto pole jsou určeny prostředím; z tohoto důvodu nesmí aplikace, které chcete portovat mezi různými prostředími, používat signály.

- V systému z/OSmusí toto pole obsahovat adresu řídicího bloku událostí (ECB). Před vydáním volání MQGET musí být ECB aplikací vymazána. Úložiště obsahující ECB nesmí být uvolněno, dokud není fronta uzavřena. ECB je vyslána správcem front s jedním z popisovaných kódů dokončení signálu. Tyto kódy dokončení jsou stanoveny v bitech 2 až 31 ECB, přičemž oblast definovaná v mapovacím makru z/OS IHAECB je určena pro kód dokončení uživatele.
- Ve všech ostatních prostředích se jedná o vyhrazené pole; jeho hodnota není významná.

Kódy dokončení signálu jsou:

MQEC_MSG_ARRXX_ENCODE_CASE_ONE tabulka
Do fronty byla přijata vhodná zpráva. Tato zpráva nebyla vyhrazena pro volajícího; musí být zadán druhý požadavek MQGET, ale jiná aplikace může zprávu načíst před provedením druhého požadavku.

MQEC_WAIT_INTERVAL_VYPRŠELA
Uvedené *WaitInterval* vypršelo, aniž by byla doručena vhodná zpráva.

MQEC_WAIT_ZRUŠENO
Čekání bylo zrušeno z neurčitého důvodu (například ukončení správce front nebo zakázání fronty). Zadejte požadavek znovu, chcete-li provést další diagnostiku.

MQEC_Q_MGR_QUIESCING
Čekání bylo zrušeno, protože správce front přešel do klidového stavu (ve volání MQGET byla zadána volba MQGMO_FAIL_IF_QUIESCING).

MQEC_CONNECTION_QUIESCING
Čekání bylo zrušeno, protože připojení přešlo do klidového stavu (ve volání MQGET bylo zadáno MQGMO_FAIL_IF_QUIESCING).

Počáteční hodnota tohoto pole je určena prostředím:

- V systému z/OSje počáteční hodnota ukazatel Null.
- Ve všech ostatních prostředích je počáteční hodnota `0`.

Signal2 (MQLONG) pro MQGMO

Jedná se o vstupní pole, které se používá pouze ve spojení s volbou MQGMO_SET_SIGNAL. Jedná se o vyhrazené pole; jeho hodnota není významná.

Počáteční hodnota tohoto pole je `0`.

ResolvedQName (MQCHAR48) pro MQGMO

Jedná se o výstupní pole, které správce front nastaví na lokální název fronty, ze které byla zpráva načtena, jak je definováno pro lokálního správce front. Liší se od názvu použitého k otevření fronty, pokud:

- Byla otevřena alias fronta (v takovém případě je vrácen název lokální fronty, do které byl alias převeden), nebo
- Byla otevřena modelová fronta (v takovém případě je vrácen název dynamické lokální fronty).

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

MatchOptions (MQLONG) pro MQGMO

Tyto volby umožňují aplikaci zvolit, která pole v parametru **MsgDesc** se mají použít k výběru zprávy vrácené voláním MQGET. Aplikace nastaví požadované volby v tomto poli a poté nastaví odpovídající pole v parametru **MsgDesc** na hodnoty požadované pro tato pole. Pouze zprávy, které mají tyto hodnoty v deskriptoru MQMD pro zprávu, jsou kandidáty pro načtení pomocí tohoto parametru **MsgDesc** ve volání MQGET. Pole, pro která není zadána odpovídající volba shody, jsou při výběru zprávy, která má být vrácena, ignorována. Pokud ve volání MQGET nezadáte žádná kritéria výběru (tj. *jakákoli* zpráva je přijatelná), nastavte parametr *MatchOptions* na hodnotu MQMO_NONE.

- V systému z/OSmohou být kritéria výběru, která lze použít, omezena typem indexu použitého pro frontu. Další podrobnosti viz atribut fronty **IndexType** .

Zadáte-li MQGMO_LOGICAL_ORDER, jsou pro návrat do následujícího volání MQGET vhodné pouze určité zprávy:

- Pokud neexistuje žádná aktuální skupina nebo logická zpráva, jsou pro návrat vhodné pouze zprávy, které mají *MsgSeqNumber* rovno 1 a *Offset* rovno `0` . V této situaci můžete použít jednu nebo více následujících voleb shody, abyste vybrali, která z vhodných zpráv se vrátí:
  - MQMO_MATCH_MSG_ID
  - MQMO_MATCH_CORREL_ID
  - MQMO_MATCH_GROUP_ID
- Pokud existuje aktuální skupina nebo logická zpráva, pouze další zpráva ve skupině nebo dalším segmentu v logické zprávě je způsobilá k vrácení, a to nelze změnit zadáním voleb MQMO_ *.

V obou předchozích případech můžete uvést volby shody, které se nepoužijí, ale hodnota příslušného pole v parametru **MsgDesc** se musí shodovat s hodnotou odpovídajícího pole ve zprávě, která se má vrátit; volání selže s kódem příčiny MQRC_MATCH_OPTIONS_ERROR je, že tato podmínka není splněna.

Parametr *MatchOptions* je ignorován, pokud zadáte hodnotu MQGMO_MSG_UNDER_CURSOR nebo MQGMO_BROWSE_MSG_UNDER_CURSOR.

Získávání zpráv na základě vlastnosti zprávy se neprovádí pomocí voleb shody; další informace viz "SelectionString (MQCHARV) pro MQOD" na stránce 495.

Můžete určit jednu nebo více následujících voleb shody:

MQMO_MATCH_MSG_ID
Zpráva, která má být načtena, musí mít identifikátor zprávy, který odpovídá hodnotě pole *MsgId* v parametru **MsgDesc** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například identifikátor korelace).

Pokud tuto volbu vynecháte, pole *MsgId* v parametru **MsgDesc** se ignoruje a jakýkoli identifikátor zprávy se bude shodovat.

Poznámka: Identifikátor zprávy MQMI_NONE je speciální hodnota, která odpovídá libovolnému identifikátoru zprávy v deskriptoru MQMD pro zprávu. Proto je uvedení MQMO_MATCH_MSG_ID s MQMI_NONE stejné jako neuvedení MQMO_MATCH_MSG_ID.

MQMO_MATCH_CORREL_ID
Zpráva, která má být načtena, musí mít identifikátor korelace, který odpovídá hodnotě pole *CorrelId* v parametru **MsgDesc** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například identifikátor zprávy).

Pokud tuto volbu vynecháte, pole *CorrelId* v parametru **MsgDesc** se ignoruje a jakýkoli identifikátor korelace se bude shodovat.

Poznámka: Identifikátor korelace MQCI_NONE je speciální hodnota, která odpovídá *libovolnému* identifikátoru korelace v deskriptoru MQMD pro zprávu. Proto je uvedení MQMO_MATCH_CORREL_ID s MQCI_NONE stejné jako neuvedení MQMO_MATCH_CORREL_ID.

MQMO_MATCH_GROUP_ID
Zpráva, která má být načtena, musí mít identifikátor skupiny, který odpovídá hodnotě pole *GroupId* v parametru **MsgDesc** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například identifikátor korelace).

Pokud tuto volbu vynecháte, pole *GroupId* v parametru **MsgDesc** se ignoruje a jakýkoli identifikátor skupiny se bude shodovat.

Poznámka: Identifikátor skupiny MQGI_NONE je speciální hodnota, která odpovídá *libovolnému* identifikátoru skupiny v deskriptoru MQMD pro zprávu. Proto je uvedení MQMO_MATCH_GROUP_ID s MQGI_NONE stejné jako neuvedení MQMO_MATCH_GROUP_ID.

MQMO_MATCH_MSG_SEQ_NUMBER
Zpráva, která má být načtena, musí mít pořadové číslo zprávy, které odpovídá hodnotě pole *MsgSeqNumber* v parametru **MsgDesc** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například identifikátor skupiny).

Pokud tuto volbu vynecháte, pole *MsgSeqNumber* v parametru **MsgDesc** se ignoruje a libovolné pořadové číslo zprávy se bude shodovat.

MQMO_MATCH_OFFSET
Zpráva, která má být načtena, musí mít posunutí, které odpovídá hodnotě pole *Offset* v parametru **MsgDesc** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například pořadové číslo zprávy).

Pokud tuto volbu vynecháte, pole *Offset* v parametru **MsgDesc** se ignoruje a jakékoli posunutí se bude shodovat.

- Tato volba není v systému z/OSpodporována.

MQMO_MATCH_MSG_TOKEN
Zpráva, která má být načtena, musí mít token zprávy, který odpovídá hodnotě pole *MsgToken* ve struktuře MQGMO určené ve volání MQGET.

Tuto volbu můžete zadat pro všechny lokální fronty. Zadáte-li jej pro frontu, která má hodnotu *IndexType* MQIT_MSG_TOKEN (fronta spravovaná WLM), nelze pro volbu MQMO_MATCH_MSG_TOKEN určit žádné další volby shody.

Nemůžete uvést MQMO_MATCH_MSG_TOKEN s MQGMO_WAIT nebo MQGMO_SET_SIGNAL. Pokud aplikace chce čekat na doručení zprávy do fronty, která má hodnotu *IndexType* MQIT_MSG_TOKEN, zadejte hodnotu MQMO_NONE.

Pokud tuto volbu vynecháte, pole *MsgToken* v MQGMO se ignoruje a libovolný token zprávy se bude shodovat.

Pokud nezadáte žádnou z popsaných voleb, můžete použít následující volbu:

MQMO_NONE
Při výběru zprávy, která má být vrácena, nepoužívejte žádné shody. Všechny zprávy ve frontě jsou vhodné pro načtení (ale podléhají řízení volbami MQGMO_ALL_MSGS_AVAILABLE, MQGMO_ALL_SEGMENTS_AVAILABLE a MQGMO_COMPLETE_MSG).

MQMO_NONE pomáhá s dokumentací programu. Není zamýšleno, aby tato volba byla použita s jinou volbou MQMO_ *, ale protože její hodnota je nula, nelze takové použití zjistit.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQMO_MATCH_MSG_ID s MQMO_MATCH_CORREL_ID. Toto pole je ignorováno, pokud je *Version* menší než MQGMO_VERSION_2.

Poznámka: Počáteční hodnota pole *MatchOptions* je definována pro kompatibilitu s dřívějšími správci front produktu MQSeries . Při čtení řady zpráv z fronty bez použití kritérií výběru však tato počáteční hodnota vyžaduje, aby aplikace resetovala pole *MsgId* a *CorrelId* na MQMI_NONE a MQCI_NONE před každým voláním MQGET. Vyvarujte se nutnosti resetovat *MsgId* a *CorrelId* nastavením parametru *Version* na hodnotu MQGMO_VERSION_2a *MatchOptions* na hodnotu MQMO_NONE.

Související pojmy
Selektory zpráv v rozhraní JMS

GroupStatus (MQCHAR) pro MQGMO
Tento příznak označuje, zda je načtená zpráva ve skupině.

Má jednu z následujících hodnot:

MQGS_NOT_IN_GROUP
Zpráva není ve skupině.

MQGS_MSG_IN_GROUP
Zpráva je ve skupině, ale není poslední ve skupině.

MQGS_LAST_MSG_IN_GROUP
Zpráva je poslední ve skupině.

Toto je také hodnota vrácená, pokud se skupina skládá pouze z jedné zprávy.

Toto je výstupní pole. Počáteční hodnota tohoto pole je MQGS_NOT_IN_GROUP. Toto pole je ignorováno, pokud je *Version* menší než MQGMO_VERSION_2.

SegmentStatus (MQCHAR) pro MQGMO

Jedná se o příznak, který označuje, zda je načtená zpráva segmentem logické zprávy. Má jednu z následujících hodnot:

MQSS_NOT_A_SEGMENT
Zpráva není segmentem.

MQSS_SEGMENT
Zpráva je segmentem, ale není posledním segmentem logické zprávy.

MQSS_LAST_SEGMENT
Zpráva je posledním segmentem logické zprávy.

Toto je také vrácená hodnota, pokud se logická zpráva skládá pouze z jednoho segmentu.

V systému z/OSnastaví správce front toto pole vždy na hodnotu MQSS_NOT_A_SEGMENT.

Toto je výstupní pole. Počáteční hodnota tohoto pole je MQSS_NOT_A_SEGMENT. Toto pole je ignorováno, pokud je *Version* menší než MQGMO_VERSION_2.

Segmentace (MQCHAR) pro MQGMO

Jedná se o příznak, který označuje, zda je pro načtenou zprávu povolena další segmentace. Má jednu z následujících hodnot:

MQSEG_INHIBITED
Segmentace není povolena.

MQSEG_POVOLENO
Segmentace je povolena.

V systému z/OSnastaví správce front toto pole vždy na hodnotu MQSEG_INHIBITED.

Toto je výstupní pole. Počáteční hodnota tohoto pole je MQSEG_INHIBITED. Toto pole je ignorováno, pokud je *Version* menší než MQGMO_VERSION_2.

Reserved1 (MQCHAR) pro MQGMO

Toto je vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak. Toto pole je ignorováno, pokud je *Version* menší než MQGMO_VERSION_2.

MsgToken (MQBYTE16) pro MQGMO

Pole MsgToken -struktura MQGMO. Toto pole používá správce front k jedinečné identifikaci zprávy.

Jedná se o bajtový řetězec, který je generován správcem front za účelem jedinečné identifikace zprávy ve frontě. Token zprávy je generován při prvním umístění zprávy do správce front a zůstává se zprávou, dokud není zpráva trvale odebrána ze správce front, pokud není správce front restartován.

Když je zpráva odebrána z fronty, soubor *MsgToken* , který identifikoval tuto instanci zprávy, již není platný a není nikdy znovu použit. Pokud je správce front restartován, může být hodnota *MsgToken* , která identifikovala zprávu ve frontě před restartováním, neplatná po restartu. Produkt *MsgToken* se však nikdy znovu nepoužívá k identifikaci jiné instance zprávy. *MsgToken* je generován správcem front a není viditelný pro žádnou externí aplikaci.

Je-li zpráva vrácena voláním MQGET s dodáním MQGMO verze 3 nebo vyšší, správce front vrátí zprávu *MsgToken* identifikující zprávu ve frontě v MQGMO. Existuje jedna výjimka: Při odebírání zprávy z fronty mimo synchronizační bod nemusí správce front vrátit zprávu *MsgToken* , protože není užitečné identifikovat vrácenou zprávu v následném volání MQGET. Aplikace by měly používat produkt *MsgToken* pouze k odkazování na zprávu při následných voláních MQGET.

Pokud je zadána volba *MsgToken* a je zadána volba *MatchOption* MQMO_MATCH_MSG_TOKEN a není zadána volba MQGMO_MSG_UNDER_CURSOR ani volba MQGMO_BROWSE_MSG_UNDER_CURSOR, může být vrácena pouze zpráva určená hodnotou *MsgToken* . Volba je platná ve všech lokálních frontách bez ohledu na INDXTYPE a v systému z/OS musíte použít INDXTYPE (MSGTOKEN) pouze ve frontách správce pracovní zátěže (WLM).

Všechny ostatní uvedené *MatchOptions* jsou zkontrolovány, a pokud se neshodují, vrátí se MQRC_NO_MSG_AVAILABLE. Pokud je MQGMO_BROWSE_NEXT kódován pomocí MQMO_MATCH_MSG_TOKEN, zpráva identifikovaná pomocí *MsgToken* je vrácena pouze v případě, že je mimo kurzor procházení pro volající popisovač.

Je-li zadána volba MQGMO_MSG_UNDER_CURSOR nebo MQGMO_BROWSE_MSG_UNDER_CURSOR, bude volba MQMO_MATCH_MSG_TOKEN ignorována.

MQMO_MATCH_MSG_TOKEN není platný s následujícími volbami získání zprávy:

- MQGMO_WAIT
- MQGMO_SET_SIGNAL

Pro volání MQGET uvádějící MQMO_MATCH_MSG_TOKEN musí být do volání dodán MQGMO verze 3 nebo novější, jinak se vrátí MQRC_WRONG_GMO_VERSION.

Není-li parametr *MsgToken* v tomto okamžiku platný, vrátí se hodnota MQCC_FAILED s hodnotou MQRC_NO_MSG_AVAILABLE, pokud nedojde k další chybě.

ReturnedLength (MQLONG) pro MQGMO

Jedná se o výstupní pole, které správce front nastaví na délku dat zprávy vrácených voláním MQGET v parametru **Buffer** v bajtech. Pokud správce front tuto schopnost nepodporuje, je parametr *ReturnedLength* nastaven na hodnotu MQRL_UNDEFINED.

Když jsou zprávy převáděny mezi kódováním nebo znakovými sadami, data zprávy mohou někdy měnit velikost. Při návratu z volání MQGET:

- Pokud *ReturnedLength* není MQRL_UNDEFINED, počet bajtů vrácených dat zprávy je dán *ReturnedLength*.
- Pokud má *ReturnedLength* hodnotu MQRL_UNDEFINED, počet bajtů vrácených dat zprávy je obvykle dán menší hodnotou *BufferLength* a *DataLength*, ale může být *menší než* , pokud je volání MQGET dokončeno s kódem příčiny MQRC_TRUNCATED_MSG_ACCEPTED. Pokud k tomu dojde, nevýznamné bajty v parametru **Buffer** jsou nastaveny na hodnoty null.

Je definována následující speciální hodnota:

MQRL_UNDEFINED
Délka vrácených dat není definována.

V systému z/OSje hodnota vrácená pro pole *ReturnedLength* vždy MQRL_UNDEFINED.

Počáteční hodnota tohoto pole je MQRL_UNDEFINED. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQGMO_VERSION_3.

Reserved2 (MQLONG) pro MQGMO

Toto je vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak. Toto pole je ignorováno, pokud je hodnota *Version* menší než **MQGMO_VERSION_4**.

MsgHandle (MQHMSG) pro MQGMO

Pokud je zadána volba MQGMO_PROPERTIES_AS_Q_DEF a atribut fronty **PropertyControl** není nastaven na hodnotu MQPROP_FORCE_MQRFH2 , jedná se o popisovač zprávy, který bude naplněn vlastnostmi zprávy načítané z fronty. Manipulátor je vytvořen voláním MQCRTMH. Všechny vlastnosti, které jsou již přidruženy k popisovači, budou před načtením zprávy vymazány.

Lze také zadat následující hodnotu:
MQHM_NONE
Nebyl dodán žádný popisovač zprávy.

Pro volání MQGET není vyžadován žádný deskriptor zprávy, pokud je zadán platný manipulátor zprávy a používá se na výstupu, aby obsahoval vlastnosti zprávy, použije se deskriptor zprávy přidružený k manipulátoru zprávy pro vstupní pole.

Je-li ve volání MQGET určen deskriptor zprávy, má vždy přednost před deskriptorem zprávy přidruženým k manipulátoru zprávy.

Pokud je zadána volba MQGMO_PROPERTIES_FORCE_MQRFH2 nebo je zadána volba MQGMO_PROPERTIES_AS_AS_Q_DEF a atribut fronty **PropertyControl** je MQPROP_FORCE_MQRFH2 , volání selže s kódem příčiny MQRC_MD_ERROR, není-li zadán žádný parametr deskriptoru zprávy.

Při návratu z volání MQGET jsou vlastnosti a deskriptor zprávy přidružené k tomuto popisovači zprávy aktualizovány tak, aby odrážely stav načtené zprávy (stejně jako deskriptor zprávy, pokud byl zadán ve volání MQGET). Vlastnosti zprávy lze poté zjišťovat pomocí volání MQINQMP.

S výjimkou rozšíření deskriptoru zpráv není vlastnost, kterou lze požadovat pomocí volání MQINQMP, obsažena v datech zprávy; pokud zpráva ve frontě obsahovala vlastnosti v datech zprávy, jsou tyto vlastnosti odebrány z dat zprávy před vrácením dat do aplikace.

Není-li uveden žádný popisovač zprávy nebo je-li verze menší než MQGMO_VERSION_4 , musíte zadat platný deskriptor zprávy pro volání MQGET. Všechny vlastnosti zprávy (kromě těch, které jsou obsaženy v deskriptoru zprávy) jsou vráceny v předmětu dat zprávy s hodnotou voleb vlastností ve struktuře MQGMO a atributu fronty **PropertyControl** .

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQHM_NONE. Toto pole je ignorováno, pokud je hodnota **Version** menší než MQGMO_VERSION_4.

Záhlaví informací MQIIH- IMS

Struktura MQIIH popisuje informace záhlaví pro zprávu odeslanou do produktu IMS přes most IMS .Pro libovolnou podporovanou platformu IBM MQ můžete vytvořit a přenést zprávu, která obsahuje strukturu MQIIH, ale most IMS může používat pouze správce front IBM MQ for z/OS . Proto, aby se zpráva dostala do produktu IMS ze správce front jiného nežz/OS , musí síť správců front obsahovat alespoň jednoho správce front z/OS , jehož prostřednictvím může být zpráva směrována.

Dostupnost

Všechny systémy IBM MQ a klienty IBM MQ .

Název formátu

MQFMT_IMS

Znaková sada a kódování

Pro znakovou sadu a kódování používané pro strukturu MQIIH a data zpráv aplikace platí zvláštní podmínky:

- Aplikace, které se připojují ke správci front vlastnícímu frontu mostu IMS , musí poskytovat strukturu MQIIH, která je ve znakové sadě a kódování správce front. Důvodem je skutečnost, že v tomto případě není proveden převod dat struktury MQIIH.
- Aplikace, které se připojují k jiným správcům front, mohou poskytovat strukturu MQIIH, která je v libovolné z podporovaných znakových sad a kódování; přijímající agent kanálu zpráv připojený ke správci front, který vlastní frontu mostu IMS , převádí MQIIH.
- Data zprávy aplikace následující po struktuře MQIIH musí být ve stejné znakové sadě a kódování jako struktura MQIIH. Pole *CodedCharSetId* a *Encoding* ve struktuře MQIIH nepoužívejte k určení znakové sady a kódování dat zprávy aplikace.

  Chcete-li převést data zprávy aplikace, která nejsou jedním z vestavěných formátů podporovaných správcem front, musíte poskytnout uživatelskou proceduru pro převod dat.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 497. Pole v MQIIH pro MQIIH

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQIIH_STRUC_ID	'IIH¬'
Verze (číslo verze struktury)	MQIIH_VERSION_1	1
StrucLength (délka struktury MQIIH)	MQIIH_LENGTH_1	84
Kódování (vyhrazeno-viz “Znaková sada a kódování” na stránce 409)	Není	0
CodedCharSetId (vyhrazeno-viz “Znaková sada a kódování” na stránce 409)	Není	0
Formát (MQ název formátu dat, která následují za MQIIH)	MQFMT_NONE	Mezery
Příznaky (příznaky)	MQIIH_NONE	0
LTermOverride (potlačení logického terminálu)	Není	Mezery
MFSMapName (název mapy služeb formátu zpráv)	Není	Mezery
ReplyToFormát (název zprávy odpovědi ve formátuMQ)	MQFMT_NONE	Mezery
Authenticator (RACF heslo nebo přístupový prvek)	MQIAUT_NONE	Mezery
TranInstanceId (identifikátor instance transakce)	MQITII_NONE	Hodnoty null
TranState (stav transakce)	MQITS_NOT_IN_CONVERSATION	'¬'
CommitMode (režim vázaného zpracování)	MQICM_COMMIT_THEN_SEND	'0'
SecurityScope (rozsah zabezpečení)	MQISS_CHECK	'C'
Vyhrazeno (vyhrazeno)	Není	'¬'

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQIIH_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQIIH MyIIH = {MQIIH_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQIIH

```
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 */
};
```

Deklarace jazyka COBOL pro MQIIH

```
**   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.
```

Prohlášení PL/I pro MQIIH

```
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 */
```

Deklarace High Level Assembler pro MQIIH

```
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)
```

Prohlášení Visual Basic pro MQIIH

```
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
```

StrucId (MQCHAR4) pro MQIIH

Jedná se o identifikátor struktury struktury záhlaví informací IMS . Vždy se jedná o vstupní pole. Jeho hodnota je MQIIH_STRUC_ID.

Hodnota musí být:

MQIIH_STRUC_ID
Identifikátor struktury záhlaví informací IMS .

Pro programovací jazyk C je definována také konstanta MQIIH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQIIH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQIIH

Toto je číslo verze struktury. Hodnota musí být:

MQIIH_VERSION_1
Číslo verze pro strukturu záhlaví informací IMS .

Následující konstanta určuje číslo verze aktuální verze:

MQIIH_CURRENT_VERSION
Aktuální verze struktury záhlaví informací IMS .

Počáteční hodnota tohoto pole je MQIIH_VERSION_1.

StrucLength (MQLONG) pro MQIIH

Jedná se o délku struktury MQIIH. Hodnota musí být:

MQIIH_LENGTH_1
Délka struktury záhlaví informací IMS .

Počáteční hodnota tohoto pole je MQIIH_LENGTH_1.

Kódování (MQLONG) pro MQIIH

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

Kódování pro podporované struktury, které následují po struktuře MQIIH, je stejné jako kódování samotné struktury MQIIH a je převzato z předchozího záhlaví produktu MQ .

CodedCharSetId (MQLONG) pro MQIIH

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

ID znakové sady pro podporované struktury, které následují po struktuře MQIIH, je stejné jako ID samotné struktury MQIIH a je převzato z předchozího záhlaví produktu MQ .

Formát (MQCHAR8) pro MQIIH

Tato volba určuje název formátu produktu MQ pro data, která se řídí strukturou MQIIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQIIH

Hodnota příznaků musí být:

MQIIH_NONE
Žádné příznaky.

MQIIH_PASS_EXPIRATION
Zpráva odpovědi obsahuje:

- Stejné volby sestavy vypršení platnosti jako zpráva požadavku
- Zbývající čas vypršení platnosti ze zprávy požadavku bez úpravy pro dobu zpracování mostu

Není-li tato hodnota nastavena, je doba vypršení platnosti nastavena na *neomezeno*.

MQIIH_REPLY_FORMAT_NONE
Nastaví MQIIH.Format odpovědi na MQFMT_NONE.

MQIIH_IGNORE_PURG
Nastaví indikátor TMAMIPRG v předponě OTMA, který požaduje, aby OTMA ignorovala volání PURG na TP PCB pro transakce CM0 .

MQIIH_CM0_REQUEST_RESPONSE
Pro transakce režimu potvrzení 0 (CM0) tento příznak nastaví indikátor TMAMHRSP v předponě OTMA. Nastavení tohoto indikátoru vyžaduje, aby OTMA/IMS generoval zprávu DFS2082 RESPONSE MODE TRANSACTION TERMINATED WITHOUT REPLY, když původní aplikační program IMS neodpoví na IOPCB ani nepřepne zprávu na jinou transakci.

Počáteční hodnota tohoto pole je MQIIH_NONE.

LTermOverride (MQCHAR8) pro MQIIH

Přepis logického terminálu umístěný v poli IO PCB. Je volitelný; pokud není uveden, použije se název TPIPE. Ignoruje se, pokud je první bajt prázdný, nebo má hodnotu null.

Délka tohoto pole je dána hodnotou MQ_LTERM_OVERRIDE_LENGTH. Počáteční hodnota tohoto pole je 8 prázdných znaků.

MFSMapName (MQCHAR8) pro MQIIH

Název mapy služeb formátu zprávy, umístěné v poli IO PCB. Tato položka není povinná. Na vstupu představuje MID, na výstupu představuje MOD. Ignoruje se, pokud je první bajt prázdný nebo má hodnotu null.

Délka tohoto pole je dána hodnotou MQ_MFS_MAP_NAME_LENGTH. Počáteční hodnota tohoto pole je 8 prázdných znaků.

ReplyToFormát (MQCHAR8) pro MQIIH

Jedná se o název formátu MQ zprávy odpovědi, která je odeslána jako odpověď na aktuální zprávu. Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

Chcete-li převést data ve zprávě odpovědi pomocí příkazu MQGMO_CONVERT, zadejte hodnotu MQIIH.replyToFormat= MQFMT_STRING nebo MQIIH.replyToFormat= MQFMT_IMS_VAR_STRING. Vysvětlení použití těchto polí viz “Formát (MQCHAR8) pro MQMD” na stránce 449.

Pokud je pro zprávu požadavku použita výchozí hodnota (MQIIH.replyToFormat= MQFMT_NONE) a zpráva odpovědi je načtena pomocí příkazu MQGMO_CONVERT, nebude proveden žádný převod dat.

Ověřovatel (MQCHAR8) pro MQIIH

Toto je RACF heslo nebo PassTicket. Je volitelný; je-li zadán, používá se spolu s ID uživatele v kontextu zabezpečení MQMD k sestavení tokenu UTOKEN, který je odeslán do produktu IMS za účelem poskytnutí kontextu zabezpečení. Není-li uvedeno, použije se ID uživatele bez ověření. To závisí na nastavení přepínačů RACF , které mohou vyžadovat přítomnost ověřovatele.

Toto se ignoruje, pokud je první bajt prázdný nebo má hodnotu null. Lze použít následující speciální hodnotu:

MQIAUT_NONE
Bez ověření.

Pro programovací jazyk C je definována také konstanta MQIAUT_NONE_ARRAY; má stejnou hodnotu jako MQIAUT_NONE, ale je to pole znaků místo řetězce.

Délka tohoto pole je dána hodnotou MQ_AUTHENTICATOR_LENGTH. Počáteční hodnota tohoto pole je MQIAUT_NONE.

TranInstanceId (MQBYTE16) pro MQIIH

Jedná se o identifikátor instance transakce. Toto pole je používáno výstupními zprávami z IMS, takže je ignorováno na prvním vstupu. Nastavíte-li parametr *TranState* na hodnotu MQITS_IN_CONVERSATION, musí být tato hodnota uvedena v dalším vstupu a ve všech následných vstupech, aby produkt IMS mohl korelovat zprávy se správnou konverzací. Můžete použít následující speciální hodnotu:

MQITII_NONE
Žádný identifikátor instance transakce.

Pro programovací jazyk C je definována také konstanta MQITII_NONE_ARRAY, která má stejnou hodnotu jako MQITII_NONE, ale je to pole znaků místo řetězce.

Délka tohoto pole je dána hodnotou MQ_TRAN_INSTANCE_ID_LENGTH. Počáteční hodnota tohoto pole je MQITII_NONE.

TranState (MQCHAR) pro MQIIH

Označuje stav konverzace IMS . Tato volba je při prvním vstupu ignorována, protože neexistuje žádná konverzace. Na následných vstupech označuje, zda je konverzace aktivní nebo ne. Na výstupu je nastavena pomocí IMS. Hodnota musí být jedna z následujících:

MQITS_IN_CONVERSATION
V rozhovoru.

MQITS_NOT_IN_CONVERSATION
Ne v konverzaci.

MQITS_ARCHITEKTED
Vrátit stavová data transakce v architektonické podobě.

Tato hodnota se používá pouze s příkazem IMS /DISPLAY TRAN . Vrací stavová data transakce v architektuře IMS namísto znakového tvaru. z/OS Další informace viz Psaní IMS transakčních programů prostřednictvím IBM MQ.

Počáteční hodnota tohoto pole je MQITS_NOT_IN_CONVERSATION.

CommitMode (MQCHAR) pro MQIIH

Jedná se o režim potvrzení IMS . Další informace o režimech potvrzení IMS viz *OTMA Reference* . Hodnota musí být jedna z následujících:

MQICM_COMMIT_THEN_SEND
Potvrdit a odeslat.

Tento režim znamená dvojité řazení výstupu do fronty, ale kratší dobu obsazenosti oblasti. Rychlé cesty a konverzační transakce nemohou být spuštěny s tímto režimem.

MQICM_SEND_THEN_COMMIT
Odeslat a poté potvrdit.

Jakákoli transakce IMS zahájená jako výsledek režimu potvrzení MQICM_SEND_THEN_COMMIT se spustí v režimu RESPONSE bez ohledu na to, jak je transakce definována v definici systému IMS (parametr MSGTYPE v makru TRANSACT). To platí i pro transakce zahájené prostřednictvím přepínače transakcí.

Počáteční hodnota tohoto pole je MQICM_COMMIT_THEN_SEND.

SecurityScope (MQCHAR) pro MQIIH

Označuje, že je vyžadováno zpracování zabezpečení IMS . Jsou definovány tyto hodnoty:

MQISS_CHECK
Zkontrolujte rozsah zabezpečení: prostředí ACEE je sestaveno v řídicí oblasti, nikoli však v závislé oblasti.

MQISS_FULL
Úplný rozsah zabezpečení: modul ACEE uložený v mezipaměti je sestaven v řídicí oblasti a v závislé oblasti je sestaven modul ACEE, který není uložen v mezipaměti. Používáte-li MQISS_FULL, ujistěte se, že ID uživatele, pro které je sestaveno ACEE, má přístup k prostředkům používaným v závislé oblasti.

Není-li pro toto pole uvedeno MQISS_CHECK ani MQISS_FULL, předpokládá se MQISS_CHECK.

Počáteční hodnota tohoto pole je MQISS_CHECK.

Vyhrazeno (MQCHAR) pro MQIIH

Toto je vyhrazené pole; musí být prázdné.

MQIMPO-Volby vlastnosti dotazové zprávy

Struktura MQIMPO umožňuje aplikacím určit volby, které řídí, jak jsou zjišťovány vlastnosti zpráv. Struktura je vstupní parametr volání MQINQMP.

Dostupnost

Všechny systémy IBM MQ a klienty IBM MQ .

Znaková sada a kódování

Data v objektu MQIMPO musí být ve znakové sadě aplikace a kódování aplikace (MQENC_NATIVE).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 498. Pole v MQIPMO

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQIMPO_STRUC_ID	'IMPO'
Verze (číslo verze struktury)	MQIMPO_VERSION_1	1
Volby (volby řídící akci MQINQMP)	MQIMPO_INQ_FIRST	
RequestedEncoding (kódování, do kterého má být požadovaná vlastnost převedena)	MQENC_NATIVE	
RequestedCCSID (znaková sada dotazované vlastnosti)	MQCCSI_APPL	
ReturnedEncoding (kódování vrácené hodnoty)	MQENC_NATIVE	
ReturnedCCSID	0	
Reserved1 (vyhrazené pole)	prázdný znak (4bajtové pole)	
ReturnedName (název dotazovaných vlastností)	VÝCHOZÍ	
TypeString (řetězcová reprezentace datového typu vlastnosti)	Prázdný řetězec nebo mezery	

Notes:

1. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyku C se jedná o proměnnou makra.MQIMPO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQIMPO MyIMPO = {MQIMPO_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQIMPO

```
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 */
};
```

Deklarace jazyka COBOL pro objekt MQIMPO

```
**   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.
```

Deklarace PL/I pro objekt MQIMPO

```
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 */
```

Deklarace High Level Assembler pro objekt MQIMPO

```
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)
```

StrucId (MQCHAR4) pro MQIMPO

Jedná se o identifikátor struktury struktury voleb vlastností dotazové zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQIMPO_STRUC_ID.

Hodnota musí být:

MQIMPO_STRUC_ID
Identifikátor pro strukturu voleb vlastností dotazové zprávy.

Pro programovací jazyk C je definována také konstanta MQIMPO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQIMPO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQIMPO

Struktura voleb vlastností dotazové zprávy-pole Verze

Toto je číslo verze struktury. Hodnota musí být:

MQIMPO_VERSION_1
Číslo verze pro strukturu voleb vlastností dotazové zprávy.

Následující konstanta určuje číslo verze aktuální verze:

MQIMPO_CURRENT_VERSION
Aktuální verze struktury voleb vlastností dotazové zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQIMPO_VERSION_1.

Volby (MQLONG) pro MQIMPO

Struktura voleb vlastností dotazové zprávy-pole Volby

Následující volby řídí akci MQINQMP. Můžete uvést jednu nebo více těchto voleb. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Kombinace voleb, které nejsou platné, jsou uvedeny; všechny ostatní kombinace jsou platné.

Volby dat hodnoty: Následující volby se vztahují ke zpracování dat hodnoty, když je vlastnost načtena ze zprávy.

MQIMPO_CONVERT_VALUE
Tato volba vyžaduje, aby byla hodnota vlastnosti převedena tak, aby byla v souladu s hodnotami *RequestedCCSID* a *RequestedEncoding* určenými před voláním MQINQMP vrátí hodnotu vlastnosti v oblasti *Value* .

- Pokud je převod úspěšný, pole *ReturnedCCSID* a *ReturnedEncoding* jsou nastavena na stejnou hodnotu jako pole *RequestedCCSID* a *RequestedEncoding* při návratu z volání MQINQMP.
- Pokud se převod nezdaří, ale volání MQINQMP se jinak dokončí bez chyby, hodnota vlastnosti se vrátí nepřevedená.

  Pokud je vlastnost řetězec, pole *ReturnedCCSID* a *ReturnedEncoding* jsou nastavena na znakovou sadu a kódování nepřevedeného řetězce.

  Kód dokončení je v tomto případě MQCC_WARNING s kódem příčiny MQRC_PROP_VALUE_NOT_CONVERTED. Kurzor vlastnosti je rozšířen na vrácenou vlastnost.

Pokud se hodnota vlastnosti během převodu rozšíří a překročí velikost parametru **Value** , hodnota se vrátí nepřevedená s kódem dokončení MQCC_FAILED; kód příčiny je nastaven na MQRC_PROPERTY_VALUE_TOO_BIG.

Parametr **DataLength** volání MQINQMP vrací délku, na kterou by byla hodnota vlastnosti převedena, aby mohla aplikace určit velikost vyrovnávací paměti potřebnou pro přizpůsobení převedené hodnoty vlastnosti. Kurzor vlastnosti je nezměněn.

Tato volba také vyžaduje, aby:

- Pokud název vlastnosti obsahuje zástupný znak a
- Pole *ReturnedName* je inicializováno s adresou nebo offsetem pro vrácené jméno,

pak je vrácený název převeden tak, aby odpovídal hodnotám *RequestedCCSID* a *RequestedEncoding* .

- Pokud je převod úspěšný, pole *VSCCSID* pro *ReturnedName* a kódování vráceného názvu se nastaví na vstupní hodnotu *RequestedCCSID* a *RequestedEncoding*.
- Pokud se převod nezdaří, ale volání MQINQMP se jinak dokončí bez chyby nebo varování, vrácený název se nepřevede. Kód dokončení je v tomto případě MQCC_WARNING s kódem příčiny MQRC_PROP_NAME_NOT_CONVERTED.

  Kurzor vlastnosti je rozšířen na vrácenou vlastnost. MQRC_PROP_VALUE_NOT_CONVERTED je vrácen, pokud hodnota i název nejsou převedeny.

Pokud se vrácený název během převodu rozšíří a překročí velikost pole *VSBufsize* v souboru *RequestedName*, vrácený řetězec zůstane nepřevedený s kódem dokončení MQCC_FAILED a kód příčiny je nastaven na MQRC_PROPERTY_NAME_TOO_BIG.

Pole *VSLength* struktury MQCHARV vrací délku, na kterou by byla hodnota vlastnosti převedena, aby mohla aplikace určit velikost vyrovnávací paměti potřebnou pro uložení převedené hodnoty vlastnosti. Kurzor vlastnosti je nezměněn.

MQIMPO_CONVERT_TYPE

Tato volba vyžaduje, aby byla hodnota vlastnosti převedena z aktuálního datového typu na datový typ určený parametrem **Type** volání MQINQMP.

- Je-li převod úspěšný, parametr **Type** se při návratu volání MQINQMP nezmění.
- Pokud se převod nezdaří, ale volání MQINQMP se jinak dokončí bez chyby, volání se nezdaří s příčinou MQRC_PROP_CONV_NOT_SUPPORTED. Kurzor vlastnosti je nezměněn.

Pokud převod datového typu způsobí, že se hodnota během převodu rozbalí a převedená hodnota překročí velikost parametru **Value** , hodnota se vrátí nepřevedená s kódem dokončení MQCC_FAILED a kód příčiny je nastaven na MQRC_PROPERTY_VALUE_TOO_BIG.

Parametr **DataLength** volání MQINQMP vrací délku, na kterou by byla hodnota vlastnosti převedena, aby mohla aplikace určit velikost vyrovnávací paměti potřebnou pro přizpůsobení převedené hodnoty vlastnosti. Kurzor vlastnosti je nezměněn.

Není-li hodnota parametru **Type** volání MQINQMP platná, volání selže s příčinou MQRC_PROPERTY_TYPE_ERROR.

Není-li požadovaný převod datového typu podporován, volání se nezdaří s příčinou MQRC_PROP_CONV_NOT_SUPPORTED. Podporovány jsou následující převody datových typů:

Tabulka 499. Podporované převody datových typů

Datový typ vlastnosti	Podporované cílové datové typy
MQTYPE_BOOLEAN	MQTYPE_STRING, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_BYTE_STRING	MQTYPE_STRING

Tabulka 499. Podporované převody datových typů (pokračování)

Datový typ vlastnosti	Podporované cílové datové typy
MQTYPE_INT8	MQTYPE_STRING, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_INT16	MQTYPE_STRING, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_INT32	MQTYPE_STRING, MQTYPE_INT64
MQTYPE_INT64	MQTYPE_STRING
MQTYPE_FLOAT32	MQTYPE_STRING, MQTYPE_FLOAT64
MQTYPE_FLOAT64	MQTYPE_STRING
MQTYPE_STRING	MQTYPE_BOOLEAN, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64, MQTYPE_FLOAT32, MQTYPE_FLOAT64
MQTYPE_NULL	Není

Obecná pravidla pro podporované převody jsou následující:

- Číselné hodnoty vlastností lze převést z jednoho datového typu na jiný za předpokladu, že během převodu nebudou ztracena žádná data.

  Například hodnotu vlastnosti s datovým typem MQTYPE_INT32 lze převést na hodnotu s datovým typem MQTYPE_INT64, ale nelze ji převést na hodnotu s datovým typem MQTYPE_INT16.
- Hodnotu vlastnosti libovolného datového typu lze převést na řetězec.
- Hodnotu vlastnosti řetězce lze převést na jakýkoli jiný datový typ za předpokladu, že je řetězec pro převod správně naformátován. Pokud se aplikace pokusí převést hodnotu vlastnosti řetězce, která není správně formátována, produkt IBM MQ vrátí kód příčiny MQRC_PROP_NUMBER_FORMAT_ERROR.
- Pokud se aplikace pokusí o převod, který není podporován, produkt IBM MQ vrátí kód příčiny MQRC_PROP_CONV_NOT_SUPPORTED.

Specifická pravidla pro převod hodnoty vlastnosti z jednoho datového typu na jiný jsou následující:

- Při převodu hodnoty vlastnosti MQTYPE_BOOLEAN na řetězec je hodnota TRUE převedena na řetězec "TRUE" a hodnota false je převedena na řetězec "FALSE".
- Při převodu hodnoty vlastnosti MQTYPE_BOOLEAN na číselný datový typ je hodnota TRUE převedena na hodnotu jedna a hodnota FALSE je převedena na nulu.
- Při převodu hodnoty vlastnosti řetězce na hodnotu MQTYPE_BOOLEAN je řetězec "TRUE" nebo "1" převeden na hodnotu TRUE a řetězec "FALSE" nebo "0" je převeden na hodnotu FALSE.

  Všimněte si, že výrazy "TRUE" a "FALSE" nerozlišují velká a malá písmena.

  Žádný jiný řetězec nelze převést; funkce IBM MQ vrátí kód příčiny MQRC_PROP_NUMBER_FORMAT_ERROR.
- Při převodu hodnoty vlastnosti řetězce na hodnotu s datovým typem MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32 nebo MQTYPE_INT64musí mít řetězec následující formát:

```
[blanks][sign]digits
```

  Význam komponent řetězce je následující:

  blanks
  Volitelné úvodní prázdné znaky

  sign
  Volitelný znak plus (+) nebo znak minus (-).

digits
Souvislá posloupnost číselných znaků (0-9). Musí být uveden alespoň jeden znak číslice.

Po posloupnosti číselných znaků může řetězec obsahovat další znaky, které nejsou číselnými znaky, ale převod se zastaví, jakmile je dosaženo prvního z těchto znaků. Předpokládá se, že řetězec představuje desetinné celé číslo.

IBM MQ vrací kód příčiny MQRC_PROP_NUMBER_FORMAT_ERROR, pokud není řetězec správně naformátován.

- Při převodu hodnoty vlastnosti řetězce na hodnotu s datovým typem MQTYPE_FLOAT32 nebo MQTYPE_FLOAT64musí mít řetězec následující formát:

```
[blanks][sign]digits[.digits][e_char[e_sign]e_digits]
```

Význam komponent řetězce je následující:

blanks
Volitelné úvodní prázdné znaky

sign
Volitelný znak plus (+) nebo znak minus (-).

digits
Souvislá posloupnost číselných znaků (0-9). Musí být uveden alespoň jeden znak číslice.

e_char
Exponent, který je buď "E", nebo "e".

e_sign
Volitelný znak plus (+) nebo minus (-) pro exponent.

e_digits
Souvislá posloupnost číselných znaků (0-9) pro exponent. Pokud řetězec obsahuje znak exponentu, musí být uveden alespoň jeden znak číslice.

Po posloupnosti číselných znaků nebo volitelných znaků reprezentujících exponent může řetězec obsahovat jiné znaky, které nejsou číselnými znaky, ale převod se zastaví, jakmile je dosaženo prvního z těchto znaků. Předpokládá se, že řetězec představuje desetinné číslo s pohyblivou řádovou čárkou s exponentem, který je mocninou 10.

IBM MQ vrací kód příčiny MQRC_PROP_NUMBER_FORMAT_ERROR, pokud není řetězec správně naformátován.

- Při převodu hodnoty číselné vlastnosti na řetězec je hodnota převedena na řetězcovou reprezentaci hodnoty jako desetinné číslo, nikoli řetězec obsahující pro tuto hodnotu znak ASCII. Například celé číslo 65 se převede na řetězec "65", nikoli na řetězec "A".
- Při převodu hodnoty vlastnosti bajtového řetězce na řetězec je každý bajt převeden na dva hexadecimální znaky, které představují bajt. Například bajtové pole {0xF1, 0x12, 0x00, 0xFF} je převedeno na řetězec "F11200FF".

MQIMPO_QUERY_LENGTH
Zadejte dotaz na typ a délku hodnoty vlastnosti. Délka je vrácena v parametru **DataLength** volání MQINQMP. Hodnota vlastnosti není vrácena.

Je-li zadána vyrovnávací paměť **ReturnedName** , je pole *VSLength* struktury MQCHARV vyplněno délkou názvu vlastnosti. Název vlastnosti není vrácen.

Volby iterace: Následující volby se týkají iterace nad vlastnostmi s použitím názvu se zástupným znakem.

MQIMPO_INQ_FIRST
Dotázat se na první vlastnost, která odpovídá zadanému názvu. Po tomto volání se na vrácené vlastnosti ustanoví kurzor.

Toto je výchozí hodnota.

Volbu MQIMPO_INQ_PROP_UNDER_CURSOR lze následně použít s voláním MQINQMP, je-li vyžadováno, k opětovnému dotazu na stejnou vlastnost.

Všimněte si, že existuje pouze jeden kurzor vlastnosti; proto, pokud název vlastnosti zadaný ve volání MQINQMP změní kurzor na reset.

Tato volba není platná s žádnou z následujících voleb:

MQIMPO_INQ_NEXT
MQIMPO_INQ_PROP_UNDER_CURSOR

MQIMPO_INQ_NEXT
Dotazuje se na další vlastnost, která odpovídá zadanému názvu, a pokračuje v hledání z kurzoru vlastnosti. Kurzor je rozšířen na vrácenou vlastnost.

Pokud se jedná o první volání MQINQMP pro uvedený název, vrátí se první vlastnost, která odpovídá zadanému názvu.

Volbu MQIMPO_INQ_PROP_UNDER_CURSOR lze v případě potřeby následně použít s voláním MQINQMP, aby se znovu dotázali na stejnou vlastnost.

Pokud byla vlastnost pod kurzorem odstraněna, vrátí MQINQMP další odpovídající vlastnost následující po té, která byla odstraněna.

Je-li přidána vlastnost, která odpovídá zástupnému znaku, zatímco probíhá iterace, tato vlastnost může nebo nemusí být vrácena během dokončení iterace. Vlastnost je vrácena po restartování iterace pomocí MQIMPO_INQ_FIRST.

Vlastnost odpovídající zástupnému znaku, který byl odstraněn během probíhající iterace, není vrácena po jejím odstranění.

Tato volba není platná s žádnou z následujících voleb:

MQIMPO_INQ_FIRST
MQIMPO_INQ_PROP_UNDER_CURSOR

MQIMPO_INQ_PROP_UNDER_CURSOR
Načtěte hodnotu vlastnosti, na kterou ukazuje kurzor vlastnosti. Vlastnost, na kterou ukazuje kurzor vlastnosti, je ta, která byla naposledy dotazována pomocí volby MQIMPO_INQ_FIRST nebo MQIMPO_INQ_NEXT.

Kurzor vlastnosti je resetován při opětném použití manipulátoru zprávy, při zadání manipulátoru zprávy v poli *MsgHandle* MQGMO ve volání MQGET nebo při zadání manipulátoru zprávy v polích *OriginalMsgHandle* nebo *NewMsgHandle* struktury MQPMO ve volání MQPUT.

Je-li tato volba použita v případě, že kurzor vlastnosti dosud nebyl vytvořen, nebo byla-li vlastnost, na kterou odkazuje kurzor vlastnosti, odstraněna, volání selže s kódem dokončení MQCC_FAILED a důvodem MQRC_PROPERTY_NOT_AVAILABLE.

Tato volba není platná s žádnou z následujících voleb:

MQIMPO_INQ_FIRST
MQIMPO_INQ_NEXT

Pokud není požadována žádná z dříve popsaných voleb, lze použít následující volbu:

MQIMPO_NONE
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

MQIMPO_NONE pomáhá s dokumentací k programu. Není zamýšleno, aby tato volba byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQIMPO_INQ_FIRST.

RequestedEncoding (MQLONG) pro objekt MQIMPO
Struktura voleb vlastností dotazovací zprávy-pole RequestedEncoding

Jedná se o kódování, do kterého má být hodnota dotazované vlastnosti převedena při zadání parametru MQIMPO_CONVERT_VALUE nebo MQIMPO_CONVERT_TYPE.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

RequestedCCSID (MQLONG) pro MQIMPO
Struktura voleb vlastností dotazové zprávy-pole RequestedCCSID

Znaková sada, na kterou má být hodnota dotazované vlastnosti převedena, pokud je hodnota znakový řetězec. Jedná se také o znakovou sadu, do které má být při zadání parametru MQIMPO_CONVERT_VALUE nebo MQIMPO_CONVERT_TYPE převedena hodnota *ReturnedName* .

Počáteční hodnota tohoto pole je MQCCSI_APPL.

ReturnedEncoding (MQLONG) pro MQIMPO
Struktura voleb vlastností dotazovací zprávy-pole ReturnedEncoding

Na výstupu se jedná o kódování vrácené hodnoty.

Je-li zadána volba MQIMPO_CONVERT_VALUE a převod proběhl úspěšně, bude pole *ReturnedEncoding* při návratu stejnou hodnotou jako předaná hodnota.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

ReturnedCCSID (MQLONG) pro MQIMPO
Struktura voleb vlastností dotazové zprávy-pole ReturnedCCSID

Na výstupu se jedná o znakovou sadu vrácené hodnoty, pokud parametr **Type** volání MQINQMP je MQTYPE_STRING.

Je-li zadána volba MQIMPO_CONVERT_VALUE a převod proběhl úspěšně, bude pole *ReturnedCCSID* při návratu stejnou hodnotou jako předaná hodnota.

Počáteční hodnota tohoto pole je nula.

Reserved1 (MQCHAR) pro MQIMPO

Toto je vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak (4bajtové pole).

ReturnedName (MQCHARV) pro MQIMPO
Struktura voleb vlastností dotazové zprávy-pole ReturnedName

Skutečný název dotazovaného objektu.

Při vstupu lze vyrovnávací paměť řetězců předat pomocí pole *VSPtr* nebo *VSOffset* struktury MQCHARV . Délka vyrovnávací paměti řetězců je určena pomocí pole *VSBufsize* struktury MQCHARV.

Při návratu z volání MQINQMP je vyrovnávací paměť řetězců dokončena s názvem vlastnosti, která byla zjišťována, za předpokladu, že vyrovnávací paměť řetězců byla dostatečně dlouhá, aby plně obsahovala název. Pole *VSLength* struktury MQCHARV je vyplněno délkou názvu vlastnosti. Pole *VSCCSID* struktury MQCHARV je vyplněno, aby označilo znakovou sadu vráceného názvu, bez ohledu na to, zda se nezdařil převod názvu.

Toto je vstupní/výstupní pole. Počáteční hodnota tohoto pole je MQCHARV_DEFAULT.

TypeString (MQCHAR8) pro objekt MQIMPO
Struktura voleb vlastností dotazové zprávy-pole TypeString

Řetězcová reprezentace datového typu vlastnosti.

Pokud byla vlastnost určena v záhlaví MQRFH2 a atribut MQRFH2 `dt` nebyl rozpoznán, lze toto pole použít k určení datového typu vlastnosti. *TypeString* je vráceno v kódované znakové sadě 1208 (UTF-8) a je to prvních osm bajtů hodnoty atributu `dt` vlastnosti, kterou se nepodařilo rozpoznat.

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je nulový řetězec v programovacím jazyku C a 8 prázdných znaků v jiných programovacích jazycích.

MQMD-Deskriptor zpráv

Struktura MQMD obsahuje řídicí informace, které doprovázejí data aplikace, když zpráva prochází mezi odesílající a přijímající aplikací. Struktura je vstupní/výstupní parametr pro volání MQGET, MQPUT a MQPUT1 .

Dostupnost

Všechny systémy IBM MQ a IBM MQ MQI clients připojené k těmto systémům.

Verze

Aktuální verze MQMD je MQMD_VERSION_2. Aplikace, které mají být přenosné mezi několika prostředími, musí zajistit, aby požadovaná verze MQMD byla podporována ve všech příslušných prostředích. Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech, které následují.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi MQMD podporovanou prostředím, ale s počáteční hodnotou pole *Version* nastavenou na MQMD_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole *Version* na číslo verze požadované verze.

Deklarace pro strukturu version-1 je k dispozici s názvem MQMD1.

Znaková sada a kódování

Data v deskriptoru MQMD musí být ve znakové sadě a kódování lokálního správce front; tato data jsou dána atributem správce front **CodedCharSetId** a MQENC_NATIVE. Pokud je však aplikace spuštěna jako IBM MQ MQI client, musí být struktura ve znakové sadě a kódování klienta.

Pokud odesílající a přijímající správci front používají různé znakové sady nebo kódování, data v MQMD se převedou automaticky. Převod deskriptoru MQMD aplikací není nutný.

Použití různých verzí MQMD

version-2 MQMD je ekvivalentem použití version-1 MQMD a předpony dat zprávy se strukturou MQMDE. Pokud však všechna pole ve struktuře MQMDE mají své výchozí hodnoty, lze MQMDE vynechat. version-1 MQMD plus MQMDE se používají podle popisu:

- Pokud aplikace ve voláních MQPUT a MQPUT1 poskytuje MQMD version-1 , může volitelně přidat k datům zprávy předponu MQMDE a nastavit pole *Format* v MQMD na MQFMT_MD_EXTENSION, aby označila přítomnost MQMDE. Pokud aplikace neposkytne prostředí MQMDE, správce front předpokládá výchozí hodnoty pro pole v prostředí MQMDE.

  Poznámka: Několik polí, která existují ve volání MQMD version-2 , ale nikoli version-1 MQMD, jsou vstupní/výstupní pole ve voláních MQPUT a MQPUT1 . Správce front však nevrací žádné hodnoty v ekvivalentních polích ve výstupu MQMDE z volání MQPUT a MQPUT1 . Pokud aplikace tyto výstupní hodnoty vyžaduje, musí použít MQMD version-2 .
- Pokud aplikace ve volání MQGET poskytuje MQMD version-1 , správce front před zprávu vrácenou s MQMDE opatřuje předponou, ale pouze v případě, že jedno nebo více polí v MQMDE má jinou než výchozí hodnotu. Pole *Format* v MQMD bude mít hodnotu MQFMT_MD_EXTENSION, která označuje přítomnost MQMDE.

Výchozí hodnoty, které správce front používá pro pole v prostředí MQMDE, jsou stejné jako počáteční hodnoty těchto polí uvedené v části Tabulka 503 na stránce 475.

Nachází-li se zpráva v přenosové frontě, jsou některá pole v deskriptoru MQMD nastavena na konkrétní hodnoty. Podrobnosti naleznete v části “MQXQH-záhlaví přenosové fronty” na stránce 621 .

kontext zprávy

Určitá pole v deskriptoru MQMD obsahují kontext zprávy. Existují dva typy kontextu zprávy: *kontext identity* a *původní kontext*. Obvykle:

- Kontext identity se vztahuje k aplikaci, která *původně* vložila zprávu.
- Původní kontext souvisí s aplikací, která *naposledy* vložila zprávu.

Tyto dvě aplikace mohou být stejné, ale mohou být také různé aplikace (například když je zpráva předána z jedné aplikace do druhé).

Ačkoli identita a původní kontext mají obvykle popsaný význam, obsah obou typů polí kontextu v deskriptoru MQMD závisí na volbách MQPMO_ * _CONTEXT, které jsou určeny při vložení zprávy. V důsledku toho nemusí kontext identity nutně souviset s aplikací, která zprávu původně vložila, a kontext původu nemusí nutně souviset s aplikací, která zprávu vložila naposledy. Záleží na návrhu sady aplikací.

Agent kanálu zpráv (MCA) nikdy nemění kontext zprávy. Adaptéry MCA, které přijímají zprávy od vzdálených správců front, používají volbu kontextu MQPMO_SET_ALL_CONTEXT ve volání MQPUT nebo MQPUT1 . To umožňuje přijímajícímu adaptéru MCA přesně zachovat kontext zprávy, který procházeli se zprávou z odesílajícího agenta MCA. Výsledkem však je, že původní kontext nesouvisí s žádnou z MCA, která zprávu odeslala a přijala. Původní kontext odkazuje na dřívější aplikaci, která vložila zprávu. Pokud všechny zprostředkující aplikace předaly kontext zprávy, původní kontext odkazuje na samotnou původní aplikaci.

V popisech jsou kontextová pole popsána tak, jako by byla použita tak, jak bylo popsáno dříve. Další informace o kontextu zprávy viz Kontext zprávy.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 500. Pole v MQMD pro MQMD

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	ID_STRUC_MQMD_STRUC_ID	'MD'
Verze (číslo verze struktury)	MQMD_VERSION_1	1
Sestava (volby pro zprávy sestavy)	MQRO_NONE	0
MsgType (typ zprávy)	MQMT_DATAGRAM	8
MQMD-Pole vypršení platnosti (životnost zprávy)	MQEI_UNLIMITED	-1
Pole MQMD-Feedback (zpětná vazba nebo kód příčiny)	MQFB_NONE	0
Kódování (číselné kódování dat zprávy)	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId (identifikátor znakové sady dat zprávy)	MQCCSI_Q_MGR	0
Formát (název formátu dat zprávy)	MQFMT_NONE	Mezery

Tabulka 500. Pole v MQMD pro MQMD (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
Priorita (priorita zprávy)	MQPRI_PRIORITY_AS_Q_DEF	-1
Perzistence (perzistence zpráv)	MQPER_PERSISTENCE_AS_Q_DEF	2
MQMD-pole MsgId (identifikátor zprávy)	MQMI_NONE	Hodnoty null
CorrelId (identifikátor korelace)	MQCI_NONE	Hodnoty null
BackoutCount (čítač vrácení)	Není	0
ReplyToQ (název fronty odpovědí)	Není	Prázdný řetězec nebo mezery
ReplyToQMgr (název správce front odpovědí)	Není	Prázdný řetězec nebo mezery
UserIdentifier (identifikátor uživatele)	Není	Prázdný řetězec nebo mezery
AccountingToken (token evidence)	MQACT_NONE	Hodnoty null
ApplIdentityData (data aplikace související s identitou)	Není	Prázdný řetězec nebo mezery
PutApplTyp (typ aplikace, která vložila zprávu)	MQAT_NO_CONTEXT	0
PutApplNázev (název aplikace, která vložila zprávu)	Není	Prázdný řetězec nebo mezery
PutDate (datum vložení zprávy)	Není	Prázdný řetězec nebo mezery
PutTime (čas vložení zprávy)	Není	Prázdný řetězec nebo mezery
ApplOriginData (data aplikace vztahující se k původu)	Není	Prázdný řetězec nebo mezery
Poznámka: Zbývající pole jsou ignorována, pokud je *Version* menší než MQMD_VERSION_2.		
GroupId (identifikátor skupiny)	MQGI_NONE	Hodnoty null
MsgSeqNumber (pořadové číslo logické zprávy ve skupině)	Není	1
Posunutí (posunutí dat ve fyzické zprávě od začátku logické zprávy)	Není	0
Pole MQMD- MsgFlags (příznaky zprávy)	MQMF_NONE	0
OriginalLength (délka původní zprávy)	MQOL_UNDEFINED	-1

Tabulka 500. Pole v MQMD pro MQMD (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**

Notes:

1. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyku C se jedná o proměnnou makra.MQMD_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQMD MyMD = {MQMD_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQMD

```
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 */
};
```

Deklarace jazyka COBOL pro MQMD

```
**   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.
```

Deklarace PL/I pro MQMD

```
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 */
```

Deklarace High Level Assembler pro MQMD

```
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)
```

Vizuální základní deklarace pro MQMD

```
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
```

StrucId (MQCHAR4) pro MQMD

Jedná se o identifikátor struktury struktury deskriptoru zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQMD_STRUC_ID.

Hodnota musí být:

ID_STRUC_MQMD_STRUC_ID
Identifikátor pro strukturu deskriptoru zprávy.

Pro programovací jazyk C je definována také konstanta MQMD_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQMD_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQMD

Toto je číslo verze struktury a musí být jedno z následujících:

MQMD_VERSION_1
Version-1 struktura deskriptoru zpráv.

Tato verze je podporována ve všech prostředích.

MQMD_VERSION_2
Version-2 struktura deskriptoru zpráv.

Tato verze je podporována ve všech prostředích produktu IBM MQ V6.0 a novějších a k těmto systémům je připojen produkt IBM MQ MQI clients .

Poznámka: Při použití version-2 MQMD provádí správce front další kontroly struktur záhlaví MQ , které mohou být přítomny na začátku dat zprávy aplikace; další podrobnosti naleznete v poznámkách k použití pro volání MQPUT.

Pole, která existují pouze v novější verzi struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MQMD_CURRENT_VERSION
Aktuální verze struktury deskriptoru zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQMD_VERSION_1.

Sestava (MQLONG) pro MQMD

Zpráva sestavy je zpráva o jiné zprávě, která se používá k informování aplikace o očekávaných nebo neočekávaných událostech souvisejících s původní zprávou. Pole *Report* umožňuje aplikaci odesílající původní zprávu určit, které zprávy sestavy jsou povinné, zda do nich mají být zahrnuta data zprávy aplikace a také (pro sestavy i odpovědi), jak mají být nastaveny identifikátory zprávy a korelace v sestavě nebo zprávě odpovědi. Lze požadovat jakýkoli nebo všechny (nebo žádný) z následujících typů zpráv sestavy:

- Výjimka

- Konec platnosti
- Potvrdit při příjezdu (COA)
- Potvrdit při doručení (COD)
- Pozitivní oznámení akce (PAN)
- Negativní oznámení akce (NAN)

Můžete uvést jednu nebo více těchto voleb. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Aplikace, která obdrží zprávu sestavy, může určit příčinu, proč byla sestava vygenerována, prozkoumáním pole *Feedback* v deskriptoru MQMD. Další podrobnosti viz pole *Feedback* .

Použití voleb sestavy při vkládání zprávy do tématu může způsobit, že se vygeneruje nula, jedna nebo mnoho zpráv sestavy a odešlou se do aplikace. Důvodem je skutečnost, že publikační zprávu lze odeslat na nulu, jednu nebo více odebírajícím aplikacím.

Volby výjimky: Zadejte jednu z voleb uvedených pro vyžádání zprávy sestavy výjimek.

MQRO_EXCEPTION
Agent kanálu zpráv generuje tento typ sestavy, když je zpráva odeslána jinému správci front a zprávu nelze doručit do určené cílové fronty. Například cílová fronta nebo mezilehlá přenosová fronta mohou být plné nebo zpráva může být pro frontu příliš velká.

Generování zprávy sestavy výjimek závisí na perzistenci původní zprávy a na rychlosti kanálu zpráv (normální nebo rychlé), kterým původní zpráva prochází:

- Pro všechny trvalé zprávy a pro přechodné zprávy procházející běžnými kanály zpráv je zpráva o výjimce generována pouze v případě, že akci určenou odesílající aplikací pro chybový stav lze úspěšně dokončit. Odesílající aplikace může při výskytu chybového stavu určit jednu z následujících akcí pro řízení dispozice původní zprávy:
  - MQRO_DEAD_LETTER_Q (tím se původní zpráva umístí do fronty nedoručených zpráv).
  - MQRO_DISCARD_MSG (zruší původní zprávu).

  Pokud akci určenou odesílající aplikací nelze úspěšně dokončit, původní zpráva je ponechána v přenosové frontě a není generována žádná zpráva sestavy výjimek.
- Pro přechodné zprávy procházející kanály rychlých zpráv je původní zpráva odebrána z přenosové fronty a sestava výjimek vygenerována *i v případě* , že uvedenou akci pro chybový stav nelze úspěšně dokončit. Je-li například zadána hodnota MQRO_DEAD_LETTER_Q, ale původní zprávu nelze umístit do fronty nedoručených zpráv, protože je tato fronta plná, bude vygenerována zpráva sestavy výjimek a původní zpráva bude zrušena.

  Další informace o běžných a rychlých kanálech zpráv naleznete v tématu Rychlost přechodných zpráv (NPMSPEED).

Zpráva o výjimce se negeneruje, pokud aplikace, která vložila původní zprávu, může být synchronně upozorněna na problém pomocí kódu příčiny vráceného voláním MQPUT nebo MQPUT1 .

Aplikace mohou také odesílat sestavy výjimek, aby označily, že zprávu nelze zpracovat (například proto, že se jedná o debetní transakci, která by způsobila, že by účet překročil svůj limit kreditu).

Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Nezadávejte více než jeden z parametrů MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA a MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_EXCEPTION_WITH_DATA
Toto je stejné jako MQRO_EXCEPTION, ale prvních 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví MQ , jsou zahrnuty do zprávy sestavy kromě 100 bajtů dat aplikace.

Nezadávejte více než jeden z parametrů MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA a MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_EXCEPTION_WITH_FULL_DATA
Jsou vyžadovány sestavy výjimek s úplnými daty.

Jedná se o stejnou zprávu jako MQRO_EXCEPTION s tím rozdílem, že do zprávy sestavy jsou zahrnuta všechna data zprávy aplikace z původní zprávy.

Nezadávejte více než jeden z parametrů MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA a MQRO_EXCEPTION_WITH_FULL_DATA.

Volby vypršení platnosti: Zadejte jednu z voleb uvedených pro vyžádání zprávy sestavy vypršení platnosti.

MQRO_EXPIRATION
Tento typ sestavy je generován správcem front, pokud je zpráva před doručením do aplikace vyřazena, protože uplynula doba vypršení platnosti (viz pole *Expiry*). Není-li tato volba nastavena, není při zrušení zprávy z tohoto důvodu (i když zadáte jednu z voleb MQRO_EXCEPTION_ *) generována žádná zpráva sestavy.

Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Nezadávejte více než jeden z parametrů MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA a MQRO_EXPIRATION_WITH_FULL_DATA.

MQRO_EXPIRATION_WITH_DATA
Toto je stejné jako MQRO_EXPIRATION s tím rozdílem, že prvních 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví MQ , jsou zahrnuty do zprávy sestavy kromě 100 bajtů dat aplikace.

Nezadávejte více než jeden z parametrů MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA a MQRO_EXPIRATION_WITH_FULL_DATA.

MQRO_EXPIRATION_WITH_FULL_DATA
Tato hodnota je stejná jako hodnota MQRO_EXPIRATION s tím rozdílem, že do zprávy sestavy jsou zahrnuta všechna data zprávy aplikace z původní zprávy.

Nezadávejte více než jeden z parametrů MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA a MQRO_EXPIRATION_WITH_FULL_DATA.

Volby potvrzení při příchodu: Zadejte jednu z voleb uvedených v seznamu pro vyžádání zprávy sestavy potvrzení při příchodu.

MQRO_COA

Tento typ sestavy je generován správcem front, který vlastní cílovou frontu, když je zpráva umístěna do cílové fronty. Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Je-li zpráva vložena jako součást pracovní jednotky a cílová fronta je lokální fronta, lze zprávu sestavy COA vygenerovanou správcem front načíst pouze v případě, že je transakce potvrzena.

Sestava COA není generována, pokud je pole *Format* v deskriptoru zprávy MQFMT_XMIT_Q_HEADER nebo MQFMT_DEAD_LETTER_HEADER. To zabraňuje generování sestavy COA, pokud je zpráva vložena do přenosové fronty nebo je nedoručitelná a vložena do fronty nedoručených zpráv.

V případě fronty mostu IMS se sestava COA vygeneruje, když zpráva dosáhne fronty IMS (potvrzení přijaté od IMS). a nikoli, když je zpráva vložena do fronty mostu MQ . To znamená, že pokud IMS není aktivní, žádná sestava COA se nevygeneruje, dokud se nespustí IMS a zpráva se zařadí do fronty IMS .

Uživatel, který spouští program, který vkládá zprávu s produktem MQMD.Report= MQRO_COA musí mít ve frontě odpovědí oprávnění + passid. Pokud uživatel nemá oprávnění + passid, zpráva sestavy COA nedosáhne fronty odpovědí. Došlo k pokusu o vložení zprávy sestavy do fronty nedoručených zpráv.

Nezadávejte více než jednu z hodnot MQRO_COA, MQRO_COA_WITH_DATA a MQRO_COA_WITH_FULL_DATA.

MQRO_COA_WITH_DATA
Toto je stejné jako MQRO_COA, kromě toho, že prvních 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví MQ , jsou zahrnuty do zprávy sestavy kromě 100 bajtů dat aplikace.

Nezadávejte více než jednu z hodnot MQRO_COA, MQRO_COA_WITH_DATA a MQRO_COA_WITH_FULL_DATA.

MQRO_COA_WITH_FULL_DATA
Toto je stejné jako MQRO_COA, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Nezadávejte více než jednu z hodnot MQRO_COA, MQRO_COA_WITH_DATA a MQRO_COA_WITH_FULL_DATA.

Volby potvrzení při doručení: Zadejte jednu z voleb uvedených pro vyžádání zprávy se zprávou o potvrzení při doručení.

MQRO_COD
Tento typ sestavy je generován správcem front, když aplikace načte zprávu z cílové fronty způsobem, který odstraní zprávu z fronty. Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Pokud je zpráva načtena jako součást pracovní jednotky, zpráva sestavy se vygeneruje v rámci stejné pracovní jednotky, takže sestava nebude k dispozici, dokud nebude transakce potvrzena. Je-li jednotka práce vrácena zpět, sestava se neodešle.

Sestava COD není vždy generována, pokud je načtena zpráva s volbou MQGMO_MARK_SKIP_BACKOUT. Pokud je primární jednotka práce vrácena zpět, ale sekundární jednotka práce je potvrzena, zpráva se odebere z fronty, ale sestava COD se nevygeneruje.

Sestava COD není generována, pokud je pole *Format* v deskriptoru zprávy MQFMT_DEAD_LETTER_HEADER. Tím zabráníte generování sestavy COD, pokud je zpráva nedoručitelná a vložená do fronty nedoručených zpráv.

MQRO_COD není platný, pokud je cílová fronta frontou XCF.

Nezadávejte více než jeden z parametrů MQRO_COD, MQRO_COD_WITH_DATA a MQRO_COD_WITH_FULL_DATA.

MQRO_COD_WITH_DATA
Toto je stejné jako MQRO_COD s tím rozdílem, že prvních 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví MQ , jsou zahrnuty do zprávy sestavy kromě 100 bajtů dat aplikace.

Je-li ve volání MQGET pro původní zprávu uvedena hodnota MQGMO_ACCEPT_TRUNCATED_MSG a načtená zpráva je oříznuta, množství dat zprávy aplikace umístěných ve zprávě sestavy závisí na prostředí:

- V systému z/OSse jedná o minimum:
  - Délka původní zprávy
  - Délka vyrovnávací paměti použité k načtení zprávy
  - 100 bajtů.
- V jiných prostředích se jedná o minimum:
  - Délka původní zprávy
  - 100 bajtů.

MQRO_COD_WITH_DATA je neplatný, pokud cílová fronta je fronta XCF.

Nezadávejte více než jeden z parametrů MQRO_COD, MQRO_COD_WITH_DATA a MQRO_COD_WITH_FULL_DATA.

MQRO_COD_WITH_FULL_DATA
Toto je stejné jako MQRO_COD s tím rozdílem, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

MQRO_COD_WITH_FULL_DATA je neplatný, pokud cílová fronta je fronta XCF.

Nezadávejte více než jeden z parametrů MQRO_COD, MQRO_COD_WITH_DATA a MQRO_COD_WITH_FULL_DATA.

Akce-volby oznámení: Zadejte jednu nebo obě vypsané volby, které vyžadují, aby přijímající aplikace odeslala zprávu sestavy kladné nebo záporné akce.

MQRO_PAN
Tento typ sestavy je generován aplikací, která načte zprávu a jedná na ní. Označuje, že akce požadovaná ve zprávě byla úspěšně provedena. Aplikace generující sestavu určuje, zda mají být do sestavy zahrnuta nějaká data.

Kromě předání tohoto požadavku aplikaci, která zprávu načítá, neprovede správce front na základě této volby žádnou akci. Aplikace, která načítá, musí v případě potřeby vygenerovat sestavu.

MQRO_NAN
Tento typ sestavy je generován aplikací, která načte zprávu a jedná na ní. Označuje, že akce požadovaná ve zprávě nebyla úspěšně provedena. Aplikace generující sestavu určuje, zda mají být do sestavy zahrnuta nějaká data. Můžete například chtít zahrnout některá data, která označují, proč nebyl požadavek proveden.

Kromě předání tohoto požadavku aplikaci, která zprávu načítá, neprovede správce front na základě této volby žádnou akci. Aplikace, která načítá, musí v případě potřeby vygenerovat sestavu.

Aplikace musí určit, které podmínky odpovídají pozitivnímu jednání a které negativnímu jednání. Pokud však byl požadavek proveden pouze částečně, vygenerujte sestavu NAN, a nikoli sestavu PAN, pokud je požadována. Každá možná podmínka musí odpovídat buď pozitivní akci, nebo negativní akci, ale ne obojí.

Volby identifikátoru zprávy: Zadejte jednu z uvedených voleb, která řídí, jak se má nastavit *MsgId* zprávy sestavy (nebo zprávy odpovědi).

MQRO_NEW_MSG_ID
Jedná se o výchozí akci a označuje, že pokud je sestava nebo odpověď vygenerována jako výsledek této zprávy, vygeneruje se pro sestavu nebo zprávu odpovědi nový soubor *MsgId* .

MQRO_PASS_MSG_ID
Pokud je sestava nebo odpověď generována jako výsledek této zprávy, *MsgId* této zprávy se zkopíruje do *MsgId* zprávy sestavy nebo odpovědi.

Hodnota *MsgId* zprávy publikování se bude lišit pro každého odběratele, který obdrží kopii publikace, a proto se hodnota *MsgId* zkopírovaná do zprávy sestavy nebo odpovědi bude pro každého z nich lišit.

Není-li tato volba uvedena, předpokládá se MQRO_NEW_MSG_ID.

Volby identifikátoru korelace: Určete jednu z uvedených voleb, která řídí, jak se má nastavit *CorrelId* zprávy sestavy (nebo zprávy odpovědi).

MQRO_COPY_MSG_ID_TO_CORREL_ID
Toto je výchozí akce a označuje, že pokud je sestava nebo odpověď vygenerována jako výsledek této zprávy, *MsgId* této zprávy se zkopíruje do *CorrelId* zprávy sestavy nebo odpovědi.

Hodnota *MsgId* publikační zprávy se bude pro každého odběratele, který obdrží kopii publikace, lišit, a proto se hodnota *MsgId* zkopírovaná do *CorrelId* zprávy sestavy nebo odpovědi bude pro každého z nich lišit.

MQRO_PASS_CORREL_ID
Pokud je sestava nebo odpověď generována jako výsledek této zprávy, *CorrelId* této zprávy se zkopíruje do *CorrelId* zprávy sestavy nebo odpovědi.

Hodnota *CorrelId* zprávy publikování bude specifická pro odběratele, pokud nepoužívá volbu MQSO_SET_CORREL_ID a nenastaví pole ID SubCorrelv tabulce MQSD na hodnotu MQCI_NONE. Proto

je možné, že se soubor *CorrelId* zkopírovaný do souboru *CorrelId* zprávy sestavy nebo odpovědi bude pro každou zprávu lišit.

Není-li tato volba určena, předpokládá se hodnota MQRO_COPY_MSG_ID_TO_CORREL_ID.

Servery, které reagují na požadavky nebo generují zprávy sestavy, musí zkontrolovat, zda byly v původní zprávě nastaveny volby MQRO_PASS_MSG_ID nebo MQRO_PASS_CORREL_ID. Pokud ano, musí servery provést akci popsanou pro tyto volby. Není-li nastavena žádná z těchto hodnot, musí servery provést odpovídající výchozí akci.

Volby odebrání: Zadejte jednu z uvedených voleb pro řízení odebrání původní zprávy, pokud ji nelze doručit do cílové fronty. Aplikace může nastavit volby odebrání nezávisle na vyžádání sestav výjimek.

MQRO_DEAD_LETTER_Q
Jedná se o výchozí akci a pokud zprávu nelze doručit do cílové fronty, umístí ji do fronty nedoručených zpráv. K tomu dochází v následujících situacích:

- Pokud aplikace, která vložila původní zprávu, nemůže být na problém upozorněna synchronně pomocí kódu příčiny vráceného voláním MQPUT nebo MQPUT1 . Pokud byla zpráva o výjimce vyžádána odesilatelem, je vygenerována zpráva o výjimce.
- Když aplikace, která vložila původní zprávu, vkládala do tématu

MQRO_DISCARD_MSG
Tato akce zruší zprávu, pokud ji nelze doručit do cílové fronty. K tomu dochází v následujících situacích:

- Pokud aplikace, která vložila původní zprávu, nemůže být na problém upozorněna synchronně pomocí kódu příčiny vráceného voláním MQPUT nebo MQPUT1 . Pokud byla zpráva o výjimce vyžádána odesilatelem, je vygenerována zpráva o výjimce.
- Když aplikace, která vložila původní zprávu, vkládala do tématu

Chcete-li vrátit původní zprávu odesílateli bez umístění původní zprávy do fronty nedoručených zpráv, musí odesílatel zadat MQRO_DISCARD_MSG s hodnotou MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_PASS_DISCARD_AND_EXPIRAČNÍ
Pokud je tato volba nastavena na zprávu a sestava nebo odpověď je kvůli ní generována, deskriptor zprávy sestavy zdědí:

- MQRO_DISCARD_MSG, pokud byl nastaven.
- Zbývající čas vypršení platnosti zprávy (pokud se nejedná o zprávu o vypršení platnosti). Jedná-li se o zprávu o vypršení platnosti, je doba vypršení platnosti nastavena na 60 sekund.

Volba aktivity

MQRO_ACTIVITY
Pomocí této hodnoty lze trasovat **libovolnou** zprávu v rámci sítě správců front. Volba sestavy může být uvedena v libovolné aktuální uživatelské zprávě, což vám umožní okamžitě začít počítat trasu zprávy přes síť.

Pokud aplikace generující zprávu nemůže povolit generování sestav aktivity, lze vytváření sestav povolit pomocí křížové uživatelské procedury rozhraní API dodané administrátory správce front.

Poznámka:

1. Čím méně správců front v síti, kteří jsou schopni generovat sestavy aktivity, tím méně je trasy podrobná.
2. Sestavy aktivit mohou být obtížné umístit ve správném pořadí, aby bylo možné určit trasu.
3. Sestavy aktivit nemusí být schopny najít trasu k požadovanému cíli.
4. Zprávy s touto sadou voleb sestavy musí být přijímány libovolným správcem front, a to i v případě, že nerozumí této volbě. To umožňuje nastavit volbu sestavy pro libovolnou uživatelskou zprávu, a to i v případě, že jsou zpracovány jiným správcem front nežIBM WebSphere MQ 6.0 nebo novějším.

5. Pokud proces, buď správce front, nebo uživatelský proces, provede aktivitu na zprávě s nastavenou touto volbou, může zvolit generování a vložení sestavy aktivity.

Výchozí volba: Pokud nejsou vyžadovány žádné volby sestavy, zadejte následující:

MQRO_NONE
Pomocí této hodnoty označíte, že nebyly zadány žádné další volby. MQRO_NONE je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby tato volba byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Obecné informace:

1. Všechny požadované typy sestav musí být specificky vyžádány aplikací odesílající původní zprávu. Je-li například požadována sestava COA, ale není-li sestava výjimek, vygeneruje se sestava COA, když je zpráva umístěna do cílové fronty, ale není generována žádná sestava výjimek, pokud je cílová fronta plná, když zpráva dorazí do cílové fronty. Nejsou-li nastaveny žádné volby *Report* , správce front nebo agent kanálu zpráv (MCA) nevygenerují žádné zprávy sestavy.

   Některé volby sestavy lze zadat i v případě, že je lokální správce front nerozpozná. To je užitečné v případě, že má být volba zpracována správcem front *destination* . Další informace viz část "Volby sestavy a příznaky zpráv" na stránce 901.

   Pokud je požadována zpráva sestavy, název fronty, do které se má odeslat sestava, musí být uveden v poli *ReplyToQ* . Když je přijata zpráva sestavy, lze povahu sestavy určit prozkoumáním pole *Feedback* v deskriptoru zprávy.
2. Pokud správce front nebo agent MCA, který generuje zprávu sestavy, nemůže vložit zprávu sestavy do fronty odpovědí (například proto, že fronta odpovědí nebo přenosová fronta jsou plné), bude zpráva sestavy umístěna do fronty nedoručených zpráv. Pokud *také* selže, nebo pokud neexistuje žádná fronta nedoručených zpráv, přijatá akce závisí na typu zprávy sestavy:
   - Pokud je zpráva sestavy sestavou výjimek, zpráva, která vygenerovala zprávu o výjimce, zůstane ve své přenosové frontě; tím se zajistí, že zpráva nebude ztracena.
   - Pro všechny ostatní typy sestav je zpráva sestavy vyřazena a zpracování pokračuje normálně. To se provádí buď proto, že původní zpráva již byla bezpečně doručena (pro zprávy sestavy COA nebo COD), nebo již není zajímavá (pro zprávu sestavy vypršení platnosti).

   Jakmile byla zpráva sestavy úspěšně umístěna do fronty (buď do cílové fronty, nebo do intermediační přenosové fronty), zpráva již není předmětem speciálního zpracování; je s ní zacházeno stejně jako s jakoukoli jinou zprávou.
3. Při generování sestavy se otevře fronta *ReplyToQ* a zpráva sestavy se vloží s použitím oprávnění *UserIdentifier* v deskriptoru MQMD zprávy, která danou sestavu způsobila, s výjimkou následujících případů:
   - Zprávy o výjimkách generované přijímajícím agentem MCA jsou vloženy s jakýmkoli oprávněním, které agent MCA použil při pokusu o vložení zprávy, která tuto zprávu způsobila.
   - Sestavy COA generované správcem front jsou vkládány s libovolným oprávněním, které bylo použito v době, kdy byla zpráva způsobující sestavu vložena do správce front generujícího sestavu. Pokud byla například zpráva vložena přijímajícím agentem MCA s použitím identifikátoru uživatele agenta MCA, správce front vloží sestavu COA s použitím identifikátoru uživatele agenta MCA.

   Aplikace generující sestavy musí používat stejné oprávnění, které používají ke generování odpovědi; toto je obvykle oprávnění identifikátoru uživatele v původní zprávě.

   Pokud má sestava cestovat do vzdáleného cíle, mohou se odesilatelé a příjemci rozhodnout, zda ji přijmou, stejně jako to dělají pro jiné zprávy.
4. Pokud je požadována zpráva sestavy s daty:
   - Zpráva sestavy je vždy generována s množstvím dat požadovaných odesilatelem původní zprávy. Pokud je zpráva sestavy příliš velká pro frontu odpovědí, dojde k výše popsanému zpracování; zpráva sestavy se nikdy nezkrátí, aby se vešla do fronty odpovědí.

- Pokud je *Format* původní zprávy MQFMT_XMIT_Q_HEADER, data obsažená v sestavě nezahrnují MQXQH. Data sestavy začínají prvním bajtem dat za hodnotou MQXQH v původní zprávě. K tomu dochází bez ohledu na to, zda se jedná o přenosovou frontu.

5. Je-li ve frontě odpovědí přijata zpráva COA, COD nebo zpráva o vypršení platnosti, je zaručeno, že původní zpráva byla doručena, byla doručena nebo vypršela její platnost. Avšak pokud je požadována jedna nebo více těchto zpráv sestavy a není přijata, nelze předpokládat opak, protože mohlo dojít k jedné z následujících událostí:

   a. Zpráva sestavy je zadržena, protože odkaz je mimo provoz.

   b. Zpráva sestavy je zadržena, protože v mezilehlé přenosové frontě nebo ve frontě odpovědí existuje blokovací podmínka (například fronta je plná nebo blokovaná pro vložení).

   c. Zpráva sestavy je ve frontě nedoručených zpráv.

   d. Když se správce front pokoušel vygenerovat zprávu sestavy, nemohl ji vložit do příslušné fronty ani do fronty nedoručených zpráv, takže zprávu sestavy nebylo možné vygenerovat.

   e. Došlo k selhání správce front mezi hlášenou akcí (příjem, doručení nebo vypršení platnosti) a generováním odpovídající zprávy sestavy. (To se nestane pro zprávy sestavy COD, pokud aplikace načte původní zprávu v rámci pracovní jednotky, protože zpráva sestavy COD je generována v rámci stejné pracovní jednotky.)

   Zprávy sestavy výjimek mohou být zadrženy stejným způsobem z výše uvedených příčin 1, 2 a 3. Pokud však agent MCA nemůže vygenerovat zprávu o výjimce (zprávu sestavy nelze vložit do fronty odpovědí ani do fronty nedoručených zpráv), původní zpráva zůstane v přenosové frontě u odesilatele a kanál bude uzavřen. K tomu dochází bez ohledu na to, zda má být zpráva sestavy generována na odesílajícím nebo přijímajícím konci kanálu.

6. Pokud je původní zpráva dočasně blokována (což má za následek vygenerování zprávy o výjimce a vložení původní zprávy do fronty nedoručených zpráv), ale zablokování se vymaže a aplikace pak přečte původní zprávu z fronty nedoručených zpráv a znovu ji umístí do svého cíle, může dojít k následujícím:

   - I když byla vygenerována zpráva sestavy výjimek, původní zpráva nakonec úspěšně dorazí na místo určení.
   - Pro jednu původní zprávu je generována více než jedna zpráva sestavy výjimek, protože původní zpráva může později narazit na další zablokování.

Hlásit zprávy při vkládání do tématu:

1. Sestavy lze generovat při vkládání zprávy do tématu. Tato zpráva bude odeslána všem odběratelům tématu, což může být nula, jedna nebo mnoho. To by mělo být zohledněno při výběru použití voleb sestavy, protože v důsledku toho by mohlo být generováno více zpráv sestavy.
2. Při vkládání zprávy do tématu může existovat mnoho cílových front, kterým má být poskytnuta kopie zprávy. Pokud se u některých z těchto cílových front vyskytl problém, například zaplnění fronty, závisí úspěšné dokončení operace MQPUT na nastavení NPMSGDLV nebo PMSGDLV (v závislosti na perzistenci zprávy). Pokud je nastavení takové, že doručení zprávy do cílové fronty musí být úspěšné (například se jedná o trvalou zprávu pro trvalého odběratele a parametr PMSGDLV je nastaven na hodnotu ALL nebo ALLDUR), úspěch je definován jako jedno z následujících kritérií:

   - Úspěšné vložení do fronty odběratele
   - Použití příkazu MQRO_DEAD_LETTER_Q a úspěšného vložení do fronty nedoručených zpráv, pokud fronta odběratele nemůže přijmout zprávu
   - Použití MQRO_DISCARD_MSG, pokud fronta odběratele nemůže přijmout zprávu.

Zprávy sestavy pro segmenty zpráv:

1. Zprávy sestavy lze požadovat pro zprávy, které mají povolenou segmentaci (viz popis příznaku MQMF_SEGMENTATION_ALLOWED). Pokud správce front zjistí, že je nutné zprávu segmentovat, může být pro každý segment, který následně zjistí příslušnou podmínku, vygenerována zpráva sestavy. Aplikace musí být připraveny přijímat více zpráv sestavy pro každý typ požadované zprávy sestavy.

Pole *GroupId* ve zprávě sestavy použijte ke korelaci více sestav s identifikátorem skupiny původní zprávy a pole *Feedback* identifikuje typ každé zprávy sestavy.

2. Pokud je MQGMO_LOGICAL_ORDER použit k načtení zpráv sestavy pro segmenty, mějte na paměti, že sestavy *různých typů* mohou být vráceny následnými voláními MQGET. Pokud jsou například pro zprávu segmentovanou správcem front požadovány sestavy COA i sestavy COD, mohou volání MQGET pro zprávy sestavy vracet zprávy sestav COA a COD prokládané nepředvídatelným způsobem. Vyhněte se tomu pomocí volby MQGMO_COMPLETE_MSG (volitelně s MQGMO_ACCEPT_TRUNCATED_MSG). MQGMO_COMPLETE_MSG způsobí, že správce front znovu sestaví zprávy sestavy, které mají stejný typ sestavy. Například první volání MQGET může znovu sestavit všechny zprávy COA související s původní zprávou a druhé volání MQGET může znovu sestavit všechny zprávy COD. To, které se sestaví jako první, závisí na tom, který typ zprávy sestavy se objeví jako první ve frontě.
3. Aplikace, které samy vkládají segmenty, mohou pro každý segment určit různé volby sestavy. Všimněte si však následujících bodů:
   - Pokud jsou segmenty načteny pomocí volby MQGMO_COMPLETE_MSG, správce front uznává pouze volby sestavy v *prvním* segmentu.
   - Pokud jsou segmenty načteny jeden po druhém a většina z nich má jednu z voleb MQRO_COD_ *, ale alespoň jeden segment nikoli, nemůžete použít volbu MQGMO_COMPLETE_MSG k načtení zpráv sestavy s jedním voláním MQGET nebo použít volbu MQGMO_ALL_SEGMENTS_AVAILABLE ke zjištění, kdy byly všechny zprávy sestavy přijaty.
4. V síti MQ mohou mít správci front různé schopnosti. Pokud je zpráva sestavy pro segment generována správcem front nebo agentem MCA, který nepodporuje segmentaci, správce front nebo agent MCA ve výchozím nastavení nezahrnou nezbytné informace o segmentu do zprávy sestavy, což může ztížit identifikaci původní zprávy, která způsobila vygenerování sestavy. Tomuto problému se vyhněte tím, že si vyžádáte data se zprávou sestavy, tj. uvedením příslušných voleb MQRO_ * _WITH_DATA nebo MQRO_ * _WITH_FULL_DATA. Mějte však na paměti, že pokud je zadána hodnota MQRO_ * _WITH_DATA, *méně než* 100 bajtů dat zprávy aplikace může být vráceno aplikaci, která načte zprávu sestavy, pokud je zpráva sestavy generována správcem front nebo agentem MCA, který nepodporuje segmentaci.

Obsah deskriptoru zprávy pro zprávu sestavy: Když správce front nebo agent kanálu zpráv (MCA) vygeneruje zprávu sestavy, nastaví pole v deskriptoru zprávy na následující hodnoty a pak zprávu vloží běžným způsobem.

Tabulka 501. Hodnoty použité pro pole MQMD, když je zpráva sestavy generována systémem

Pole v deskriptoru MQMD	Použitá hodnota
StrucId	ID_STRUC_MQMD_STRUC_ID
Version	MQMD_VERSION_2
Report	MQRO_NONE
MsgType	MQMT_REPORT
Expiry	MQEI_UNLIMITED
Feedback	Podle povahy sestavy (MQFB_COA, MQFB_COD, MQFB_EXPIRATION nebo hodnota MQRC_ *)
Encoding	Zkopírováno z původního deskriptoru zprávy
CodedCharSetId	Zkopírováno z původního deskriptoru zprávy
Format	Zkopírováno z původního deskriptoru zprávy
Priority	Zkopírováno z původního deskriptoru zprávy
Persistence	Zkopírováno z původního deskriptoru zprávy
MsgId	Jak je určeno volbami sestavy v původním deskriptoru zprávy
CorrelId	Jak je určeno volbami sestavy v původním deskriptoru zprávy

Tabulka 501. Hodnoty použité pro pole MQMD, když je zpráva sestavy generována systémem (pokračování)

Pole v deskriptoru MQMD	Použitá hodnota
BackoutCount	0
ReplyToQ	Mezery
ReplyToQMgr	Název správce front
UserIdentifier	Jak je nastaveno volbou MQPMO_PASS_IDENTITY_CONTEXT
AccountingToken	Jak je nastaveno volbou MQPMO_PASS_IDENTITY_CONTEXT
ApplIdentityData	Jak je nastaveno volbou MQPMO_PASS_IDENTITY_CONTEXT
PutApplType	MQAT_QMGR nebo podle potřeby pro agenta kanálu zpráv
PutApplName	Prvních 28 bajtů názvu správce front nebo názvu agenta kanálu zpráv. Pro zprávy sestavy generované mostem IMS toto pole obsahuje název skupiny XCF a název člena XCF systému IMS , ke kterému se zpráva vztahuje.
PutDate	Datum odeslání zprávy sestavy
PutTime	Čas odeslání zprávy sestavy
ApplOriginData	Mezery
GroupId	Zkopírováno z původního deskriptoru zprávy
MsgSeqNumber	Zkopírováno z původního deskriptoru zprávy
Offset	Zkopírováno z původního deskriptoru zprávy
MsgFlags	Zkopírováno z původního deskriptoru zprávy
OriginalLength	Zkopírováno z původního deskriptoru zprávy, pokud není MQOL_UNDEFINED, a nastaveno na délku původních dat zprávy, jinak

Aplikace generující sestavu se doporučuje nastavit podobné hodnoty, s výjimkou následujících:

- Pole *ReplyToQMgr* může být nastaveno na mezery (správce front jej při vložení zprávy změní na název lokálního správce front).
- Nastavte pole kontextu pomocí volby, která by byla použita pro odpověď, obvykle MQPMO_PASS_IDENTITY_CONTEXT.

Analýza pole sestavy: Pole *Report* obsahuje dílčí pole; kvůli tomu musí aplikace, které potřebují zkontrolovat, zda odesilatel zprávy požadoval určitou sestavu, použít jednu z technik popsaných v části "Analýza pole sestavy" na stránce 903.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQRO_NONE.

MsgType (MQLONG) pro MQMD

Označuje typ zprávy. Typy zpráv jsou seskupeny takto:

MQMT_SYSTEM_FIRST
Nejnižší hodnota pro typy zpráv definované systémem.

MQMT_SYSTEM_LAST
Nejvyšší hodnota pro typy zpráv definované systémem.

Následující hodnoty jsou momentálně definovány v rozsahu systému:

MQMT_DATAGRAM
Zpráva je taková, která nevyžaduje odpověď.

MQMT_REQUEST
Zpráva je zpráva, která vyžaduje odpověď.

Do pole *ReplyToQ* zadejte název fronty, do které se má odeslat odpověď. Pole *Report* uvádí, jak nastavit *MsgId* a *CorrelId* odpovědi.

MQMT_REPLY
Zpráva je odpovědí na dřívější zprávu požadavku (MQMT_REQUEST). Zpráva musí být odeslána do fronty označené polem *ReplyToQ* zprávy požadavku. Pomocí pole *Report* požadavku můžete řídit, jak nastavit *MsgId* a *CorrelId* odpovědi.

Poznámka: Správce front nevynucuje vztah požadavek-odezva. Jedná se o odpovědnost aplikace.

MQMT_REPORT
Zpráva vykazuje očekávaný nebo neočekávaný výskyt, obvykle související s jinou zprávou (například byla přijata zpráva požadavku, která obsahovala neplatná data). Odešlete zprávu do fronty označené polem *ReplyToQ* deskriptoru původní zprávy. Nastavte pole *Feedback* tak, aby označovala povahu sestavy. Pomocí pole *Report* původní zprávy můžete řídit, jak nastavit *MsgId* a *CorrelId* zprávy sestavy.

Zprávy sestav generované správcem front nebo agentem kanálu zpráv jsou vždy odesílány do fronty *ReplyToQ* s nastavenými poli *Feedback* a *CorrelId* , jak je popsáno výše.

Lze také použít hodnoty definované aplikací. Musí být v tomto rozsahu:

MQMT_APPL_FIRST
Nejnižší hodnota pro typy zpráv definované aplikací.

MQMT_APPL_LAST
Nejvyšší hodnota pro typy zpráv definované aplikací.

Pro volání MQPUT a MQPUT1 musí být hodnota *MsgType* buď v rozsahu definovaném systémem, nebo v rozsahu definovaném aplikací; pokud není, volání selže s kódem příčiny MQRC_MSG_TYPE_ERROR.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQMT_DATAGRAM.

Vypršení platnosti (MQLONG) pro MQMD

Jedná se o časové období vyjádřené v desetinách sekundy, nastavené aplikací, která vkládá zprávu. Zpráva se stane způsobilou k vyřazení, pokud nebyla odebrána z cílové fronty před uplynutím této doby.

Chcete-li například nastavit jednu minutu pro čas vypršení platnosti, musíte nastavit **MQMD**.**Expiry** až 600.

Hodnota je snížena tak, aby odrážela čas, který zpráva stráví v cílové frontě, a také v jakýchkoli intermediačních přenosových frontách, pokud je vložení do vzdálené fronty. Může být také dekrementován agenty kanálů zpráv, aby odrážel doby přenosu, pokud jsou významné. Podobně může aplikace, která předává tuto zprávu do jiné fronty, snížit hodnotu, je-li to nutné, pokud tuto zprávu uchovala po dlouhou dobu. Čas vypršení platnosti je však považován za přibližný a hodnota nemusí být snížena, aby odrážela malé časové intervaly.

Když je zpráva načtena aplikací pomocí volání MQGET, pole *Expiry* představuje čas vypršení platnosti, který stále zůstává.

Po uplynutí doby vypršení platnosti zprávy je vhodné ji správce front zrušit. Zpráva je vyřazena, pokud dojde k volání MQGET pro procházení nebo neprocházení, které by vrátilo zprávu, pokud by ještě nevypršela její platnost. Například volání MQGET bez procházení s polem *MatchOptions* v sadě MQGMO na čtení MQMO_NONE z řazené fronty FIFO vyřadí všechny zprávy s vypršenou platností až do první zprávy s vypršenou platností. V případě fronty seřazené podle priority stejné volání zahodí zprávy s vypršenou platností s vyšší prioritou a zprávy se stejnou prioritou, které dorazily do fronty před první nevypršenou zprávou.

Aplikaci se nikdy nevrátí zpráva, jejíž platnost vypršela (buď procházením, nebo voláním MQGET bez procházení), takže hodnota v poli *Expiry* deskriptoru zprávy po úspěšném volání MQGET je buď větší než nula, nebo speciální hodnota MQEI_UNLIMITED.

Pokud je zpráva vložena do vzdálené fronty, může její platnost vypršet (a může být vyřazena), zatímco je ve střední přenosové frontě, než zpráva dosáhne cílové fronty.

Sestava se vygeneruje, když je zpráva s vypršenou platností vyřazena, pokud zpráva zadala jednu z voleb sestavy MQRO_EXPIRATION_ *. Není-li zadána žádná z těchto voleb, žádná taková sestava se nevygeneruje; zpráva se po tomto časovém období již nepovažuje za relevantní (možná proto, že ji později zpráva nahradila).

Pro zprávu vloženou do synchronizačního bodu začíná interval vypršení platnosti v době vložení zprávy, nikoli v době potvrzení synchronizačního bodu. Je možné, že před potvrzením synchronizačního bodu může uplynout interval vypršení platnosti. V tomto případě bude zpráva po operaci potvrzení zrušena a nebude vrácena aplikaci v reakci na operaci MQGET.

Jakýkoli jiný program, který vyřadí zprávy na základě času vypršení platnosti, musí také odeslat odpovídající zprávu sestavy, pokud byla požadována.

Notes:

1. Je-li vložena zpráva s hodnotou *Expiry* čas nula nebo číslo větší než 999 999 999, volání MQPUT nebo MQPUT1 se nezdaří s kódem příčiny MQRC_EXPIRY_ERROR; v tomto případě se nevygeneruje žádná zpráva sestavy.

   Chcete-li povolit kód příčiny 2013, MQRC_EXPIRY_ERROR, musíte povolit proměnnou prostředí AMQ_ENFORCE_MAX_EXPIRY_ERROR.

   Následující text používá příklad pro Linux:

   ```
   $ export AMQ_ENFORCE_MAX_EXPIRY_ERROR=True
   ```

   Všimněte si, že:

   - Důležité je exportovat proměnnou
   - Skutečná hodnota se ignoruje, avšak použití `True` může být užitečné při kontrole nastavení.
2. Vzhledem k tomu, že zpráva s uplynulým časem vypršení platnosti nemusí být vyřazena až později, mohou se ve frontě nacházet zprávy, které překročily čas vypršení platnosti, a které proto nejsou vhodné pro načtení. Tyto zprávy se přesto započítávají do počtu zpráv ve frontě pro všechny účely, včetně spuštění hloubky.

   Pokud se odběratel/spotřebitel (klient) pokusí získat zprávu a této zprávě vypršela platnost, klient neobdrží nic, protože zpráva byla vyřazena, protože byla příliš stará. Kromě toho klient neobdrží žádnou chybovou zprávu.
3. Sestava vypršení platnosti je generována, je-li požadována, když je zpráva vyřazena, ne když se stane způsobilou pro vyřazení.
4. Vyřazení vypršelé zprávy a generování sestavy vypršení platnosti, je-li požadováno, nejsou nikdy součástí jednotky práce aplikace, a to ani v případě, že byla zpráva naplánována pro vyřazení v důsledku volání MQGET fungujícího v rámci jednotky práce.
5. Pokud je zpráva s téměř vypršenou platností načtena voláním MQGET v rámci pracovní jednotky a jednotka práce je následně vrácena zpět, může být zpráva způsobilá k vyřazení, než bude možné ji znovu načíst.
6. Pokud je zpráva s téměř vypršenou platností uzamknuta voláním MQGET s hodnotou MQGMO_LOCK, může se stát, že zpráva bude způsobilá k vyřazení, než ji bude možné načíst voláním MQGET s hodnotou MQGMO_MSG_UNDER_CURSOR; kód příčiny MQRC_NO_MSG_UNDER_CURSOR je vrácen při tomto následném volání MQGET, pokud k tomu dojde.
7. Když je načtena zpráva požadavku s dobou vypršení platnosti větší než nula, aplikace může při odesílání zprávy odpovědi provést jednu z následujících akcí:

   - Zkopírujte zbývající čas vypršení platnosti ze zprávy požadavku do zprávy odpovědi.
   - Nastavte čas vypršení platnosti ve zprávě odpovědi na explicitní hodnotu větší než nula.
   - Nastavte čas vypršení platnosti ve zprávě odpovědi na hodnotu MQEI_UNLIMITED.

Akce, která se má provést, závisí na návrhu aplikace. Výchozí akce pro vložení zpráv do fronty nedoručených zpráv (nedoručených zpráv) však musí být zachování zbývajícího času vypršení platnosti zprávy a pokračovat v jejím snížení.

8. Zprávy spouštěče jsou vždy generovány s hodnotou MQEI_UNLIMITED.
9. Zpráva (obvykle v přenosové frontě), která má název *Format* MQFMT_XMIT_Q_HEADER, má druhý deskriptor zprávy v rámci MQXQH. Proto má k sobě přidružena dvě pole *Expiry* . V tomto případě je třeba poznamenat následující dodatečné body:
   - Když aplikace vloží zprávu do vzdálené fronty, správce front umístí zprávu na počátku do lokální přenosové fronty a před data zprávy aplikace umístí strukturu MQXQH. Správce front nastaví hodnoty dvou polí *Expiry* tak, aby byly stejné jako hodnoty určené aplikací.

     Pokud aplikace vkládá zprávu přímo do lokální přenosové fronty, musí data zprávy již začínat strukturou MQXQH a název formátu musí být MQFMT_XMIT_Q_HEADER. V tomto případě nemusí aplikace nastavit hodnoty těchto dvou polí *Expiry* tak, aby byly stejné. (Správce front kontroluje, zda pole *Expiry* v rámci MQXQH obsahuje platnou hodnotu a zda jsou data zprávy dostatečně dlouhá, aby je bylo možné zahrnout). Pro aplikaci, která může zapisovat přímo do přenosové fronty, musí aplikace vytvořit záhlaví přenosové fronty s vloženým deskriptorem zprávy. Pokud je však hodnota vypršení platnosti v deskriptoru zprávy zapsané do přenosové fronty nekonzistentní s hodnotou ve vloženém deskriptoru zprávy, dojde k odmítnutí chyby vypršení platnosti.
   - Když je zpráva s názvem *Format* MQFMT_XMIT_Q_HEADER načtena z fronty (bez ohledu na to, zda se jedná o normální nebo přenosovou frontu), správce front sníží *obě* tato *Expiry* pole s časem stráveným čekáním na frontu. Pokud data zprávy nejsou dostatečně dlouhá pro zahrnutí pole *Expiry* do MQXQH, dojde k chybě.
   - Správce front používá pole *Expiry* v samostatném deskriptoru zprávy (tj. nikoli v deskriptoru zprávy vloženém ve struktuře MQXQH) k testování, zda je zpráva vhodná pro vyřazení.
   - Pokud se počáteční hodnoty dvou polí *Expiry* liší, může být čas *Expiry* v samostatném deskriptoru zprávy při načtení zprávy větší než nula (takže zpráva není vhodná pro vyřazení), zatímco doba podle pole *Expiry* v MQXQH uplynula. V tomto případě je pole *Expiry* v MQXQH nastaveno na nulu.
10. Doba vypršení platnosti zprávy odpovědi vrácené z mostu IMS je neomezená, pokud není v poli s příznaky MQIIH nastavena hodnota MQIIH_PASS_EXPIRATION. Další informace viz Příznaky .

Je rozpoznána následující speciální hodnota:

MQEI_UNLIMITED
Zpráva má neomezenou dobu vypršení platnosti.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQEI_UNLIMITED.

Vypršelé zprávy na z/OS
V systému IBM MQ for z/OSjsou zprávy, jejichž platnost vypršela, vyřazeny dalším příslušným voláním MQGET.

Pokud však nedojde k žádnému takovému volání, zpráva s vypršenou platností nebude vyřazena a v některých frontách se může hromadit velký počet zpráv s vypršenou platností. Chcete-li to napravit, nastavte správce front tak, aby pravidelně skenoval fronty a zahazoval zprávy s vypršenou platností v jedné nebo více frontách jedním z následujících způsobů:

Periodické skenování
Pomocí atributu správce front EXPRYINT (interval vypršení platnosti) můžete určit období. Pokaždé, když je dosažen interval vypršení platnosti, správce front hledá kandidátní fronty, které stojí za to skenovat, aby vyřadil zprávy s vypršenou platností.

Správce front uchovává informace o zprávách s vypršenou platností v jednotlivých frontách a ví, zda je procházení zpráv s vypršenou platností užitečné. Takže vždy je skenován pouze výběr front.

Sdílené fronty jsou skenovány pouze jedním správcem front ve skupině sdílení front. Obecně se jedná o prvního správce front, který má být restartován, nebo o prvního správce front, který má nastaven

EXPRYINT. Pokud je tento správce front ukončen, převezme skenování front jiný správce front ve skupině sdílení front. Nastavte hodnotu intervalu vypršení platnosti pro všechny správce front v rámci skupiny sdílení front na stejnou hodnotu.

Mějte na zřeteli, že zpracování vypršení platnosti probíhá pro každou frontu při restartování správce front bez ohledu na nastavení EXPRYINT.

Explicitní požadavek
Zadejte příkaz REFRESH QMGR TYPE (EXPIRAČNÍ) s uvedením fronty nebo front, které chcete skenovat.

Vynucení nižšího času vypršení platnosti
Administrátoři mohou omezit dobu vypršení platnosti libovolné zprávy vkládané do fronty nebo tématu pomocí atributu **CAPEXPRY** uvedeného v atributu **CUSTOM** ve frontě nebo tématu.

Důležité: z/OS Nemůžete použít atribut **CAPEXPRY** zavedený v produktu IBM MQ 9.3.1 s klastrem, pokud je úplné úložiště na systému z/OS.

Multi V 9.3.1 V produktu IBM MQ 9.3.1mohou administrátoři omezit dobu vypršení platnosti pomocí atributu **CAPEXPRY** fronty nebo tématu, aniž by jej museli zadat uvnitř atributu **CUSTOM** . Pokud fronta nebo téma již má atribut **CAPEXPRY** nastavený uvnitř atributu **CUSTOM** , musíte před úpravou nového atributu **CAPEXPRY** odebrat atribut **CAPEXPRY** z atributu **CUSTOM** . To můžete provést v jednom příkazu, například:

```
ALTER QLOCAL(Q1) CAPEXPRY(1000) CUSTOM('')
```

Poznámka: Při migraci objektu z dřívější verze produktu je hodnota **CAPEXPRY** nastavena na výchozí hodnotu NOLIMIT. Pokud byl atribut **CAPEXPRY** nastaven uvnitř atributu **CUSTOM** , má tato volba přednost.

Chcete-li použít nový atribut **CAPEXPRY** , musíte nejprve odebrat atribut **CAPEXPRY** z atributu **CUSTOM** . Nastavení obou voleb nefunguje.

Multi V 9.3.1 Revidovaný atribut **CAPEXPRY** nastavený přímo na frontách nebo tématech (ne uvnitř atributu **CUSTOM**) je klastrovaný atribut. Všimněte si, že všechny instance fronty klastru by měly používat stejnou hodnotu pro svůj atribut **CAPEXPRY** . Je stále možné, že přenosová fronta sníží dobu vypršení platnosti zprávy, pokud byl v přenosové frontě nastaven parametr **CAPEXPRY** a hodnota je nižší než atribut **CAPEXPRY** fronty klastru.

Doba vypršení platnosti určená v poli **Expiry** deskriptoru MQMD aplikací, která je větší než hodnota **CAPEXPRY** určená ve frontě nebo tématu, bude nahrazena touto hodnotou **CAPEXPRY** . Použije se čas vypršení platnosti určený aplikací, který je nižší než hodnota **CAPEXPRY** .

Všimněte si, že hodnota **CAPEXPRY** je vyjádřena v desetinách sekund, takže jedna minuta má hodnotu 600.

Pokud je na cestě k rozlišení použit více než jeden objekt, například když je zpráva vložena do aliasu nebo vzdálené fronty, pak se jako horní limit pro vypršení platnosti zprávy použije nejnižší ze všech hodnot **CAPEXPRY** .

Změny hodnot **CAPEXPRY** se projeví okamžitě. Hodnota vypršení platnosti je vyhodnocena pro každé vložení do fronty nebo tématu, a proto je citlivá na rozlišení objektu, které se může u každé operace vložení lišit.

Všimněte si však, že existující zprávy ve frontě před změnou v produktu **CAPEXPRY**nejsou změnou ovlivněny (tj. jejich doba vypršení platnosti zůstává nedotčena). Pouze nové zprávy, které jsou vloženy do fronty po změně v souboru **CAPEXPRY** , mají nový čas vypršení platnosti.

V klastru, v němž je provedeno vložení do fronty otevřené příkazem MQOO_BIND_NOT_FIXED, lze zprávám při každém vložení přiřadit různé hodnoty vypršení platnosti v závislosti na hodnotě **CAPEXPRY** nastavené pro přenosovou frontu používanou kanálem, která odesílá zprávu vybranému cílovému správci front.

Všimněte si, že vložení do fronty nebo tématu aplikací JMS , která uvádí prodlevu doručení, se nezdaří s chybou MQRC_EXPIRY_ERROR, pokud je prodleva doručení delší než vyřešený čas vypršení platnosti pro cílovou frontu nebo téma. Tuto chybu může způsobit atribut **CAPEXPRY** nastavený na frontě vyřešené pro cíl systému JMS .

Poznámka: Soubor **CAPEXPRY** nesmí být použit v žádných frontách, které budou obsahovat IBM MQ interně generované zprávy, jako např. SYSTEM.CLUSTER.* fronty a systému SYSTEM.PROTECTION.POLICY.QUEUE.

Související odkazy

DEFINOVAT fronty

Téma DEFINE

Zpětná vazba (MQLONG) pro MQMD

Pole Zpětná vazba se používá se zprávou typu MQMT_REPORT k označení povahy sestavy a má smysl pouze s tímto typem zprávy.

Pole může obsahovat jednu z hodnot MQFB_ * nebo jednu z hodnot MQRC_ *. Kódy zpětné vazby jsou seskupeny takto:

MQFB_NONE
Nebyla poskytnuta žádná zpětná vazba.

MQFB_SYSTEM_FIRST
Nejnižší hodnota pro zpětnou vazbu generovanou systémem.

MQFB_SYSTEM_LAST
Nejvyšší hodnota pro zpětnou vazbu generovanou systémem.

Rozsah systémem generovaných kódů zpětné vazby MQFB_SYSTEM_FIRST prostřednictvím MQFB_SYSTEM_LAST zahrnuje obecné kódy zpětné vazby uvedené v tomto tématu (MQFB_ *) a také kódy příčiny (MQRC_ *), které se mohou vyskytnout, když zprávu nelze vložit do cílové fronty.

MQFB_APPL_FIRST
Nejnižší hodnota pro zpětnou vazbu generovanou aplikací.

MQFB_APPL_LAST
Nejvyšší hodnota pro zpětnou vazbu generovanou aplikací.

Aplikace, které generují zprávy sestav, nesmí používat kódy zpětné vazby v rozsahu systému (jiném než MQFB_QUIT), pokud nechtějí simulovat zprávy sestav generované správcem front nebo agentem kanálu zpráv.

U volání MQPUT nebo MQPUT1 musí být určená hodnota buď MQFB_NONE, nebo musí být v rozsahu systému nebo aplikace. Tato volba je zaškrtnuta bez ohledu na hodnotu parametru *MsgType*.

Obecné kódy zpětné vazby:

MQFB_COA
Potvrzení o přijetí do cílové fronty (viz MQRO_COA).

MQFB_COD
Potvrzení o doručení do přijímající aplikace (viz MQRO_COD).

MQFB_EXPIRATION
Zpráva byla zrušena, protože nebyla odebrána z cílové fronty před uplynutím doby platnosti.

MQFB_PAN
Pozitivní oznámení akce (viz MQRO_PAN).

MQFB_NAN
Negativní oznámení akce (viz MQRO_NAN).

MQFB_QUIT
Ukončení aplikace.

Tuto možnost může použít plánovací program pracovní zátěže k řízení počtu instancí aplikačního programu, které jsou spuštěny. Odeslání zprávy MQMT_REPORT s tímto kódem zpětné vazby do

instance aplikačního programu označuje této instanci, že by měla zastavit zpracování. Dodržování této konvence je však záležitostí aplikace; správce front ji nevynucuje.

Kódy zpětné vazby kanálu:

MQFB_CHANNEL_DOKONČENO
Kanál byl ukončen normálně.

MQFB_CHANNEL_FAIL-selhání
Kanál byl ukončen nestandardním způsobem a přešel do stavu ZASTAVENO.

MQFB_CHANNEL_FAIL_RETRY
Kanál byl ukončen nestandardně a přejde do stavu RETRY.

IMS-Kódy zpětné vazby mostu

Tyto kódy se používají, když je přijat neočekávaný kód IMS-OTMA sense. Chybový kód, nebo je-li chybový kód 0x1A kód příčiny přidružený k tomuto chybovému kódu, je označen v poli *Zpětná vazba*.

1. Pro kódy *Feedback* v rozsahu MQFB_IMS_FIRST (300) až MQFB_IMS_LAST (399) byl přijat jiný chybový kód než 0x1A . *Kód příčiny* je dán výrazem (*Feedback* - MQFB_IMS_FIRST+1).
2. Pro kódy *Zpětná vazba* v rozsahu MQFB_IMS_NACK_1A_REASON_FIRST (600) až MQFB_IMS_NACK_1A_REASON_LAST (855) byl přijat chybový kód 0x1A . *Kód příčiny* přidružený k kódu příčiny je dán výrazem (*Zpětná vazba* - MQFB_IMS_NACK_1A_REASON_FIRST)

Význam kódů příčiny IMS-OTMA a odpovídajících kódů příčiny je popsán v příručce *Open Transaction Manager Access Guide and Reference*.

Most IMS může generovat následující kódy zpětné vazby:

MQFB_DATA_LENGTH_ZERO
V datech aplikace zprávy byla nulová délka segmentu.

MQFB_DATA_LENGTH_NEGATIVE
Délka segmentu byla v datech aplikace zprávy záporná.

MQFB_DATA_LENGTH_TOO_BIG
Délka segmentu byla v datech aplikace zprávy příliš velká.

MQFB_BUFFER_OVERFLOW
Hodnota jednoho z polí délky způsobí, že data přetečou vyrovnávací paměť zpráv.

MQFB_LENGTH_OFF_BY_ONE
Hodnota jednoho z polí délky byla 1 bajt příliš krátká.

MQFB_IIH_CHYBA
Pole *Format* v MQMD uvádí MQFMT_IMS, ale zpráva nezačíná platnou strukturou MQIIH.

MQFB_NOT_AUTHORIZED_FOR_IMS
ID uživatele obsažené v deskriptoru zprávy MQMD nebo heslo obsažené v poli *Authenticator* ve struktuře MQIIH selhalo při ověření provedeném mostem IMS . V důsledku toho nebyla zpráva předána do adresáře IMS.

MQFB_IMS_ERROR
Funkce IMSvrátila neočekávanou chybu. Další informace o chybě naleznete v protokolu chyb systému IBM MQ , na kterém je umístěn most IMS .

MQFB_IMS_FIRST
Není-li chybový kód IMS-OTMA 0x1A, IMSjsou kódy zpětné vazby generované pomocí v rozsahu MQFB_IMS_FIRST (300) až MQFB_IMS_LAST (399). Samotný chybový kód IMS-OTMA je *Feedback* minus MQFB_IMS_ERROR.

MQFB_IMS_LAST
Nejvyšší hodnota pro zpětnou vazbu generovanou pomocí IMS, když chybový kód není 0x1A.

MQFB_IMS_NACK_1A_REASON_FIRST
Je-li chybový kód 0x1A, IMSkódy zpětné vazby generované pomocí příkazu jsou v rozsahu MQFB_IMS_NACK_1A_REASON_FIRST (600) až MQFB_IMS_NACK_1A_REASON_LAST (855).

MQFB_IMS_NACK_1A_REASON_LAST
Nejvyšší hodnota pro zpětnou vazbu generovanou pomocí IMS, když je chybový kód 0x1A

CICS-bridge feedback codes: Produkt CICS bridgemůže generovat následující kódy zpětné vazby:

MQFB_CICS_APPL_ABENDED
Aplikační program uvedený ve zprávě byl nestandardně ukončen. Tento kód zpětné vazby se vyskytuje pouze v poli *Reason* struktury MQDLH.

MQFB_CICS_APPL_NOT_STARTED
Došlo k selhání EXEC CICS LINK pro aplikační program uvedený ve zprávě. Tento kód zpětné vazby se vyskytuje pouze v poli *Reason* struktury MQDLH.

MQFB_CICS_BRIDGE_FAILURE
Funkce CICS bridge byla nestandardně ukončena bez dokončení normálního zpracování chyb.

MQFB_CICS_CCSID_ERROR
Identifikátor znakové sady není platný.

MQFB_CICS_CIH_ERROR
Struktura záhlaví informací CICS chybí nebo není platná.

MQFB_CICS_COMMAREA_ERROR
Délka CICS COMMAREA není platná.

MQFB_CICS_CORREL_ID_ERROR
Identifikátor korelace není platný.

MQFB_CICS_DLQ_ERROR
Úloha CICS bridge nemohla zkopírovat odpověď na tento požadavek do fronty nedoručených zpráv. Požadavek byl odvolán.

MQFB_CICS_ENCODING_ERROR
Kódování není platné.

MQFB_CICS_INTERNAL_ERROR
Produkt CICS bridge zjistil neočekávanou chybu.

Tento kód zpětné vazby se vyskytuje pouze v poli *Reason* struktury MQDLH.

MQFB_CICS_NOT_AUTHORIZED
Identifikátor uživatele není autorizován nebo heslo není platné.

Tento kód zpětné vazby se vyskytuje pouze v poli *Reason* struktury MQDLH.

MQFB_CICS_UOW_BACKED_OUT
Jednotka práce byla odvolána z jednoho z následujících důvodů:

- Při zpracování jiného požadavku v rámci stejné pracovní jednotky bylo zjištěno selhání.
- Během probíhající jednotky práce se vyskytlo nestandardního ukončení CICS .

MQFB_CICS_UOW_ERROR
Pole řízení jednotky práce *UOWControl* není platné.

Kódy zpětné vazby trasovacích zpráv:

MQFB_ACTIVITY
Používá se s formátem MQFMT_EMBEDDED_PCF k povolení volby uživatelských dat po sestavách aktivity.

MQFB_MAX_ACTIVITY
Vrací se, když je zpráva trasovací trasy vyřazena, protože počet aktivit, ve kterých byla zpráva zahrnuta, překračuje maximální limit aktivit.

MQFB_NOT_POSTOUPENO
Vrací se v případě, že je zpráva trasovací trasy vyřazena, protože má být odeslána vzdálenému správci front, který nepodporuje zprávy trasovací trasy.

MQFB_NEDODÁNO
Vrací se, když je zpráva trasovací trasy vyřazena, protože se má vložit do lokální fronty.

MQFB_UNSUPPORTED_FORWARDING
Vrací se, když je zpráva trasovací trasy vyřazena, protože hodnota v parametru postoupení není rozpoznána a je v zamítnuté bitové masce.

MQFB_UNSUPPORTED_DELIVERY
Vrací se, když je zpráva trasovací trasy vyřazena, protože hodnota v parametru doručení není rozpoznána a je v zamítnuté bitové masce.

IBM MQ kódy příčiny: Pro zprávy sestavy výjimek *Feedback* obsahuje IBM MQ kód příčiny. Mezi možné kódy příčiny patří:

MQRC_PUT_INHIBITED
(2051, X'803 ') Volání Put bylo pro frontu zablokováno.

MQRC_Q_FULL
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_Q_SPACE_NOT_AVAILABLE
(2056, X'808 ') Není k dispozici místo na disku pro frontu.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031, X'7EF') Délka zprávy pro správce front je větší než maximum.

MQRC_MSG_TOO_BIG_FOR_Q
(2030, X'7EE') Délka zprávy je větší než maximum pro frontu.

Úplný seznam kódů příčiny viz:

- Informace o operačním systému IBM MQ for z/OSnaleznete v tématu Kódy příčiny a dokončení rozhraní API.
- Pro všechny ostatní platformy viz Kódy dokončení a příčiny rozhraní API.

.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQFB_NONE.

Kódování (MQLONG) pro MQMD

Tato volba určuje číselné kódování číselných dat ve zprávě; nevztahuje se na číselná data v samotné struktuře MQMD. Číselné kódování definuje reprezentaci používanou pro binární celá čísla, pakovaná desetinná celá čísla a čísla s pohyblivou řádovou čárkou.

V systému z/OSse binární celočíselná část pole `Encoding` také používá k určení celočíselného kódování znakových dat v těle zprávy, když odpovídající identifikátor znakové sady označuje, že reprezentace znakové sady je závislá na kódování použitém pro binární celá čísla. To ovlivňuje pouze některé vícebajtové znakové sady (například znakové sady UTF-16).

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je pole platné. Je definována následující speciální hodnota:

MQENC_NATIVE
Kódování je výchozí pro programovací jazyk a počítač, na kterém je aplikace spuštěna.

Poznámka: Hodnota této konstanty závisí na programovacím jazyku a prostředí. Z tohoto důvodu musí být aplikace kompilovány pomocí souborů záhlaví, makra, COPY nebo INCLUDE odpovídajících prostředí, ve kterém bude aplikace spuštěna.

Aplikace, které vkládají zprávy, obvykle uvádějí MQENC_NATIVE. Aplikace, které načítají zprávy, musí toto pole porovnat s hodnotou MQENC_NATIVE; pokud se hodnoty liší, aplikace bude možná muset převést číselná data ve zprávě. Pomocí volby MQGMO_CONVERT požádejte správce front o převod zprávy v rámci

zpracování volání MQGET. Podrobnosti o tom, jak je pole Encoding sestaveno, viz “Strojové kódování” na stránce 898 .

Pokud pro volání MQGET zadáte volbu MQGMO_CONVERT, jedná se o vstupní/výstupní pole. Hodnota určená aplikací je kódování, do kterého se mají v případě potřeby převést data zprávy. Pokud je převod úspěšný nebo zbytečný, hodnota se nezmění. Pokud je převod neúspěšný, hodnota po volání MQGET představuje kódování nepřevedené zprávy, která je vrácena aplikaci.

V jiných případech se jedná o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG) pro MQMD

Toto pole určuje identifikátor znakové sady znakových dat v těle zprávy.

Poznámka: Znaková data v deskriptoru MQMD a dalších datových strukturách produktu MQ , které jsou parametry volání, musí být ve znakové sadě správce front. Tento atribut je definován atributem **CodedCharSetId** správce front. Podrobnosti o tomto atributu naleznete v části “Atributy pro správce front” na stránce 799 .

Je-li toto pole nastaveno na hodnotu MQCCSI_Q_MGR při volání příkazu MQGET s volbou MQGMO_CONVERT ve volbách, chování se liší mezi aplikacemi klienta a serveru. V případě serverových aplikací je kódovou stránkou používanou pro převod znaků CodedCharSetId správce front; v případě klientských aplikací je kódovou stránkou používanou pro převod znaků aktuální kódová stránka národního prostředí.

V případě klientských aplikací je MQCCSI_Q_MGR vyplněn na základě národního prostředí klienta a nikoli národního prostředí ve správci front. Výjimkou z tohoto pravidla je situace, kdy vložíte zprávu do fronty mostu IMS . Vrácená hodnota v poli *CodedCharSetId* deskriptoru MQMD je CCSID správce front.

Nesmíte použít následující speciální hodnotu:

MQCCSI_APPL
To má za následek chybnou hodnotu v poli CodedCharSetId deskriptoru MQMD a způsobí návratový kód MQRC_SOURCE_CCSID_ERROR (nebo MQRC_FORMAT_ERROR pro z/OS) při přijetí zprávy pomocí volání MQGET s volbou MQGMO_CONVERT.

Můžete použít následující speciální hodnoty:

MQCCSI_Q_MGR
Znaková data ve zprávě jsou ve znakové sadě správce front.

Ve voláních MQPUT a MQPUT1 správce front změní tuto hodnotu v deskriptoru MQMD, který je odeslán spolu se zprávou, na skutečný identifikátor znakové sady správce front. V důsledku toho volání MQGET nikdy nevrátí hodnotu MQCCSI_Q_MGR.

VÝCHOZÍ-MQCCSI_DEFAULT
CodedCharSetId dat v poli *String* je definováno polem CodedCharSetId ve struktuře záhlaví, která předchází struktuře MQCFH, nebo polem CodedCharSetId v deskriptoru MQMD, pokud je MQCFH na začátku zprávy.

MQCCSI_INHERIT
Znaková data ve zprávě jsou ve stejné znakové sadě jako tato struktura; jedná se o znakovou sadu správce front. (Pouze pro MQMD má MQCCSI_INHERIT stejný význam jako MQCCSI_Q_MGR).

Správce front změní tuto hodnotu v deskriptoru MQMD, který je odeslán spolu se zprávou, na identifikátor skutečné znakové sady deskriptoru MQMD. Za předpokladu, že nedojde k žádné chybě, není hodnota MQCCSI_INHERIT vrácena voláním MQGET.

Nepoužívejte MQCCSI_INHERIT, pokud je hodnota pole PutApplType v MQMD MQAT_BROKER.

MQCCSI_EMBEDDED
Znaková data ve zprávě jsou ve znakové sadě s identifikátorem, který je obsažen v samotných datech zprávy. V datech zprávy může být vložen libovolný počet identifikátorů znakové sady, které se vztahují na různé části dat. Tato hodnota musí být použita pro zprávy PCF (ve formátu MQFMT_ADMIN, MQFMT_EVENT nebo MQFMT_PCF), které obsahují data ve směsi znakových sad. Každá struktura

MQCFST, MQCFSL a MQCFSF obsažená ve zprávě PCF musí mít uveden explicitní identifikátor znakové sady, nikoli MQCCSI_DEFAULT.

Pokud má zpráva ve formátu MQFMT_EMBEDDED_PCF obsahovat data ve směsi znakových sad, nepoužívejte volbu MQCCSI_EMBEDDED. Místo toho nastavte MQEPH_CCSID_EMBEDDED v poli Příznaky ve struktuře MQEPH. Jedná se o ekvivalent nastavení MQCCSI_EMBEDDED v předchozí struktuře. Pro každou strukturu MQCFST, MQCFSL a MQCFSF obsaženou ve zprávě PCF musí být zadán explicitní identifikátor znakové sady, nikoli MQCCSI_DEFAULT. Další informace o struktuře MQEPH viz "MQEPH-Vložené záhlaví PCF" na stránce 366.

Tuto hodnotu zadejte pouze u volání MQPUT a MQPUT1 . Pokud je zadán ve volání MQGET, zabrání převodu zprávy.

Při volání MQPUT a MQPUT1 správce front změní hodnoty MQCCSI_Q_MGR a MQCCSI_INHERIT v deskriptoru MQMD, který je odeslán se zprávou, jak je popsáno výše, ale nezmění MQMD určené ve volání MQPUT nebo MQPUT1 . Na uvedené hodnotě se neprovádí žádná další kontrola.

Aplikace, které načítají zprávy, musí porovnat toto pole s hodnotou, kterou aplikace očekává; pokud se hodnoty liší, může aplikace potřebovat převést znaková data ve zprávě.

V systému z/OSse pole Encoding deskriptoru MQMD používá k určení celočíselného kódování znakových dat v těle zprávy, když pole CodedCharSetId deskriptoru MQMD označuje, že reprezentace znakové sady je závislá na kódování použitém pro binární celá čísla. V systému Multiplatformsse předpokládá, že pořadí bajtů znakových dat je stejné jako nativní kódování celých čísel pro platformu, kde je spuštěn správce front. Toto se týká pouze určitých vícebajtových znakových sad (například znakových sad UTF-16).

Pokud pro volání MQGET zadáte volbu MQGMO_CONVERT, jedná se o vstupní/výstupní pole. Hodnota uvedená aplikací je identifikátor kódované znakové sady, na který se v případě potřeby převedou data zprávy. Pokud je převod úspěšný nebo zbytečný, hodnota se nezmění (kromě toho, že hodnota MQCCSI_Q_MGR nebo MQCCSI_INHERIT se převede na skutečnou hodnotu). Pokud je převod neúspěšný, hodnota po volání MQGET představuje identifikátor kódované znakové sady nepřevedené zprávy, která je vrácena aplikaci.

Jinak se jedná o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQCCSI_Q_MGR.

Formát (MQCHAR8) pro MQMD

Jedná se o jméno, které odesilatel zprávy používá k označení povahy dat ve zprávě příjemci. Pro název lze zadat libovolné znaky, které jsou obsaženy ve znakové sadě správce front, je však nutné omezit název na následující:

- Velká písmena A až Z
- Číselné číslice 0 až 9

Jsou-li použity jiné znaky, nemusí být možné přeložit název mezi znakovými sadami odesílajících a přijímajících správců front.

Vyplnit název mezerami do délky pole nebo použít znak null k ukončení názvu před koncem pole; s hodnotou null a dalšími znaky se zachází jako s mezerami. Neuvádějte název s úvodními nebo vloženými mezerami. Pro volání MQGET vrátí správce front název vyplněný mezerami na délku pole.

Správce front nekontroluje, zda je název v souladu s výše popsanými doporučeními.

Názvy začínající MQ velkými, malými a smíšenými písmeny mají význam definovaný správcem front. Pro vlastní formáty nepoužívejte názvy začínající těmito písmeny. Vestavěné formáty správce front jsou:

MQFMT_NONE
Povaha dat není definována: data nelze převést při načtení zprávy z fronty pomocí volby MQGMO_CONVERT.

Zadáte-li ve volání MQGET hodnotu MQGMO_CONVERT a znaková sada nebo kódování dat ve zprávě se liší od znakové sady nebo kódování dat v parametru **MsgDesc** , bude zpráva vrácena s následujícími kódy dokončení a příčiny (bez dalších chyb):

- Kód dokončení MQCC_WARNING a kód příčiny MQRC_FORMAT_ERROR, pokud jsou data MQFMT_NONE na začátku zprávy.
- Kód dokončení MQCC_OK a kód příčiny MQRC_NONE, pokud jsou data MQFMT_NONE na konci zprávy (tj. před jednou nebo více strukturami záhlaví MQ). Struktury záhlaví MQ jsou v tomto případě převedeny na požadovanou znakovou sadu a kódování.

Pro programovací jazyk C je definována také konstanta MQFMT_NONE_ARRAY, která má stejnou hodnotu jako MQFMT_NONE, ale je to pole znaků místo řetězce.

MQFMT_ADMIN
Zpráva je požadavek příkazového serveru nebo zpráva odpovědi v programovatelném formátu příkazu (PCF). Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT. Další informace o použití zpráv programovatelného formátu příkazů naleznete v tématu Použití programovatelných formátů příkazů .

Pro programovací jazyk C je definována také konstanta MQFMT_ADMIN_ARRAY; má stejnou hodnotu jako MQFMT_ADMIN, ale je to pole znaků místo řetězce.

MQFMT_CICS:
Data zprávy začínají na záhlaví informací CICS MQCIH, následované daty aplikace. Název formátu dat aplikace je dán polem `Format` ve struktuře MQCIH.

z/OS V systému z/OSzadejte pro volání MQGET volbu MQGMO_CONVERT pro převod zpráv ve formátu MQFMT_CICS.

Pro programovací jazyk C je definována také konstanta MQFMT_CICS_ARRAY, která má stejnou hodnotu jako MQFMT_CICS, ale je to pole znaků místo řetězce.

MQFMT_COMMAND_1
Zpráva je zpráva odpovědi příkazového serveru MQSC obsahující počet objektů, kód dokončení a kód příčiny. Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Pro programovací jazyk C je definována také konstanta MQFMT_COMMAND_1_ARRAY ; má stejnou hodnotu jako MQFMT_COMMAND_1, ale je to pole znaků namísto řetězce.

MQFMT_COMMAND_2
Zpráva je zpráva odpovědi příkazového serveru MQSC obsahující informace o požadovaných objektech. Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Pro programovací jazyk C je definována také konstanta MQFMT_COMMAND_2_ARRAY ; má stejnou hodnotu jako MQFMT_COMMAND_2, ale pole znaků namísto řetězce.

MQFMT_DEAD_LETTER_HEADER
Data zprávy začínají záhlavím nedoručených zpráv MQDLH. Data z původní zprávy bezprostředně následují strukturu MQDLH. Název formátu původních dat zprávy je dán polem `Format` ve struktuře MQDLH; podrobnosti o této struktuře viz “MQDLH-Záhlaví nedoručených zpráv” na stránce 354 . Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Sestavy COA a COD nejsou generovány pro zprávy, které mají `Format` MQFMT_DEAD_LETTER_HEADER.

Pro programovací jazyk C je definována také konstanta MQFMT_DEAD_LETTER_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_DEAD_LETTER_HEADER, ale je to pole znaků místo řetězce.

MQFMT_DIST_HEADER
Data zprávy začínají záhlavím distribučního seznamu MQDH; to zahrnuje pole záznamů MQOR a MQPMR. Za záhlavím rozdělovníku mohou následovat další data. Formát případných dalších dat je dán polem *`Format`* ve struktuře MQDH; podrobnosti o této struktuře viz “Záhlaví MQDH-Distribution” na stránce 348 . Zprávy ve formátu MQFMT_DIST_HEADER lze převést, pokud je ve volání MQGET zadána volba MQGMO_CONVERT.

Tento formát je podporován v následujících prostředích:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Pro programovací jazyk C je definována také konstanta MQFMT_DIST_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_DIST_HEADER, ale je to pole znaků namísto řetězce.

MQFMT_EMBEDDED_PCF
Formát zprávy trasování trasy za předpokladu, že hodnota příkazu PCF je nastavena na MQCMD_TRACE_ROUTE. Použití tohoto formátu umožňuje odeslání uživatelských dat spolu se zprávou trasovací trasy, za předpokladu, že jejich aplikace mohou zvládnout předchozí parametry PCF.

Záhlaví PCF musí být prvním záhlavím, jinak nebude zpráva považována za zprávu trasovací trasy. To znamená, že zpráva nemůže být ve skupině a že zprávy trasovací trasy nemohou být segmentovány. Pokud je zpráva trasovací trasy odeslána ve skupině, zpráva je odmítnuta s kódem příčiny MQRC_MSG_NOT_ALLOWED_IN_GROUP.

Všimněte si, že MQFMT_ADMIN lze také použít pro formát zprávy trasovací trasy, ale v tomto případě nelze spolu se zprávou trasovací trasy odeslat žádná uživatelská data.

MQFMT_EVENT
Zpráva je zpráva události produktu MQ , která hlásí událost, k níž došlo. Zprávy událostí mají stejnou strukturu jako programovatelné příkazy; další informace o této struktuře viz Zprávy příkazů PCF a Monitorování událostí , kde získáte informace o událostech.

Zprávy událostí Version-1 lze převést ve všech prostředích, pokud je ve volání MQGET zadána volba MQGMO_CONVERT. Zprávy událostí Version-2 lze převést pouze v systému z/OS.

Pro programovací jazyk C je definována také konstanta MQFMT_EVENT_ARRAY; má stejnou hodnotu jako MQFMT_EVENT, ale je to pole znaků místo řetězce.

MQFMT_IMS
Data zprávy začínají záhlaví informací IMS MQIIH, za nímž následují data aplikace. Název formátu dat aplikace je dán polem `Format` ve struktuře MQIIH.

Podrobnosti o způsobu zpracování struktury MQIIH při použití příkazu MQGET s volbou MQGMO_CONVERT naleznete v části “Formát (MQCHAR8) pro MQIIH” na stránce 413 a “ReplyToFormát (MQCHAR8) pro MQIIH” na stránce 414.

Pro programovací jazyk C je definována také konstanta MQFMT_IMS_ARRAY, která má stejnou hodnotu jako MQFMT_IMS, ale je to pole znaků místo řetězce.

MQFMT_IMS_VAR_STRING
Zpráva je řetězec proměnné IMS , což je řetězec ve tvaru `ll`*`zz`*`ccc`, kde:

`ll`
je 2bajtové pole délky určující celkovou délku položky řetězce proměnné IMS . Tato délka se rovná délce `ll` (2 bajty) plus délce *`zz`* (2 bajty) plus délce samotného znakového řetězce. `ll` je dvoubajtové binární celé číslo v kódování uvedeném v poli `Encoding` .

`zz`
je 2bajtové pole obsahující příznaky, které jsou důležité pro IMS. *`zz`* je bajtový řetězec skládající se ze dvou polí MQBYTE a je přenášen beze změny z odesilatele na příjemce (to znamená, že *`zz`* není předmětem žádného převodu).

`ccc`
je znakový řetězec s proměnnou délkou obsahující `ll-4` znaků. *`ccc`* je ve znakové sadě uvedené v poli `CodedCharSetId` .

V systému z/OSmohou data zprávy sestávat z posloupnosti řetězců proměnné IMS , které jsou společně s každým řetězcem ve tvaru `llzzccc`. Mezi následujícími řetězci proměnných IMS nesmí být vynechány žádné bajty. To znamená, že pokud má první řetězec lichou délku, druhý řetězec bude nesprávně zarovnán, to znamená, že nebude začínat na hranici, která je násobkem dvou. Dávejte pozor při konstrukci takových řetězců na strojích, které vyžadují zarovnání elementárních datových typů.

Pomocí volby MQGMO_CONVERT ve volání MQGET převeďte zprávy ve formátu MQFMT_IMS_VAR_STRING.

Pro programovací jazyk C je definována také konstanta MQFMT_IMS_VAR_STRING_ARRAY; má stejnou hodnotu jako MQFMT_IMS_VAR_STRING, ale je to pole znaků místo řetězce.

MQFMT_MD_EXTENSION
Data zprávy začínají příponou MQMDE deskriptoru zprávy a jsou volitelně následována dalšími daty (obvykle daty zprávy aplikace). Název formátu, znaková sada a kódování dat, která následují za MQMDE, jsou dány poli `Format`, `CodedCharSetId`a `Encoding` v MQMDE. Podrobnosti o této struktuře viz “MQMDE-Rozšíření deskriptoru zpráv” na stránce 473 . Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Pro programovací jazyk C je definována také konstanta MQFMT_MD_EXTENSION_ARRAY; má stejnou hodnotu jako MQFMT_MD_EXTENSION, ale je to pole znaků místo řetězce.

MQFMT_PCF
Zpráva je uživatelem definovaná zpráva, která je v souladu se strukturou zprávy PCF (programmable command format). Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT. Další informace o použití zpráv programovatelného formátu příkazů naleznete v tématu Použití programovatelných formátů příkazů .

Pro programovací jazyk C je definována také konstanta MQFMT_PCF_ARRAY; má stejnou hodnotu jako MQFMT_PCF, ale je to pole znaků místo řetězce.

MQFMT_REF_MSG_HEADER
Data zprávy začínají záhlavím referenční zprávy MQRMH a jsou volitelně následována dalšími daty. Název formátu, znaková sada a kódování dat jsou dány poli `Format`, `CodedCharSetId`a `Encoding` v MQRMH. Podrobnosti o této struktuře viz “MQRMH-Záhlaví referenční zprávy” na stránce 550 . Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Tento formát je podporován v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Pro programovací jazyk C je definována také konstanta MQFMT_REF_MSG_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_REF_MSG_HEADER, ale je to pole znaků místo řetězce.

MQFMT_RF_HEADER
Data zprávy začínají pravidly a formátováním záhlaví MQRFH a jsou volitelně následována dalšími daty. Název formátu, znaková sada a kódování dat (jsou-li nějaká) jsou dány poli `Format`, `CodedCharSetId`a `Encoding` v MQRFH. Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Pro programovací jazyk C je definována také konstanta MQFMT_RF_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_RF_HEADER, ale je to pole znaků místo řetězce.

MQFMT_RF_HEADER_2
Data zprávy začínají pravidly version-2 a formátováním záhlaví MQRFH2a jsou volitelně následována dalšími daty. Název formátu, znaková sada a kódování případných volitelných dat jsou dány poli

`Format`, `CodedCharSetId`a `Encoding` ve struktuře MQRFH2. Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Pro programovací jazyk C je definována také konstanta MQFMT_RF_HEADER_2_ARRAY ; má stejnou hodnotu jako MQFMT_RF_HEADER_2, ale je to pole znaků místo řetězce.

MQFMT_STRING
Data zprávy aplikace mohou být buď SBCS řetězec (jednobajtová znaková sada), nebo DBCS řetězec (dvoubajtová znaková sada). Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Pro programovací jazyk C je definována také konstanta MQFMT_STRING_ARRAY; má stejnou hodnotu jako MQFMT_STRING, ale je to pole znaků místo řetězce.

MQFMT_TRIGGER
Zpráva je zpráva spouštěče popsaná strukturou MQTM. Podrobnosti o této struktuře viz “MQTM-zpráva spouštěče” na stránce 601 . Zprávy v tomto formátu lze převést, pokud je ve volání MQGET určena volba MQGMO_CONVERT.

Pro programovací jazyk C je definována také konstanta MQFMT_TRIGGER_ARRAY; má stejnou hodnotu jako MQFMT_TRIGGER, ale je to pole znaků místo řetězce.

MQFMT_WORK_INFO_HEADER
Data zprávy začínají záhlavím MQWIH, za nímž následují data aplikace. Název formátu dat aplikace je dán polem `Format` ve struktuře MQWIH.

z/OS V systému z/OSzadejte pro volání MQGET volbu MQGMO_CONVERT a převeďte uživatelská data ve zprávách ve formátu MQFMT_WORK_INFO_HEADER. Samotná struktura MQWIH je však vždy vrácena ve znakové sadě a kódování správce front (tj. struktura MQWIH je převedena bez ohledu na to, zda je zadána volba MQGMO_CONVERT).

Pro programovací jazyk C je definována také konstanta MQFMT_WORK_INFO_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_WORK_INFO_HEADER, ale je to pole znaků namísto řetězce.

MQFMT_XMIT_Q_HEADER
Data zprávy začínají záhlavím přenosové fronty MQXQH. Data z původní zprávy bezprostředně následují strukturu MQXQH. Název formátu původních dat zprávy je dán polem `Format` ve struktuře MQMD, které je součástí záhlaví přenosové fronty MQXQH. Podrobnosti o této struktuře viz “MQXQH-záhlaví přenosové fronty” na stránce 621 .

Sestavy COA a COD nejsou generovány pro zprávy, které mají `Format` MQFMT_XMIT_Q_HEADER.

Pro programovací jazyk C je definována také konstanta MQFMT_XMIT_Q_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_XMIT_Q_HEADER, ale je to pole znaků místo řetězce.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

Priorita (MQLONG) pro MQMD

Pro volání MQPUT a MQPUT1 musí být hodnota větší nebo rovna nule; nula je nejnižší priorita. Lze také použít následující speciální hodnotu:

MQPRI_PRIORITY_AS_Q_DEF

- Pokud se jedná o frontu klastru, priorita zprávy je převzata z atributu **`DefPriority`** definovaného ve správci front *destination* , který vlastní konkrétní instanci fronty, do níž je zpráva umístěna.

  Pokud existuje více instancí fronty klastrů lišících se v tomto atributu, vybere se hodnota od jedné z nich bez toho, že by šlo předpovědět, která to bude. Proto byste měli ve všech instancích tento atribut nastavit na stejnou hodnotu. Není-li to tento případ, je do protokolů správce front generována chybová zpráva AMQ9407. Viz také Jak jsou vyřešeny atributy cílového objektu pro aliasy, vzdálené fronty a fronty klastru?

Hodnota *DefPriority* se zkopíruje do pole *Priority* , když se zpráva umístí do cílové fronty. Pokud se *DefPriority* následně změní, zprávy, které již byly umístěny do fronty, nebudou ovlivněny.

- Pokud fronta není frontou klastru, priorita zprávy je převzata z atributu **DefPriority** definovaného v *lokálním* správci front, a to i v případě, že je cílový správce front vzdálený.

  Pokud je v cestě k rozlišení názvu fronty více než jedna definice, výchozí priorita je převzata z hodnoty tohoto atributu v *první* definici v cestě. Může se jednat o:

  – Alias fronty
  – Lokální fronta
  – Lokální definice vzdálené fronty
  – Alias správce front
  – Přenosová fronta (například fronta *DefXmitQName*).

  Hodnota *DefPriority* se při vložení zprávy zkopíruje do pole *Priority* . Pokud se *DefPriority* následně změní, zprávy, které již byly vloženy, nebudou ovlivněny.

Hodnota vrácená voláním MQGET je vždy větší nebo rovna nule; hodnota MQPRI_PRIORITY_AS_Q_DEF není nikdy vrácena.

Pokud je vložena zpráva s prioritou vyšší, než je maximální hodnota podporovaná lokálním správcem front (toto maximum je dáno atributem správce front **MaxPriority**), je zpráva přijata správcem front, ale umístěna do fronty s maximální prioritou správce front; volání MQPUT nebo MQPUT1 je dokončeno s hodnotou MQCC_WARNING a kódem příčiny MQRC_PRIORITY_PŘEKROČENÍ maximální hodnoty. V poli *Priority* je však zachována hodnota určená aplikací, která zprávu vložila.

Pokud je v systému z/OSvložena zpráva s číslem MsgSeq1 do fronty, která má posloupnost doručení zprávy MQMDS_PRIORITY a typ indexu MQIT_GROUP_ID, fronta může zacházet se zprávou s jinou prioritou. Pokud byla zpráva umístěna do fronty s prioritou 0 nebo 1, bude zpracována tak, jako by měla prioritu 2. Důvodem je, že pořadí zpráv umístěných do tohoto typu fronty je optimalizováno tak, aby umožňovaly efektivní skupinové testy úplnosti. Další informace o posloupnosti doručení zpráv MQMDS_PRIORITY a typu indexu MQIT_GROUP_ID naleznete v tématu MsgDeliverySequence attribute.

Při odpovídání na zprávu musí aplikace použít prioritu zprávy požadavku pro zprávu odpovědi. V jiných situacích lze zadáním parametru MQPRI_PRIORITY_AS_Q_DEF povolit ladění priority bez změny aplikace.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQPRI_PRIORITY_AS_Q_DEF.

Perzistence (MQLONG) pro MQMD

Označuje, zda zpráva přežívá selhání systému a restartování správce front. Pro volání MQPUT a MQPUT1 musí být hodnota jedna z následujících:

MQPER_PERSISTENT
Zpráva přežije selhání systému a restartuje správce front. Jakmile je zpráva vložena a jednotka práce, do které byla vložena, byla potvrzena (pokud je zpráva vložena jako součást jednotky práce), je zpráva uchována v pomocné paměti. Zůstane tam, dokud nebude zpráva odebrána z fronty, a jednotka práce, ve které byla přijata, byla potvrzena (pokud je zpráva načtena jako součást jednotky práce).

Je-li do vzdálené fronty odeslána trvalá zpráva, mechanizmus uložit-a-předat uchovává zprávu v každém správci front na trase do místa určení, dokud není známo, že zpráva dorazila do dalšího správce front.

Trvalé zprávy nelze umístit na:

- Dočasné dynamické fronty
- Sdílené fronty, které se mapují na objekt CFSTRUCT na úrovni CFLEVEL (2) nebo níže, nebo kde je objekt CFSTRUCT definován jako RECOVER (NO).

Trvalé zprávy lze umístit do trvalých dynamických front a do předdefinovaných front.

MQPER_NOT_PERSISTENT
Zpráva obvykle nepřežije selhání systému nebo restartování správce front. To platí i v případě, že je při restartování správce front v pomocné paměti nalezena neporušená kopie zprávy.

V případě front NPMCLASS (HIGH) přechodné zprávy přežijí normální ukončení a restart správce front.

V případě sdílených front přechodné zprávy přežijí restartování správce front ve skupině sdílení front, ale nepřežijí selhání prostředku Coupling Facility používaného pro ukládání zpráv ve sdílených frontách.

MQPER_PERSISTENCE_AS_Q_DEF

- Pokud se jedná o frontu klastru, perzistence zprávy je převzata z atributu **DefPersistence** definovaného ve správci front *destination* , který vlastní konkrétní instanci fronty, do níž je zpráva umístěna.

  Pokud existuje více instancí fronty klastrů lišících se v tomto atributu, vybere se hodnota od jedné z nich bez toho, že by šlo předpovědět, která to bude. Proto byste měli ve všech instancích tento atribut nastavit na stejnou hodnotu. Není-li to tento případ, je do protokolů správce front generována chybová zpráva AMQ9407. Viz také Jak jsou vyřešeny atributy cílového objektu pro aliasy, vzdálené fronty a fronty klastru?

  Hodnota *DefPersistence* se zkopíruje do pole *Persistence* , když se zpráva umístí do cílové fronty. Pokud se *DefPersistence* následně změní, zprávy, které již byly umístěny do fronty, nebudou ovlivněny.
- Pokud fronta není frontou klastru, je perzistence zprávy převzata z atributu **DefPersistence** definovaného v *lokálním* správci front, a to i v případě, že je cílový správce front vzdálený.

  Pokud je v cestě k rozlišení názvu fronty více než jedna definice, výchozí perzistence je převzata z hodnoty tohoto atributu v *první* definici v cestě. Může se jednat o:

  – Alias fronty
  – Lokální fronta
  – Lokální definice vzdálené fronty
  – Alias správce front
  – Přenosová fronta (například fronta *DefXmitQName*).

  Hodnota *DefPersistence* se při vložení zprávy zkopíruje do pole *Persistence* . Pokud se *DefPersistence* následně změní, zprávy, které již byly vloženy, nebudou ovlivněny.

Ve stejné frontě mohou existovat trvalé i přechodné zprávy.

Při odpovídání na zprávu musí aplikace použít perzistenci zprávy požadavku pro zprávu odpovědi.

Pro volání MQGET je vrácená hodnota buď MQPER_PERSISTENT, nebo MQPER_NOT_PERSISTENT.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQPER_PERSISTENCE_AS_Q_DEF.

MsgId (MQBYTE24) pro MQMD

Jedná se o bajtový řetězec, který se používá k odlišení jedné zprávy od druhé. Obecně platí, že žádné dvě zprávy by neměly mít stejný identifikátor zprávy, ačkoli to správce front nepovoluje. Identifikátor zprávy je trvalou vlastností zprávy a přetrvává po restartování správce front. Vzhledem k tomu, že identifikátor zprávy je bajtový řetězec a nikoli znakový řetězec, není identifikátor zprávy převeden mezi znakovými sadami, pokud zpráva přechází z jednoho správce front do jiného.

Pro volání MQPUT a MQPUT1 platí, že pokud aplikace zadá MQMI_NONE nebo MQPMO_NEW_MSG_ID, správce front vygeneruje jedinečný identifikátor zprávy.[3]když je zpráva vložena a umístí ji do deskriptoru

[3] *MsgId* generovaný správcem front se skládá z 4bajtového identifikátoru produktu (AMQ¬ nebo CSQ¬ v ASCII nebo EBCDIC, kde ¬ představuje prázdný znak), následovaného implementací jedinečného řetězce specifickou pro produkt. V produktu IBM MQ obsahuje prvních 12 znaků názvu správce front a hodnotu

zprávy odeslaného se zprávou. Správce front také vrací tento identifikátor zprávy v deskriptoru zprávy, který náleží odesílající aplikaci. Aplikace může tuto hodnotu použít k zaznamenávání informací o konkrétních zprávách a k odpovídání na dotazy z jiných částí aplikace.

Pokud je zpráva vkládána do tématu, správce front generuje jedinečné identifikátory zpráv podle potřeby pro každou publikovanou zprávu. Pokud je hodnota MQPMO_NEW_MSG_ID určena aplikací, správce front vygeneruje jedinečný identifikátor zprávy, který se má vrátit při výstupu. Pokud je hodnota MQMI_NONE určena aplikací, hodnota pole *MsgId* v deskriptoru MQMD se při návratu z volání nezmění.

Další podrobnosti o zachovaných publikacích naleznete v popisu položky MQPMO_RETAIN v souboru "Volby (MQLONG) pro MQPMO" na stránce 507 .

Pokud je zpráva vkládána do distribučního seznamu, správce front generuje jedinečné identifikátory zpráv podle potřeby, ale hodnota pole *MsgId* v MQMD se při návratu z volání nezmění, a to i v případě, že byla zadána hodnota MQMI_NONE nebo MQPMO_NEW_MSG_ID. Pokud aplikace potřebuje znát identifikátory zpráv generované správcem front, musí poskytnout záznamy MQPMR obsahující pole *MsgId* .

Odesílající aplikace může také zadat jinou hodnotu identifikátoru zprávy než MQMI_NONE; tím se zastaví generování jedinečného identifikátoru zprávy správcem front. Aplikace, která předává zprávu, ji může použít k šíření identifikátoru zprávy původní zprávy.

Správce front toto pole nepoužívá s výjimkou následujících:

- Generovat jedinečnou hodnotu, je-li požadována, jak je popsáno výše
- Dodat hodnotu do aplikace, která vydá požadavek na získání pro zprávu
- Zkopírujte hodnotu do pole *CorrelId* libovolné zprávy sestavy, kterou generuje o této zprávě (v závislosti na volbách *Report*).

Když správce front nebo agent kanálu zpráv vygeneruje zprávu sestavy, nastaví pole *MsgId* způsobem uvedeným v poli *Report* původní zprávy, buď MQRO_NEW_MSG_ID, nebo MQRO_PASS_MSG_ID. Aplikace, které generují zprávy sestav, to musí také provést.

Pro volání MQGET je *MsgId* jedním z pěti polí, které lze použít k načtení konkrétní zprávy z fronty. Volání MQGET obvykle vrací další zprávu ve frontě, ale konkrétní zprávu lze získat zadáním jednoho nebo více z pěti kritérií výběru v libovolné kombinaci; tato pole jsou:

- *MsgId*
- *CorrelId*
- *GroupId*
- *MsgSeqNumber*
- *Offset*

Aplikace nastaví jedno nebo více z těchto polí na požadované hodnoty a poté nastaví odpovídající volby shody MQMO_ * v poli *MatchOptions* v MQGMO tak, aby používala tato pole jako kritéria výběru. Kandidáty na načtení jsou pouze zprávy, které mají uvedené hodnoty v těchto polích. Výchozí hodnotou pro pole *MatchOptions* (pokud není aplikací změněno) je shoda jak s identifikátorem zprávy, tak s identifikátorem korelace.

V systému z/OSjsou kritéria výběru, která můžete použít, omezena typem indexu použitého pro frontu. Další podrobnosti viz atribut fronty **IndexType** .

Za normálních okolností je vrácená zpráva *první* zpráva ve frontě, která splňuje kritéria výběru. Je-li však zadána hodnota MQGMO_BROWSE_NEXT, vrátí se zpráva *další* zpráva, která splňuje kritéria výběru; skenování této zprávy začíná zprávou *následující* aktuální pozici kurzoru.

---

odvozenou z hodin systému. Všichni správci front, kteří mohou komunikovat, proto musí mít názvy, které se liší v prvních 12 znacích, aby bylo zajištěno, že identifikátory zpráv jsou jedinečné. Schopnost generovat jedinečný řetězec také závisí na tom, že systémové hodiny nejsou měněny dozadu. Chcete-li vyloučit možnost vytvoření identifikátoru zprávy generovaného správcem front, který duplikuje identifikátor generovaný aplikací, musí se aplikace vyvarovat generování identifikátorů s počátečními znaky v rozsahu A až I v kódu ASCII nebo EBCDIC (X'41 'až X'49' a X'C1'až X'C9'). Aplikaci však není bráněno v generování identifikátorů s počátečními znaky v těchto rozsazích.

Poznámka: Fronta je skenována sekvenčně pro zprávu, která splňuje kritéria výběru, takže doby načtení jsou pomalejší, než když nejsou uvedena žádná kritéria výběru, zvláště pokud je třeba skenovat mnoho zpráv před nalezením vhodné zprávy. Výjimkou jsou:

- Multi Volání MQGET pomocí *CorrelId* na 64bitových platformách Multiplatforms, kde index *CorrelId* eliminuje potřebu provádět skutečné sekvenční skenování.
- z/OS volání MQGET od *IndexType* v systému z/OS.

V obou těchto případech je výkon načítání zlepšen.

Další informace o tom, jak se kritéria výběru používají v různých situacích, viz Tabulka 495 na stránce 396 .

Zadání MQMI_NONE jako identifikátoru zprávy má stejný efekt jako neuvedení MQMO_MATCH_MSG_ID, tj. *libovolná* shoda identifikátoru zprávy.

Toto pole je ignorováno, pokud je zadána volba MQGMO_MSG_UNDER_CURSOR v parametru **GetMsgOpts** volání MQGET.

Při návratu z volání MQGET je pole *MsgId* nastaveno na identifikátor vrácené zprávy (pokud existuje).

Lze použít následující speciální hodnotu:

MQMI_NONE
Není uveden žádný identifikátor zprávy.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQMI_NONE_ARRAY; má stejnou hodnotu jako MQMI_NONE, ale je to pole znaků místo řetězce.

Toto je vstupní/výstupní pole pro volání MQGET, MQPUT a MQPUT1 . Délka tohoto pole je dána hodnotou MQ_MSG_ID_LENGTH. Počáteční hodnota tohoto pole je MQMI_NONE.

CorrelId (MQBYTE24) pro MQMD

Pole CorrelId je vlastnost v záhlaví zprávy, kterou lze použít k identifikaci specifické zprávy nebo skupiny zpráv.

Jedná se o bajtový řetězec, který může aplikace použít k přiřazení jedné zprávy k jiné nebo k přiřazení zprávy k jiné práci, kterou aplikace provádí. Identifikátor korelace je trvalou vlastností zprávy a přetrvává po restartování správce front. Vzhledem k tomu, že identifikátor korelace je bajtový řetězec a nikoli znakový řetězec, není identifikátor korelace převeden mezi znakovými sadami, pokud zpráva přechází z jednoho správce front do jiného.

Pro volání MQPUT a MQPUT1 může aplikace zadat libovolnou hodnotu. Správce front odešle tuto hodnotu se zprávou a doručí ji aplikaci, která vydá požadavek na získání zprávy.

Pokud aplikace uvádí MQPMO_NEW_CORREL_ID, správce front vygeneruje jedinečný identifikátor korelace, který je odeslán se zprávou a také vrácen odesílající aplikaci na výstupu z volání MQPUT nebo MQPUT1 .

Identifikátor korelace generovaný správcem front se skládá z 3bajtového identifikátoru produktu (AMQ nebo CSQ v ASCII nebo EBCDIC) následovaného jedním vyhrazeným bajtem a implementací jedinečného řetězce specifickou pro produkt. V produktu IBM MQ obsahuje tento řetězec implementace specifický pro daný produkt prvních 12 znaků názvu správce front a hodnotu odvozenou od systémových hodin. Všichni správci front, kteří mohou komunikovat, proto musí mít názvy, které se liší v prvních 12 znacích, aby se zajistilo, že identifikátory zpráv jsou jedinečné. Schopnost generovat jedinečný řetězec také závisí na tom, že systémové hodiny nejsou měněny dozadu. Chcete-li vyloučit možnost vytvoření identifikátoru zprávy generovaného správcem front, který duplikuje identifikátor generovaný aplikací, musí se aplikace vyvarovat generování identifikátorů s počátečními znaky v rozsahu A až I v kódu ASCII nebo EBCDIC (X'41 'až X'49' a X'C1'až X'C9'). Aplikaci však není bráněno v generování identifikátorů s počátečními znaky v těchto rozsazích.

Tento generovaný identifikátor korelace je uchován se zprávou, pokud je zachován, a používá se jako identifikátor korelace, když je zpráva odeslána jako publikování odběratelům, kteří uvádějí MQCI_NONE v poli ID SubCorrelv MQSD předaném při volání MQSUB. Další podrobnosti o zachovaných publikacích naleznete v tématu Volby MQPMO .

Když správce front nebo agent kanálu zpráv vygeneruje zprávu sestavy, nastaví pole *CorrelId* způsobem určeným v poli *Report* původní zprávy, buď MQRO_COPY_MSG_ID_TO_CORREL_ID, nebo MQRO_PASS_CORREL_ID. Aplikace, které generují zprávy sestav, to musí také provést.

Pro volání MQGET je *CorrelId* jedním z pěti polí, které lze použít k výběru konkrétní zprávy, která má být načtena z fronty. Podrobnosti o tom, jak zadat hodnoty pro toto pole, naleznete v popisu pole *MsgId* .

Zadání MQCI_NONE jako identifikátoru korelace má stejný efekt jako neuvedení MQMO_MATCH_CORREL_ID, tj. *libovolný* identifikátor korelace se bude shodovat.

Je-li v parametru **GetMsgOpts** ve volání MQGET určena volba MQGMO_MSG_UNDER_CURSOR, bude toto pole ignorováno.

Při návratu z volání MQGET je pole *CorrelId* nastaveno na identifikátor korelace vrácené zprávy (pokud existuje).

Lze použít následující speciální hodnoty:

MQCI_NONE
Není uveden žádný identifikátor korelace.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQCI_NONE_ARRAY; má stejnou hodnotu jako MQCI_NONE, ale je to pole znaků namísto řetězce.

MQCI_NEW_SESSION
Zpráva je začátek nové relace.

Tato hodnota je rozpoznána produktem CICS bridge jako označení začátku nové relace, tj. začátku nové posloupnosti zpráv.

Pro programovací jazyk C je definována také konstanta MQCI_NEW_SESSION_ARRAY; má stejnou hodnotu jako MQCI_NEW_SESSION, ale je to pole znaků místo řetězce.

Pro volání MQGET se jedná o vstupní/výstupní pole. Pro volání MQPUT a MQPUT1 se jedná o vstupní pole, pokud není uvedeno MQPMO_NEW_CORREL_ID, a výstupní pole, pokud je uvedeno MQPMO_NEW_CORREL_ID. Délka tohoto pole je dána hodnotou MQ_CORREL_ID_LENGTH. Počáteční hodnota tohoto pole je MQCI_NONE.

Poznámka:

Nelze předat identifikátor korelace publikování v hierarchii. Pole používá správce front.

BackoutCount (MQLONG) pro MQMD

Jedná se o počet případů, kdy byla zpráva dříve vrácena voláním MQGET jako součást pracovní jednotky a následně vrácena zpět. Pomáhá aplikaci zjistit chyby zpracování, které jsou založeny na obsahu zprávy. Tento počet vylučuje volání MQGET, která určují libovolnou z voleb MQGMO_BROWSE_ *.

Přesnost tohoto počtu je ovlivněna atributem fronty **HardenGetBackout** ; viz “Atributy pro fronty” na stránce 836.

V systému z/OShodnota 255 znamená, že zpráva byla vrácena zpět 255 nebo vícekrát; vrácená hodnota nikdy není větší než 255.

Toto je výstupní pole pro volání MQGET. Pro volání MQPUT a MQPUT1 se ignoruje. Počáteční hodnota tohoto pole je 0.

ReplyToQ (MQCHAR48) pro MQMD

Jedná se o název fronty zpráv, do které aplikace, která vydala požadavek na získání pro zprávu, odesílá zprávy MQMT_REPLY a MQMT_REPORT. Název je lokální název fronty, která je definována ve správci front

určeném pomocí *ReplyToQMgr*. Tato fronta nesmí být modelovou frontou, ačkoli odesílající správce front tuto skutečnost při vložení zprávy neověřuje.

Pro volání MQPUT a MQPUT1 nesmí být toto pole prázdné, pokud má pole *MsgType* hodnotu MQMT_REQUEST, nebo pokud pole *Report* požaduje nějaké zprávy sestavy. Avšak uvedená hodnota (nebo nahrazená) se předá aplikaci, která vydá požadavek na získání zprávy, bez ohledu na typ zprávy.

Je-li pole *ReplyToQMgr* prázdné, lokální správce front vyhledá název *ReplyToQ* ve svých vlastních definicích front. Pokud existuje lokální definice vzdálené fronty s tímto názvem, hodnota *ReplyToQ* v přenášené zprávě se nahradí hodnotou atributu **RemoteQName** z definice vzdálené fronty a tato hodnota se vrátí v deskriptoru zprávy, když přijímající aplikace vydá pro zprávu volání MQGET. Pokud lokální definice vzdálené fronty neexistuje, *ReplyToQ* se nezmění.

Je-li uveden název, může obsahovat koncové mezery; první znak null a následující znaky jsou považovány za mezery. Jinak se nekontroluje, zda název splňuje pravidla pojmenování pro fronty. To platí i pro název přenášený, pokud je *ReplyToQ* nahrazen v přenášené zprávě. Jediná kontrola je, že název byl uveden, pokud to okolnosti vyžadují.

Není-li odpověď na frontu požadována, nastavte pole *ReplyToQ* na mezery nebo (v programovacím jazyce C) na nulový řetězec nebo na jednu nebo více mezer následovaných znakem null; nenechávejte pole neinicializované.

Pro volání MQGET vrací správce front vždy název vyplněný mezerami na délku pole.

Pokud zprávu, která vyžaduje zprávu sestavy, nelze doručit a zprávu sestavy také nelze doručit do uvedené fronty, původní zpráva i zpráva sestavy přejdou do fronty nedoručených zpráv (nedoručených zpráv) (viz atribut **DeadLetterQName** popsaný v části “Atributy pro správce front” na stránce 799).

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ReplyToQMgr (MQCHAR48) pro MQMD

Jedná se o název správce front, kterému má být odeslána zpráva odpovědi nebo zpráva sestavy. *ReplyToQ* je lokální název fronty, která je definována v tomto správci front.

Je-li pole *ReplyToQMgr* prázdné, lokální správce front vyhledá název *ReplyToQ* ve svých definicích front. Pokud existuje lokální definice vzdálené fronty s tímto názvem, hodnota *ReplyToQMgr* v přenášené zprávě se nahradí hodnotou atributu **RemoteQMgrName** z definice vzdálené fronty a tato hodnota se vrátí v deskriptoru zprávy, když přijímající aplikace vydá pro zprávu volání MQGET. Pokud lokální definice vzdálené fronty neexistuje, je název *ReplyToQMgr* , který je přenášen se zprávou, názvem lokálního správce front.

Je-li uveden název, může obsahovat koncové mezery; první znak null a následující znaky jsou považovány za mezery. Jinak nebude provedena žádná kontrola, zda název splňuje pravidla pojmenování pro správce front nebo zda je tento název pro odesílajícího správce front znám. To platí i pro přenášený název, pokud je v přenášené zprávě nahrazen parametr *ReplyToQMgr* .

Není-li odpověď na frontu požadována, nastavte pole *ReplyToQMgr* na mezery nebo (v programovacím jazyce C) na nulový řetězec nebo na jednu nebo více mezer následovaných znakem null; nenechávejte pole neinicializované.

Pro volání MQGET vrací správce front vždy název vyplněný mezerami na délku pole.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

UserIdentifier (MQCHAR12) pro MQMD

Toto je část **kontextu identity** zprávy. Další informace o kontextu zprávy viz “MQMD-Deskriptor zpráv” na stránce 424 a Kontext zprávy.

UserIdentifier uvádí identifikátor uživatele aplikace, která zprávu vyvolala. Správce front považuje tyto informace za znaková data, ale nedefinuje jejich formát.

Po přijetí zprávy použijte *UserIdentifier* v poli *AlternateUserId* parametru **ObjDesc** následného volání MQOPEN nebo MQPUT1 k provedení kontroly autorizace pro uživatele *UserIdentifier* namísto aplikace provádějící otevření.

Když správce front vygeneruje tyto informace pro volání MQPUT nebo MQPUT1 :

- V systému z/OSpoužívá správce front parametr *AlternateUserId* z parametru **ObjDesc** volání MQOPEN nebo MQPUT1 , pokud byla zadána volba MQOOO_ALTERNATE_USER_AUTHORITY nebo MQPMO_ALTERNATE_USER_AUTHORITY. Pokud nebyla zadána příslušná volba, použije správce front identifikátor uživatele určený z daného prostředí.
- V jiných prostředích používá správce front vždy identifikátor uživatele určený z daného prostředí.

Když je identifikátor uživatele určen z prostředí:

- V systému z/OSpoužívá správce front:
  - Pro MVS (dávka), identifikátor uživatele z karty JES JOB nebo spuštěné úlohy
  - V případě TSO se identifikátor uživatele šířil do úlohy během odesílání úlohy.
  - V případě systému CICSse jedná o identifikátor uživatele přidružený k úloze.
  - V případě systému IMSzávisí identifikátor uživatele na typu aplikace:
    - Počet:
      - Oblasti bez zprávy BMP
      - Regiony veřejné instituce IFP bez zpráv
      - Zpráva BMP a zpráva oblasti IFP, které nevyvolaly úspěšné volání GU

      Správce front používá identifikátor uživatele z karty JES JOB oblasti nebo identifikátor uživatele TSO. Pokud jsou prázdné nebo mají hodnotu null, použije název bloku specifikace programu (PSB).
    - Počet:
      - Zpráva BMP a zpráva oblastí IFP, které *vydaly* úspěšné volání GU
      - Regiony veřejných zakázek

      správce front používá jednu z následujících možností:
      - Identifikátor přihlášeného uživatele přidružený ke zprávě
      - Název logického terminálu (LTERM)
      - Identifikátor uživatele z oblasti karty JES JOB
      - Identifikátor uživatele TSO
      - Název PSB
- V systému IBM ipoužívá správce front název profilu uživatele přidruženého k úloze aplikace.
- V systému AIX and Linuxpoužívá správce front:
  - Přihlašovací jméno aplikace
  - Efektivní identifikátor uživatele procesu, pokud není k dispozici žádné přihlášení
  - Identifikátor uživatele přidružený k transakci, pokud je aplikace transakcí CICS
- V systémech Windows používá správce front prvních 12 znaků jména přihlášeného uživatele.

Toto pole je obvykle výstupní pole generované správcem front, ale pro volání MQPUT nebo MQPUT1 můžete toto pole nastavit jako vstupní/výstupní pole a zadat pole UserIdentification místo toho, aby správce front tyto informace generoval. Zadejte buď MQPMO_SET_IDENTITY_CONTEXT, nebo MQPMO_SET_ALL_CONTEXT v parametru PutMsgOpts a uveďte ID uživatele v poli UserIdentifier , pokud nechcete, aby správce front generoval pole UserIdentifier pro volání MQPUT nebo MQPUT1 .

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PutMsgOpts** určena hodnota MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT. Jakékoli informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána volba MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor *UserIdentifier* , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude to hodnota *UserIdentifier* , která se spolu se zprávou zachová, pokud je zachována (další podrobnosti o zachovaných publikacích viz popis MQPMO_RETAIN), ale nepoužívá se jako *UserIdentifier* , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání *UserIdentifier* ve všech publikacích, která jim byla odeslána. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 12 prázdných znaků v jiných programovacích jazycích.

AccountingToken (MQBYTE32) pro MQMD

Jedná se o token evidence, který je součástí *kontextu identity* zprávy. Další informace o kontextu zprávy viz “MQMD-Deskriptor zpráv” na stránce 424 a Kontext zprávy.

Produkt `AccountingToken` umožňuje aplikaci účtovat odpovídající poplatek za práci provedenou jako výsledek zprávy. Správce front považuje tyto informace za řetězec bitů a nekontroluje jejich obsah.

Správce front generuje tyto informace následujícím způsobem:

- První bajt pole je nastaven na délku účetních informací přítomných v následujících bajtech; tato délka je v rozsahu od nuly do 30 a je uložena v prvním bajtu jako binární celé číslo.
- Druhý a následující bajty (jak jsou uvedeny v poli délky) jsou nastaveny na informace o účtování odpovídající prostředí.
  - z/OS V systému z/OS jsou informace o účtování nastaveny na:
    - V případě dávky z/OS se jedná o účtovací informace z karty JES JOB nebo z příkazu JES ACCT v kartě EXEC (čárkové oddělovače jsou změněny na X'FF '). Tato informace je v případě potřeby oříznuta na 31 bajtů.
    - V případě TSO se jedná o číslo účtu uživatele.
    - V případě operačního systému CICSse jedná o identifikátor jednotky práce LU 6.2 (UEPUOWDS) (26 bajtů).
    - V případě systému IMSse jedná o 8znakový název PSB zřetězený s 16znakovým tokenem zotavení IMS .
  - IBM i V systému IBM ijsou účtovací informace nastaveny na účtovací kód úlohy.
  - Linux AIX V systému AIX and Linuxjsou účtovací informace nastaveny na číselný identifikátor uživatele ve znacích ASCII.
  - Windows V systému Windowsjsou informace o účtování nastaveny na identifikátor zabezpečení Windows (SID) v komprimovaném formátu. Identifikátor SID jedinečně identifikuje identifikátor uživatele uložený v poli *UserIdentifier* . Je-li identifikátor SID uložen v poli *AccountingToken* , je vynechán 6bajtový identifikátor autority (který se nachází ve třetím a následných bajtech identifikátoru SID). Pokud je například identifikátor SID Windows dlouhý 28 bajtů, do pole *AccountingToken* se uloží 22 bajtů informací o identifikátoru SID.
- Poslední bajt (bajt 32) pole evidence je nastaven na typ tokenu evidence (v tomto případě MQACTT_NT_SECURITY_ID, x '0b'):

  MQACTT_CICS_LUOW_ID
  CICS Identifikátor LUOW.

Windows **MQACTT_NT_SECURITY_ID**
Windows identifikátor zabezpečení.

IBM i **MQACTT_OS400_ACCOUNT_TOKEN**
IBM i token evidence.

UNIX **MQACTT_UNIX_NUMERIC_ID**
UNIX číselný identifikátor.

MQACTT_USER
Token evidence definovaný uživatelem.

MQACTT_UNKNOWN (neznámý)
Neznámý typ účtovacího tokenu.

Typ evidenčního tokenu je nastaven na explicitní hodnotu pouze v následujících prostředích:

- AIX AIX
- IBM i IBM i
- Linux Linux
- Windows Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům. V jiných prostředích je typ tokenu evidence nastaven na hodnotu MQACTT_UNKNOWN. V těchto prostředích použijte pole *PutApplType* k vyvodit typ přijatého účtovacího tokenu.

- Všechny ostatní bajty jsou nastaveny na binární nulu.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PutMsgOpts** určena hodnota MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT. Není-li zadána volba MQPMO_SET_IDENTITY_CONTEXT ani MQPMO_SET_ALL_CONTEXT, je toto pole ve vstupu ignorováno a jedná se o pole pouze pro výstup. Další informace o kontextu zprávy viz Kontext zprávy.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor `AccountingToken` , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude se jednat o hodnotu `AccountingToken` , která je uchována se zprávou, pokud je zachována (další podrobnosti o zachovaných publikacích viz popis MQPMO_RETAIN v souboru “Volby (MQLONG) pro MQPMO” na stránce 507), ale nepoužívá se jako `AccountingToken` , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání `AccountingToken` ve všech publikacích, které jim byly odeslány. Pokud zpráva nemá žádný kontext, pole je zcela binární nula.

Toto je výstupní pole pro volání MQGET.

Toto pole není předmětem žádného překladu na základě znakové sady správce front; pole je považováno za řetězec bitů, a nikoli za řetězec znaků.

Správce front neprovádí žádnou činnost s informacemi v tomto poli. Aplikace musí interpretovat informace, pokud chce tyto informace použít pro účely účtování.

Pro pole `AccountingToken` můžete použít následující speciální hodnotu:

MQACT_NONE
Není uveden žádný token evidence.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQACT_NONE_ARRAY, která má stejnou hodnotu jako MQACT_NONE, ale je to pole znaků namísto řetězce.

Délka tohoto pole je dána hodnotou MQ_ACCOUNTING_TOKEN_LENGTH. Počáteční hodnota tohoto pole je MQACT_NONE.

ApplIdentity-data (MQCHAR32) pro MQMD

Toto je část **kontextu identity** zprávy. Další informace o kontextu zprávy viz "MQMD-Deskriptor zpráv" na stránce 424 a Kontext zprávy.

ApplIdentityData je informace, která je definována sadou aplikací, a lze ji použít k poskytnutí dalších informací o zprávě nebo jejím původci. Správce front považuje tyto informace za znaková data, ale nedefinuje jejich formát. Když správce front generuje tyto informace, jsou zcela prázdné.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PutMsgOpts** určena hodnota MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT. Je-li přítomen znak null, správce front převede hodnotu null a všechny následující znaky na mezery. Není-li zadána volba MQPMO_SET_IDENTITY_CONTEXT ani MQPMO_SET_ALL_CONTEXT, je toto pole ve vstupu ignorováno a jedná se o pole pouze pro výstup. Další informace o kontextu zprávy viz Kontext zprávy.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor *ApplIdentityData* , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude to hodnota *ApplIdentityData* , která se spolu se zprávou zachová, pokud je zachována (další podrobnosti o zachovaných publikacích viz popis MQPMO_RETAIN), ale nepoužívá se jako *ApplIdentityData* , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání *ApplIdentityData* ve všech publikacích, která jim byla odeslána. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou MQ_APPL_IDENTITY_DATA_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 32 prázdných znaků v jiných programovacích jazycích.

PutApplTyp (MQLONG) pro MQMD

Jedná se o typ aplikace, která vložila zprávu, a je součástí **původního kontextu** zprávy. Další informace o kontextu zprávy viz "MQMD-Deskriptor zpráv" na stránce 424 a Kontext zprávy.

Produkt *PutApplType* může mít jeden z následujících standardních typů. Můžete také definovat vlastní typy, ale pouze s hodnotami v rozsahu MQAT_USER_FIRST až MQAT_USER_LAST.

MQAT_AIX-operační systém
Aplikace AIX (stejná hodnota jako MQAT_UNIX).

MQAT_AMQP
Aplikace protokolu AMQP

MQAT_BROKER
Zprostředkovatel.

MQAT_CICS
CICS transakce.

MQAT_CICS_BRIDGE
CICS bridge.

MQAT_CICS_VSE
CICS/VSE transakce.

MQAT_DOS
Aplikace IBM MQ MQI client na PC DOS.

MQAT_DQM
Agent distribuovaného správce front.

MQAT_STRÁŽCE
Aplikace Tandem Guardian (stejná hodnota jako MQAT_NSK).

MQAT_IMS
IMS .

MQAT_IMS_BRIDGE
Most IMS .

MQAT_JAVA
Java.

MQAT_MVS
Aplikace MVS nebo TSO (stejná hodnota jako MQAT_ZOS).

MQAT_NOTES_AGENT
Lotus Notes Aplikace agenta.

MQAT_OS390
Aplikace OS/390 (stejná hodnota jako MQAT_ZOS).

MQAT_OS400
IBM i .

MQAT_QMGR
Správce front.

MQAT_UNIX
UNIX .

MQAT_VOS
Stratus VOS aplikace.

MQAT_WINDOWS
16bitová aplikace Windows .

MQAT_WINDOWS_NT
32bitová aplikace Windows .

MQAT_WLM
z/OS aplikace správce pracovní zátěže.

MQAT_XCF
XCF.

MQAT_ZOS
z/OS .

VÝCHOZÍ
Výchozí typ aplikace.

Jedná se o výchozí typ aplikace pro platformu, na které je aplikace spuštěna.

Poznámka: Hodnota této konstanty je specifická pro dané prostředí. Z tohoto důvodu vždy zkompilujte aplikaci pomocí souborů záhlaví, include nebo COPY, které jsou vhodné pro platformu, na které bude aplikace spuštěna.

MQAT_UNKNOWN (neznámý)
Pomocí této hodnoty označíte, že typ aplikace je neznámý, i když jsou k dispozici další informace o kontextu.

MQAT_USER_FIRST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

MQAT_USER_LAST
Nejvyšší hodnota pro typ aplikace definovaný uživatelem.

Může se také vyskytnout následující speciální hodnota:

MQAT_NO_CONTEXT
Tato hodnota je nastavena správcem front při vložení zprávy bez kontextu (tj. je zadána volba kontextu MQPMO_NO_CONTEXT).

Při načtení zprávy lze pro tuto hodnotu otestovat produkt *PutApplType* a rozhodnout, zda má zpráva kontext (doporučuje se, aby aplikace používající MQPMO_SET_ALL_CONTEXT nikdy nenastavila proměnnou *PutApplType* na hodnotu MQAT_NO_CONTEXT, pokud některá z ostatních polí kontextu není prázdná).

Když správce front vygeneruje tyto informace v důsledku vložení aplikace, pole se nastaví na hodnotu, která je určena prostředím. V systému IBM ije nastaven na hodnotu MQAT_OS400; správce front nikdy nepoužívá MQAT_CICS v systému IBM i.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, je-li v parametru **PutMsgOpts** zadána hodnota MQPMO_SET_ALL_CONTEXT. Není-li zadána hodnota MQPMO_SET_ALL_CONTEXT, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Toto je výstupní pole pro volání MQGET. Počáteční hodnota tohoto pole je MQAT_NO_CONTEXT.

PutApplNázev (MQCHAR28) pro MQMD

Jedná se o název aplikace, která vložila zprávu, a je součástí *původního kontextu* zprávy. Obsah se mezi jednotlivými platformami liší a může se lišit mezi jednotlivými verzemi.

Další informace o kontextu zprávy viz “MQMD-Deskriptor zpráv” na stránce 424 a Kontext zprávy.

V produktu IBM MQ 9.1.2 můžete zadat název aplikace v dalších programovacích jazycích. Další informace viz určení názvu aplikace v podporovaných programovacích jazycích .

Formát proměnné *PutApplName* závisí na hodnotě proměnné *PutApplType* a může se měnit z jednoho vydání na druhé. Změny jsou vzácné, ale stávají se, pokud se změní prostředí.

Když správce front nastaví toto pole (tj. pro všechny volby kromě MQPMO_SET_ALL_CONTEXT), nastaví pole na hodnotu určenou prostředím:

- z/OS V systému z/OSpoužívá správce front:
  - V případě dávky z/OS se jedná o osmiznakový název úlohy z karty JES JOB.
  - V případě TSO se jedná o 7znakový identifikátor uživatele TSO.
  - V případě systému CICSse jedná o osmiznakovou aplikaci následovanou čtyřznakovým identifikátorem tranid.
  - V případě systému IMSse jedná o 8znakový identifikátor systému IMS následovaný 8znakovým názvem PSB.
  - V případě XCF se jedná o 8znakový název skupiny XCF následovaný 16znakovým názvem člena XCF.
  - V případě zprávy generované správcem front se jedná o prvních 28 znaků názvu správce front.
  - V případě distribuovaného řazení do fronty bez produktu CICSse jedná o osmiznakový název úlohy inicializátoru kanálu následovaný osmiznakovým názvem modulu, který se vkládá do fronty nedoručených zpráv, za nímž následuje osmiznakový identifikátor úlohy.

  Každý název nebo názvy jsou vyplněny vpravo mezerami, stejně jako jakákoli mezera ve zbytku pole. Pokud existuje více než jeden název, není mezi nimi žádný oddělovač.
- Windows V systémech Windows používá správce front následující názvy:
  - V případě aplikace CICS se jedná o název transakce CICS .
  - V případě aplikací jiných nežCICS se jedná o 28 znaků zcela vpravo od úplného názvu spustitelného souboru.
- IBM i V systému IBM ipoužívá správce front úplný název úlohy.
- Linux AIX V systému AIX and Linuxpoužívá správce front následující názvy:
  - V případě aplikace CICS se jedná o název transakce CICS .
  - V případě aplikací jiných nežCICS produkt MQ požádá operační systém o název procesu. Vrací se jako název souboru programu bez úplné cesty. Poté produkt MQ umístí tento název procesu do deskriptoru MQMD.PutApplName je následující:

    AIX **AIX**

    Je-li název menší nebo roven 28 bajtům, bude název vložen a vpravo vyplněn mezerami.

    Je-li název větší než 28 bajtů, vloží se 28 bajtů názvu nejvíce vlevo.

    Linux **Linux**

    Je-li název menší nebo roven 15 bajtům, bude název vložen a vpravo vyplněn mezerami.

    Je-li název větší než 15 bajtů, vloží se 15 bajtů názvu zleva doprava s mezerami.

Pokud například spustíte `/opt/mqm/samp/bin/amqsput QNAME QMNAME`, název PutApplbude `'amqsput'`. V tomto poli MQCHAR28 je 21 mezer znaků výplně. Všimněte si, že úplná cesta včetně cesty `/opt/mqm/samp/bin` není zahrnuta v názvu PutAppl.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, je-li v parametru **PutMsgOpts** zadána hodnota MQPMO_SET_ALL_CONTEXT. Jakékoli informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota MQPMO_SET_ALL_CONTEXT, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

PutDate (MQCHAR8) pro MQMD

Toto je datum, kdy byla zpráva vložena, a je součástí **původního kontextu** zprávy. Další informace o kontextu zprávy viz "MQMD-Deskriptor zpráv" na stránce 424 a Kontext zprávy.

Formát použitý pro datum, kdy je toto pole generováno správcem front, je:

- RRRRMMDD

kde znaky představují:

yyyy
rok (čtyři číslice)

MM
měsíc v roce (01 až 12)

DD
den v měsíci (01 až 31)

Greenwichský střední čas (GMT) se používá pro pole *PutDate* a *PutTime* pod podmínkou, že systémové hodiny jsou přesně nastaveny na GMT.

Pokud byla zpráva vložena jako součást pracovní jednotky, datum je datum, kdy byla zpráva vložena, a ne datum, kdy byla pracovní jednotka potvrzena.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, je-li v parametru **PutMsgOpts** zadána hodnota MQPMO_SET_ALL_CONTEXT. Obsah pole není kontrolován správcem front s tím rozdílem, že veškeré informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota MQPMO_SET_ALL_CONTEXT, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou MQ_PUT_DATE_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 8 prázdných znaků v jiných programovacích jazycích.

PutTime (MQCHAR8) pro MQMD

Jedná se o čas, kdy byla zpráva vložena, a je součástí **původního kontextu** zprávy. Další informace o kontextu zprávy viz "MQMD-Deskriptor zpráv" na stránce 424 a Kontext zprávy.

Formát použitý pro čas, kdy je toto pole generováno správcem front, je:

- HHMMSSTH

kde znaky představují (v pořadí):

HH
hodiny (00 až 23)

MM
minut (00 až 59)

SS
sekundy (00 až 59; viz poznámka)

T
desetiny sekundy (0 až 9)

H
setiny sekundy (0 až 9)

Poznámka: Pokud jsou systémové hodiny synchronizovány s velmi přesným časovým standardem, je možné, že se ve vzácných případech vrátí 60 nebo 61 sekund v souboru *PutTime*. K tomu dochází, když jsou do globálního časového standardu vloženy přestupné sekundy.

Greenwichský střední čas (GMT) se používá pro pole *PutDate* a *PutTime* pod podmínkou, že systémové hodiny jsou přesně nastaveny na GMT.

Pokud byla zpráva vložena jako součást pracovní jednotky, čas je ten, kdy byla zpráva vložena, a ne čas, kdy byla transakce potvrzena.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, je-li v parametru **PutMsgOpts** zadána hodnota MQPMO_SET_ALL_CONTEXT. Správce front nekontroluje obsah pole s výjimkou toho, že jakékoli informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota MQPMO_SET_ALL_CONTEXT, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou MQ_PUT_TIME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 8 prázdných znaků v jiných programovacích jazycích.

ApplOriginData (MQCHAR4) pro MQMD

Toto je část *původního kontextu* zprávy. Další informace o kontextu zprávy viz "MQMD-Deskriptor zpráv" na stránce 424 a Kontext zprávy.

`ApplOriginData` je informace definovaná sadou aplikací, kterou lze použít k poskytnutí dalších informací o původu zprávy. Může být například nastaven aplikacemi, které jsou spuštěny s vhodným oprávněním uživatele, aby označily, zda jsou data identity důvěryhodná.

Správce front považuje tyto informace za znaková data, ale nedefinuje jejich formát. Když správce front generuje tyto informace, jsou zcela prázdné.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, je-li v parametru **PutMsgOpts** zadána hodnota MQPMO_SET_ALL_CONTEXT. Jakékoli informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota MQPMO_SET_ALL_CONTEXT, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou MQ_APPL_ORIGIN_DATA_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 4 prázdné znaky v jiných programovacích jazycích.

Je-li zpráva publikována, ačkoli je nastavena hodnota `ApplOriginData` , je v odběru, který obdrží, prázdná.

GroupId (MQBYTE24) pro MQMD

Jedná se o bajtový řetězec, který se používá k identifikaci konkrétní skupiny zpráv nebo logické zprávy, ke které fyzická zpráva patří. *GroupId* se také používá, pokud je pro zprávu povolena segmentace. Ve všech těchto případech má *GroupId* nenulovou hodnotu a v poli *MsgFlags* je nastaven jeden nebo více následujících příznaků:

- MQMF_MSG_IN_GROUP
- MQMF_LAST_MSG_IN_GROUP
- MQMF_SEGMENT
- MQMF_LAST_SEGMENT
- MQMF_SEGMENTATION_ALLOWED

Není-li nastaven žádný z těchto příznaků, má parametr *GroupId* speciální hodnotu null MQGI_NONE.

Aplikace nemusí toto pole nastavit ve volání MQPUT nebo MQGET, pokud:

- Ve volání MQPUT je určena hodnota MQPMO_LOGICAL_ORDER.
- Ve volání MQGET není uvedeno MQMO_MATCH_GROUP_ID.

Toto jsou doporučené způsoby použití těchto volání pro zprávy, které nejsou zprávami sestavy. Pokud však aplikace vyžaduje větší kontrolu nebo volání je MQPUT1, musí aplikace zajistit, aby byl parametr *GroupId* nastaven na odpovídající hodnotu.

Skupiny zpráv a segmenty lze správně zpracovat pouze v případě, že je identifikátor skupiny jedinečný. Z tohoto důvodu *aplikace nesmí generovat své vlastní identifikátory skupin* ; místo toho musí aplikace provést jednu z následujících možností:

- Je-li zadána volba MQPMO_LOGICAL_ORDER, správce front automaticky vygeneruje jedinečný identifikátor skupiny pro první zprávu ve skupině nebo segmentu logické zprávy a použije tento identifikátor skupiny pro zbývající zprávy ve skupině nebo segmentech logické zprávy, takže aplikace nemusí provádět žádné speciální akce. Toto je doporučený postup.
- Není-li zadána hodnota MQPMO_LOGICAL_ORDER, musí aplikace požádat správce front o vygenerování identifikátoru skupiny nastavením parametru *GroupId* na hodnotu MQGI_NONE pro první volání MQPUT nebo MQPUT1 pro zprávu ve skupině nebo segmentu logické zprávy. Identifikátor skupiny vrácený správcem front na výstupu z tohoto volání musí být poté použit pro zbývající zprávy ve skupině nebo segmentech logické zprávy. Pokud skupina zpráv obsahuje segmentované zprávy, musí být pro všechny segmenty a zprávy ve skupině použit stejný identifikátor skupiny.

  Není-li zadána hodnota MQPMO_LOGICAL_ORDER, lze zprávy ve skupinách a segmentech logických zpráv vkládat v libovolném pořadí (například v opačném pořadí), identifikátor skupiny však musí být přidělen voláním *first* MQPUT nebo MQPUT1 , které je vydáno pro kteroukoli z těchto zpráv.

Na vstupu volání MQPUT a MQPUT1 používá správce front hodnotu popsanou v části Fyzické pořadí ve frontě. Ve výstupu volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána spolu se zprávou, pokud je otevřený objekt jedinou frontou a nikoli distribučním seznamem, ale ponechá jej beze změny, pokud je otevřený objekt distribučním seznamem. V druhém případě, pokud aplikace potřebuje znát vygenerované identifikátory skupin, musí aplikace poskytnout záznamy MQPMR obsahující pole *GroupId* .

Při vstupu do volání MQGET používá správce front hodnotu popsanou v části Tabulka 495 na stránce 396. Ve výstupu volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Je definována následující speciální hodnota:

MQGI_NONE
Není uveden žádný identifikátor skupiny.

Hodnota je binární nula pro délku pole. Jedná se o hodnotu, která se používá pro zprávy, které nejsou ve skupinách, nejsou segmenty logických zpráv a pro které není segmentace povolena.

Pro programovací jazyk C je definována také konstanta MQGI_NONE_ARRAY; má stejnou hodnotu jako MQGI_NONE, ale je to pole znaků namísto řetězce.

Délka tohoto pole je dána hodnotou MQ_GROUP_ID_LENGTH. Počáteční hodnota tohoto pole je MQGI_NONE. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQMD_VERSION_2.

MsgSeqPočet (MQLONG) pro MQMD

Toto je pořadové číslo logické zprávy ve skupině.

Pořadová čísla začínají na 1 a zvyšují se o 1 pro každou novou logickou zprávu ve skupině, maximálně 999 999 999 999. Fyzická zpráva, která se nenachází ve skupině, má pořadové číslo 1.

Aplikace nemusí nastavit toto pole ve volání MQPUT nebo MQGET, pokud:

- Ve volání MQPUT je určena hodnota MQPMO_LOGICAL_ORDER.
- Ve volání MQGET není uvedeno MQMO_MATCH_MSG_SEQ_NUMBER.

Toto jsou doporučené způsoby použití těchto volání pro zprávy, které nejsou zprávami sestavy. Pokud však aplikace vyžaduje větší kontrolu nebo volání je MQPUT1, musí aplikace zajistit, aby byl parametr *MsgSeqNumber* nastaven na odpovídající hodnotu.

Na vstupu volání MQPUT a MQPUT1 používá správce front hodnotu popsanou v části Fyzické pořadí ve frontě. Ve výstupu volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána se zprávou.

Při vstupu do volání MQGET používá správce front hodnotu uvedenou v části Tabulka 495 na stránce 396. Ve výstupu volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Počáteční hodnota tohoto pole je jedna. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQMD_VERSION_2.

Posunutí (MQLONG) pro MQMD

Jedná se o posun v bajtech dat ve fyzické zprávě od začátku logické zprávy, jejíž část tvoří data. Tato data se nazývají *segment*. Posun je v rozsahu 0 až 999 999 999 999. Fyzická zpráva, která není segmentem logické zprávy, má offset nula.

Aplikace nemusí toto pole nastavit ve volání MQPUT nebo MQGET, pokud:

- Ve volání MQPUT je určena hodnota MQPMO_LOGICAL_ORDER.
- Ve volání MQGET není určen parametr MQMO_MATCH_OFFSET.

Toto jsou doporučené způsoby použití těchto volání pro zprávy, které nejsou zprávami sestavy. Pokud však aplikace nesplňuje tyto podmínky nebo volání je MQPUT1, musí aplikace zajistit, aby byl parametr *Offset* nastaven na odpovídající hodnotu.

Na vstupu volání MQPUT a MQPUT1 používá správce front hodnotu popsanou v části Fyzické pořadí ve frontě. Ve výstupu volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána se zprávou.

Pro zprávu sestavy vykazující segment logické zprávy se pole *OriginalLength* (za předpokladu, že není MQOL_UNDEFINED) používá k aktualizaci offsetu v informacích o segmentu uchovávaných správcem front.

Při vstupu do volání MQGET používá správce front hodnotu uvedenou v části Tabulka 495 na stránce 396. Ve výstupu volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Počáteční hodnota tohoto pole je nula. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQMD_VERSION_2.

MsgFlags (MQLONG) pro MQMD

MsgFlags jsou příznaky, které určují atributy zprávy nebo řídí její zpracování.

Položky MsgFlags jsou rozděleny do následujících kategorií:

- Příznaky segmentace
- Příznaky stavu

Příznaky segmentace: Pokud je zpráva pro frontu příliš velká, pokus o vložení zprávy do fronty obvykle selže. Segmentace je technika, při které správce front nebo aplikace rozdělí zprávu na menší části nazývané segmenty a umístí jednotlivé segmenty do fronty jako samostatnou fyzickou zprávu. Aplikace, která načte zprávu, může buď načíst segmenty jeden po druhém, nebo požádat správce front o opětovné sestavení segmentů do jedné zprávy, která je vrácena voláním MQGET. Toho je dosaženo určením volby MQGMO_COMPLETE_MSG ve volání MQGET a dodáním vyrovnávací paměti, která je dostatečně velká, aby pojmula celou zprávu. (Podrobnosti o volbě MQGMO_COMPLETE_MSG naleznete v části “MQGMO-Volby získání zprávy” na stránce 371 .) Zprávu lze segmentovat v odesílajícím správci front, v zprostředkujícím správci front nebo v cílovém správci front.

Chcete-li řídit segmentaci zprávy, můžete určit jednu z následujících možností:

MQMF_SEGMENTATION_INHIBITED
Tato volba brání tomu, aby správce front rozdělil zprávu na segmenty. Je-li uvedeno pro zprávu, která je již segmentem, tato volba zabrání rozdělení segmentu na menší segmenty.

Hodnota tohoto příznaku je binární nula. Toto nastavení je výchozí.

MQMF_SEGMENTATION_ALLOWED
Tato volba umožňuje správci front rozdělit zprávu na segmenty. Je-li uvedeno pro zprávu, která je již segmentem, tato volba umožňuje rozdělení segmentu na menší segmenty. MQMF_SEGMENTATION_ALLOWED lze nastavit bez nastavení MQMF_SEGMENT nebo MQMF_LAST_SEGMENT.

- V systému z/OSsprávce front nepodporuje segmentaci zpráv. Pokud je zpráva pro frontu příliš velká, volání MQPUT nebo MQPUT1 selže s kódem příčiny MQRC_MSG_TOO_BIG_FOR_Q. Volba MQMF_SEGMENTATION_ALLOWED však může být i nadále určena a umožňuje segmentaci zprávy ve vzdáleném správci front.

Když správce front segmentuje zprávu, zapne správce front příznak MQMF_SEGMENT v kopii deskriptoru MQMD, který je odeslán s každým segmentem, ale nezmění nastavení těchto příznaků v deskriptoru MQMD poskytnutém aplikací ve volání MQPUT nebo MQPUT1 . Pro poslední segment v logické zprávě správce front také zapne příznak MQMF_LAST_SEGMENT v deskriptoru MQMD, který je odeslán spolu se segmentem.

Poznámka: Dávejte pozor při vkládání zpráv s parametrem MQMF_SEGMENTATION_ALLOWED, ale bez parametru MQPMO_LOGICAL_ORDER. Pokud je zpráva:

- Nejedná se o segment a
- Není ve skupině a
- Nepředávají se,

Aplikace musí resetovat pole *GroupId* na hodnotu MQGI_NONE před *každým voláním* MQPUT nebo MQPUT1 , aby mohl správce front generovat jedinečný identifikátor skupiny pro každou zprávu. Pokud se to neprovede, nesouvisející zprávy mohou mít stejný identifikátor skupiny, což může vést k následnému chybnému zpracování. Další informace o tom, kdy resetovat pole *GroupId* , naleznete v popisech pole *GroupId* a volby MQPMO_LOGICAL_ORDER.

Správce front rozdělí zprávy do segmentů podle potřeby tak, aby se segmenty (plus všechna požadovaná data záhlaví) vešly do fronty. Existuje však dolní limit velikosti segmentu generovaného správcem front a pouze poslední segment vytvořený ze zprávy může být menší než tento limit (dolní limit pro velikost segmentu generovaného aplikací je jeden bajt). Segmenty generované správcem front mohou mít nestejnou délku. Správce front zpracuje zprávu následujícím způsobem:

- Uživatelem definované formáty jsou rozděleny na hranice, které jsou násobky 16 bajtů; správce front negeneruje segmenty, které jsou menší než 16 bajtů (jiné než poslední segment).
- Vestavěné formáty jiné než MQFMT_STRING jsou rozděleny v bodech odpovídajících povaze přítomných dat. Správce front však nikdy nerozdělí zprávu uprostřed struktury záhlaví IBM MQ . To znamená, že segment obsahující jedinou strukturu záhlaví produktu MQ nemůže správce front dále rozdělit, a v důsledku toho je minimální možná velikost segmentu pro tuto zprávu větší než 16 bajtů.

  Druhý nebo novější segment vygenerovaný správcem front začíná jedním z následujících:

  – Struktura záhlaví MQ
  – Spuštění dat zprávy aplikace
  – Část cesty přes data zprávy aplikace
- MQFMT_STRING je rozdělen bez ohledu na povahu přítomných dat (SBCS, DBCS nebo smíšený SBCS/DBCS). Je-li řetězec DBCS nebo smíšený SBCS/DBCS, může to vést k segmentům, které nelze převést z jedné znakové sady na jinou. Správce front nikdy nerozdělí zprávy MQFMT_STRING na segmenty, které jsou menší než 16 bajtů (jiné než poslední segment).
- Správce front nastaví pole *Format*, *CodedCharSetId*a *Encoding* v deskriptoru MQMD každého segmentu tak, aby správně popisovala data přítomná na *začátku* segmentu; název formátu je buď název vestavěného formátu, nebo název formátu definovaného uživatelem.
- Pole *Report* v deskriptoru MQMD segmentů s hodnotou *Offset* větší než nula bylo upraveno. Pro každý typ sestavy platí, že pokud je volba sestavy MQRO_ * _WITH_DATA, ale segment nemůže

obsahovat prvních 100 bajtů uživatelských dat (tj. data následující po případných strukturách záhlaví IBM MQ), volba sestavy se změní na MQRO_ *.

Správce front dodržuje výše uvedená pravidla, ale jinak nepředvídatelně rozděluje zprávy; nepředpokládejte, kde je zpráva rozdělena.

Pro *trvalé* zprávy může správce front provádět segmentaci pouze v rámci pracovní jednotky:

- Pokud volání MQPUT nebo MQPUT1 pracuje v rámci uživatelem definované jednotky práce, použije se tato jednotka práce. Pokud se volání během procesu segmentace nezdaří, správce front odebere všechny segmenty, které byly umístěny do fronty v důsledku selhání volání. Selhání však nebrání úspěšnému potvrzení transakce.
- Pokud volání pracuje mimo uživatelem definovanou jednotku práce a neexistuje žádná uživatelem definovaná jednotka práce, správce front vytvoří jednotku práce pouze po dobu trvání volání. Je-li volání úspěšné, správce front automaticky potvrdí transakci. Pokud volání selže, správce front odvolá transakci.
- Pokud volání pracuje mimo uživatelem definovanou jednotku práce, ale existuje uživatelem definovaná jednotka práce, nemůže správce front provést segmentaci. Pokud zpráva nevyžaduje segmentaci, volání může být stále úspěšné. Pokud však zpráva vyžaduje segmentaci, volání selže s kódem příčiny MQRC_UOW_NOT_AVAILABLE.

Pro *přechodné* zprávy správce front nevyžaduje, aby byla k dispozici jednotka práce pro provedení segmentace.

Při převodu dat ve zprávách, které mohou být segmentovány, věnujte zvláštní pozornost:

- Pokud přijímající aplikace převádí data ve volání MQGET a určuje volbu MQGMO_COMPLETE_MSG, je uživatelské proceduře převodu dat předána úplná zpráva pro uživatelskou proceduru, která má být převedena, a skutečnost, že zpráva byla segmentována, je pro uživatelskou proceduru zřejmá.
- Pokud přijímající aplikace načte jeden segment v daném okamžiku, je vyvolána uživatelská procedura převodu dat, aby převedla jeden segment v daném okamžiku. Ukončení proto musí převést data v segmentu nezávisle na datech v kterémkoli z ostatních segmentů.

  Pokud je povaha dat ve zprávě taková, že libovolná segmentace dat na 16bajtových hranicích může vést k segmentům, které nelze převést uživatelskou procedurou, nebo formát je MQFMT_STRING a znaková sada je DBCS nebo smíšená SBCS/DBCS, musí odesílající aplikace vytvořit a vložit segmenty s uvedením MQMF_SEGMENTATION_INHIBITED, aby potlačila další segmentaci. Tímto způsobem může odesílající aplikace zajistit, aby každý segment obsahoval dostatečné informace, které umožní uživatelské proceduře pro převod dat úspěšně převést segment.
- Pokud je pro agenta MCA (odeslání kanálu zpráv) zadán převod odesilatele, adaptér MCA převede pouze zprávy, které nejsou segmenty logických zpráv; agent MCA se nikdy nepokusí převést zprávy, které jsou segmenty.

Tento příznak je vstupním příznakem volání MQPUT a MQPUT1 a výstupním příznakem volání MQGET. Při druhém volání správce front také zopakuje hodnotu příznaku do pole *Segmentation* v MQGMO.

Počáteční hodnota tohoto příznaku je MQMF_SEGMENTATION_INHIBITED.

Příznaky stavu: Jedná se o příznaky, které označují, zda fyzická zpráva patří do skupiny zpráv, zda se jedná o segment logické zprávy, obojí, nebo obojí. Ve volání MQPUT nebo MQPUT1 lze zadat jednu nebo více následujících možností nebo je lze vrátit voláním MQGET:

MQMF_MSG_IN_GROUP
Zpráva je členem skupiny.

MQMF_LAST_MSG_IN_GROUP
Zpráva je poslední logická zpráva ve skupině.

Je-li tento příznak nastaven, správce front zapne MQMF_MSG_IN_GROUP v kopii MQMD, která je odeslána se zprávou, ale nezmění nastavení těchto příznaků v MQMD poskytnutém aplikací ve volání MQPUT nebo MQPUT1 .

Je platné, aby skupina sestával pouze z jedné logické zprávy. V tomto případě je nastavena hodnota MQMF_LAST_MSG_IN_GROUP, ale pole *MsgSeqNumber* má hodnotu jedna.

MQMF_SEGMENT
Zpráva je segmentem logické zprávy.

Je-li volba MQMF_SEGMENT zadána bez volby MQMF_LAST_SEGMENT, musí být délka dat zprávy aplikace v segmentu (*s výjimkou* délek libovolných struktur záhlaví IBM MQ , které mohou být přítomny) alespoň jedna. Je-li délka nulová, volání MQPUT nebo MQPUT1 selže s kódem příčiny MQRC_SEGMENT_LENGTH_ZERO.

V systému z/OSnení tato volba podporována, pokud je zpráva vkládána do fronty s typem indexu MQIT_GROUP_ID.

MQMF_LAST_SEGMENT
Zpráva je posledním segmentem logické zprávy.

Je-li tento příznak nastaven, správce front zapne funkci MQMF_SEGMENT v kopii MQMD, která je odeslána se zprávou, ale nezmění nastavení těchto příznaků v deskriptoru MQMD poskytnutém aplikací ve volání MQPUT nebo MQPUT1 .

Logická zpráva se může skládat pouze z jednoho segmentu. Pokud ano, je nastaven segment MQMF_LAST_SEGMENT, ale pole *Offset* má hodnotu nula.

Je-li zadána volba MQMF_LAST_SEGMENT, může být délka dat zprávy aplikace v segmentu (*s výjimkou* délek libovolných struktur záhlaví, které mohou být přítomny) nulová.

V systému z/OSnení tato volba podporována, pokud je zpráva vkládána do fronty s typem indexu MQIT_GROUP_ID.

Aplikace musí zajistit správné nastavení těchto příznaků při vkládání zpráv. Pokud je zadána volba MQPMO_LOGICAL_ORDER nebo byla zadána v předchozím volání MQPUT pro manipulátor fronty, musí být nastavení příznaků konzistentní s informacemi o skupině a segmentu uchovávanými správcem front pro manipulátor fronty. Následující podmínky platí pro *následná* volání MQPUT pro popisovač fronty při zadání MQPMO_LOGICAL_ORDER:

- Pokud neexistuje žádná aktuální skupina nebo logická zpráva, všechny tyto příznaky (a jejich kombinace) jsou platné.
- Jakmile je zadána hodnota MQMF_MSG_IN_GROUP, musí zůstat zapnutý, dokud není zadána hodnota MQMF_LAST_MSG_IN_GROUP. Volání selže s kódem příčiny MQRC_INCOMPLETE_GROUP, pokud není tato podmínka splněna.
- Jakmile je zadána volba MQMF_SEGMENT, musí zůstat zapnutý, dokud není zadána volba MQMF_LAST_SEGMENT. Volání selže s kódem příčiny MQRC_INCOMPLETE_MSG, pokud není tato podmínka splněna.
- Po zadání parametru MQMF_SEGMENT bez parametru MQMF_MSG_IN_GROUP musí skupina MQMF_MSG_IN_GROUP zůstat *vypnutá* , dokud nebude zadán parametr MQMF_LAST_SEGMENT. Volání selže s kódem příčiny MQRC_INCOMPLETE_MSG, pokud není tato podmínka splněna.

Fyzické pořadí ve frontě zobrazuje platné kombinace příznaků a hodnoty použité pro různá pole.

Jedná se o vstupní příznaky pro volání MQPUT a MQPUT1 a výstupní příznaky pro volání MQGET. Při druhém volání správce front také zopakuje hodnoty příznaků do polí *GroupStatus* a *SegmentStatus* v produktu MQGMO.

Nelze použít seskupené nebo segmentované zprávy s publikování/odběrem.

Výchozí příznaky: Pomocí následujícího příkazu lze určit, že zpráva má výchozí atributy:

MQMF_NONE
Žádné příznaky zprávy (výchozí atributy zprávy).

To brání segmentaci a označuje, že zpráva není ve skupině a není segmentem logické zprávy. MQMF_NONE je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby byl tento příznak použit s jinými, ale protože jeho hodnota je nula, nelze takové použití zjistit.

Pole *MsgFlags* je rozděleno na dílčí pole; podrobnosti viz "Volby sestavy a příznaky zpráv" na stránce 901.

Počáteční hodnota tohoto pole je MQMF_NONE. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQMD_VERSION_2.

OriginalLength (MQLONG) pro MQMD

Toto pole je relevantní pouze pro zprávy sestavy, které jsou segmenty. Uvádí délku segmentu zprávy, ke kterému se zpráva sestavy vztahuje; neuvádí délku logické zprávy, jejíž část tvoří segment, nebo délku dat ve zprávě sestavy.

Poznámka: Při generování zprávy sestavy pro zprávu, která je segmentem, zkopírují správce front a agent kanálu zpráv do MQMD pro zprávu sestavy pole *GroupId*, *MsgSeqNumber*, *Offset*a *MsgFlags*z původní zprávy. Výsledkem je, že zpráva sestavy je také segmentem. Aplikace, které generují zprávy sestavy, musí provést totéž a správně nastavit pole *OriginalLength* .

Je definována následující speciální hodnota:

MQOL_UNDEFINED
Původní délka zprávy není definována.

OriginalLength je vstupní pole pro volání MQPUT a MQPUT1 , ale hodnota, kterou aplikace poskytuje, je přijata pouze za určitých okolností:

- Pokud je vkládaná zpráva segmentem a je také zprávou sestavy, správce front přijme zadanou hodnotu. Hodnota musí být:
  - Větší než nula, pokud segment není posledním segmentem
  - Není menší než nula, pokud je segment posledním segmentem
  - Ne méně než délka dat přítomných ve zprávě

  Nejsou-li tyto podmínky splněny, volání selže s kódem příčiny MQRC_ORIGINAL_LENGTH_ERROR.
- Pokud je vkládaná zpráva segmentem, ale ne zprávou sestavy, správce front pole ignoruje a místo toho použije délku dat zprávy aplikace.
- Ve všech ostatních případech správce front ignoruje pole a místo toho použije hodnotu MQOL_UNDEFINED.

Toto je výstupní pole pro volání MQGET.

Počáteční hodnota tohoto pole je MQOL_UNDEFINED. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQMD_VERSION_2.

MQMDE-Rozšíření deskriptoru zpráv

Struktura MQMDE popisuje data, která se někdy vyskytují před daty zprávy aplikace. Struktura obsahuje pole MQMD, která existují v deskriptoru MQMD version-2 , ale nikoli v deskriptoru MQMD version-1 .

Dostupnost

Všechny systémy IBM MQ a IBM MQ MQI clients připojené k těmto systémům.

Název formátu

MQFMT_MD_EXTENSION

Znaková sada a kódování

Data v MQMDE musí být ve znakové sadě a kódování lokálního správce front; tato data jsou dána atributem správce front **CodedCharSetId** a MQENC_NATIVE pro programovací jazyk C.

Nastavte znakovou sadu a kódování MQMDE do polí *CodedCharSetId* a *Encoding* v:

- MQMD (pokud je struktura MQMDE na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQMDE (všechny ostatní případy).

Pokud prostředí MQMDE není ve znakové sadě a kódování správce front, bude prostředí MQMDE přijato, ale nebude uznáno, tj. prostředí MQMDE bude považováno za data zprávy.

Poznámka: V systému Windowspoužívají aplikace kompilované s mikrofokusem COBOL hodnotu MQENC_NATIVE, která se liší od kódování správce front. Ačkoli číselná pole ve struktuře MQMD ve voláních MQPUT, MQPUT1a MQGET musí být v kódování Micro Focus COBOL, číselná pole ve struktuře MQMDE musí být v kódování správce front. Ten je dán hodnotou MQENC_NATIVE pro programovací jazyk C a má hodnotu 546.

Pouľití

Aplikace, které používají MQMD version-2 , nebudou mít strukturu MQMDE. Avšak specializované aplikace a aplikace, které nadále používají MQMD version-1 , mohou v některých situacích narazit na MQMDE. Struktura MQMDE se může vyskytnout za následujících okolností:

- Určeno pro volání MQPUT a MQPUT1 .
- Vráceno voláním MQGET
- Ve zprávách v přenosových frontách

MQMDE určeno pro volání MQPUT a MQPUT1

Pokud aplikace ve voláních MQPUT a MQPUT1 poskytuje MQMD version-1 , může volitelně přidat k datům zprávy předponu MQMDE a nastavit pole *Format* v MQMD na MQFMT_MD_EXTENSION, aby označila přítomnost MQMDE. Pokud aplikace neposkytne prostředí MQMDE, správce front předpokládá výchozí hodnoty pro pole v prostředí MQMDE. Výchozí hodnoty, které správce front používá, jsou stejné jako počáteční hodnoty pro strukturu; viz Tabulka 503 na stránce 475.

Pokud aplikace poskytuje předpony version-2 MQMD *a* před data zprávy aplikace s MQMDE, struktury se zpracují, jak ukazuje následující tabulka.

Tabulka 502. Akce správce front při zadání MQMDE na MQPUT nebo MQPUT1 pro MQMDE

Verze MQMD	Hodnoty polí version-2	Hodnoty odpovídajících polí v MQMDE	Akce prováděná správcem front
1	-	Platný	MQMDE je uznána
2	Výchozí	Platný	MQMDE je uznána
2	Není výchozí	Platný	MQMDE se považuje za data zprávy
1 nebo 2	Libovolný	Neplatné	Volání selže s odpovídajícím kódem příčiny
1 nebo 2	Libovolný	MQMDE je v nesprávné znakové sadě nebo kódování nebo se jedná o nepodporovanou verzi	MQMDE se považuje za data zprávy
Poznámka: V systému z/OSplatí, že pokud aplikace uvádí version-1 MQMD s MQMDE, správce front ověří MQMDE pouze v případě, že fronta má hodnotu *IndexType* MQIT_GROUP_ID.			

Existuje jeden zvláštní případ. Pokud aplikace používá příkaz MQMD version-2 k vložení zprávy, která je segmentem (tj. je nastaven příznak MQMF_SEGMENT nebo MQMF_LAST_SEGMENT), a název formátu v záhlaví MQMD je MQFMT_DEAD_LETTER_HEADER, správce front vygeneruje strukturu MQMDE a vloží ji *mezi* strukturu MQDLH a data, která ji následují. V deskriptoru MQMD, který správce front uchovává se zprávou, jsou pole version-2 nastavena na výchozí hodnoty.

Některá pole, která existují v deskriptoru version-2 MQMD, ale nikoli version-1 MQMD, jsou vstupní/výstupní pole v deskriptoru MQPUT a MQPUT1. Správce front však nevrací žádné hodnoty v ekvivalentních polích ve výstupu MQMDE z volání MQPUT a MQPUT1 . Pokud aplikace tyto výstupní hodnoty vyžaduje, musí použít MQMD version-2 .

MQMDE vráceno voláním MQGET

Pokud aplikace ve volání MQGET poskytuje MQMD version-1 , správce front před zprávu vrácenou s MQMDE předřadí, ale pouze v případě, že jedno nebo více polí v MQMDE má jinou než výchozí hodnotu. Správce front nastaví pole *Format* v MQMD na hodnotu MQFMT_MD_EXTENSION, aby označil přítomnost MQMDE.

Pokud aplikace poskytuje MQMDE na začátku parametru **Buffer** , je MQMDE ignorováno. Při návratu z volání MQGET je nahrazen MQMDE pro zprávu (je-li potřeba) nebo přepsán daty zprávy aplikace (není-li MQMDE potřeba).

Pokud volání MQGET vrátí prostředí MQMDE, data v prostředí MQMDE jsou obvykle ve znakové sadě a kódování správce front. Prostředí MQMDE však může být v jiné znakové sadě a kódování, pokud:

- S MQMDE bylo zacházeno jako s daty volání MQPUT nebo MQPUT1 (okolnosti, které to mohou způsobit, viz Tabulka 502 na stránce 474).
- Zpráva byla přijata ze vzdáleného správce front připojeného prostřednictvím připojení TCP a přijímající agent kanálu zpráv (MCA) nebyl správně nastaven.

Poznámka: V systému Windowspoužívají aplikace kompilované s mikrofokusem COBOL hodnotu MQENC_NATIVE, která se liší od kódování správce front (viz výše).

MQMDE ve zprávách v přenosových frontách

Zprávy v přenosových frontách mají předponu ve struktuře MQXQH, která v sobě obsahuje version-1 MQMD. Může být také přítomno prostředí MQMDE umístěné mezi strukturou MQXQH a daty zprávy aplikace, ale obvykle je přítomno pouze v případě, že jedno nebo více polí v prostředí MQMDE má jinou než výchozí hodnotu.

Mezi strukturou MQXQH a daty zpráv aplikace se mohou vyskytovat i jiné struktury záhlaví produktu MQ . Pokud je například přítomno záhlaví nedoručených zpráv MQDLH a zpráva není segmentem, je pořadí následující:

- MQXQH (obsahující version-1 MQMD)
- MQMDE
- MQDLH
- data zprávy aplikace

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 503. Pole v MQMDE pro MQMDE

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQMDE_STRUC_ID	'MDE¬'
Verze (číslo verze struktury)	MQMDE_VERSION_2	2
StrucLength (délka struktury MQMDE)	MQMDE_LENGTH_2	72

Tabulka 503. Pole v MQMDE pro MQMDE (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
Kódování (číselné kódování dat, která následují za MQMDE)	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId (identifikátor znakové sady dat, která následují za MQMDE)	MQCCSI_UNDEFINED	0
Formát (název formátu dat, která následují za MQMDE)	MQFMT_NONE	Mezery
Příznaky (obecné příznaky)	MQMDEF_NONE	0
GroupId (identifikátor skupiny)	MQGI_NONE	Hodnoty null
MsgSeqNumber (pořadové číslo logické zprávy ve skupině)	Není	1
Posunutí (posunutí dat ve fyzické zprávě od začátku logické zprávy)	Není	0
MsgFlags (příznaky zprávy)	MQMF_NONE	0
OriginalLength (délka původní zprávy)	MQOL_UNDEFINED	-1

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQMDE_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQMDE MyMDE = {MQMDE_DEFAULT};
```

Deklarace jazyka

Deklarace jazyka C pro prostředí MQMDE

```
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 */
};
```

Deklarace jazyka COBOL pro prostředí MQMDE

```
**   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.
```

Deklarace PL/I pro MQMDE

```
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 */
```

Deklarace High Level Assembler pro prostředí MQMDE

```
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)
```

Deklarace jazyka Visual Basic pro prostředí MQMDE

```
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
```

StrucId (MQCHAR4) pro MQMDE

Jedná se o identifikátor struktury struktury rozšíření deskriptoru zpráv. Vždy se jedná o vstupní pole. Jeho hodnota je MQMDE_STRUC_ID.

Hodnota musí být:

MQMDE_STRUC_ID
Identifikátor pro strukturu rozšíření deskriptoru zpráv.

Pro programovací jazyk C je definována také konstanta MQMDE_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQMDE_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQMDE

Toto je číslo verze struktury; hodnota musí být:

MQMDE_VERSION_2
Version-2 struktura rozšíření deskriptoru zpráv.

Následující konstanta určuje číslo verze aktuální verze:

MQMDE_CURRENT_VERSION
Aktuální verze struktury rozšíření deskriptoru zpráv.

Počáteční hodnota tohoto pole je MQMDE_VERSION_2.

StrucLength (MQLONG) pro MQMDE

Toto je délka struktury MQMDE; je definována následující hodnota:

MQMDE_LENGTH_2
Délka struktury rozšíření deskriptoru zpráv version-2 .

Počáteční hodnota tohoto pole je MQMDE_LENGTH_2.

Kódování (MQLONG) pro MQMDE

Tato volba určuje číselné kódování dat, která jsou v souladu se strukturou MQMDE; nevztahuje se na číselná data v samotné struktuře MQMDE.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je pole platné. Další informace o kódování dat viz pole *Encoding* popsané v části “MQMD-Deskriptor zpráv” na stránce 424 .

Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG) pro MQMDE

Tato volba určuje identifikátor znakové sady dat, která následují za strukturou MQMDE; nevztahuje se na znaková data v samotné struktuře MQMDE.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je toto pole platné. Lze použít následující speciální hodnotu:

MQCCSI_INHERIT
Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota MQCCSI_INHERIT vrácena voláním MQGET.

MQCCSI_INHERIT nelze použít, pokud hodnota pole *PutApplType* v MQMD je MQAT_BROKER.

Tato hodnota je podporována v následujících prostředích:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

a pro klienty IBM MQ připojené k těmto systémům.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8) pro MQMDE

Tato volba určuje název formátu dat, která se řídí strukturou MQMDE.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je toto pole platné. Další informace o názvech formátů viz pole *Format* popsané v části"MQMD-Deskriptor zpráv" na stránce 424.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQMDE

Lze zadat následující příznak:

MQMDEF_NONE
Žádné příznaky.

Počáteční hodnota tohoto pole je MQMDEF_NONE.

GroupId (MQBYTE24) pro prostředí MQMDE

Viz pole *GroupId* popsané v části"MQMD-Deskriptor zpráv" na stránce 424. Počáteční hodnota tohoto pole je MQGI_NONE.

MsgSeqPočet (MQLONG) pro MQMDE

Viz pole *MsgSeqNumber* popsané v části"MQMD-Deskriptor zpráv" na stránce 424. Počáteční hodnota tohoto pole je 1.

Posunutí (MQLONG) pro MQMDE

Viz pole *Offset* popsané v části"MQMD-Deskriptor zpráv" na stránce 424. Počáteční hodnota tohoto pole je 0.

MsgFlags (MQLONG) pro MQMDE

Viz pole *MsgFlags* popsané v části "MQMD-Deskriptor zpráv" na stránce 424. Počáteční hodnota tohoto pole je MQMF_NONE.

OriginalLength (MQLONG) pro MQMDE

Viz pole *OriginalLength* popsané v části"MQMD-Deskriptor zpráv" na stránce 424. Počáteční hodnota tohoto pole je MQOL_UNDEFINED.

MQMHBO-Možnosti zpracování zprávy do vyrovnávací paměti

Struktura MQMHBO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou vyrovnávací paměti vytvářeny z manipulátorů zpráv. Struktura je vstupní parametr pro volání MQMHBUF.

Znaková sada a kódování

Data v MQMHBO musí být ve znakové sadě aplikace a kódování aplikace (MQENC_NATIVE).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 504. Pole v MQMHBO

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQMHBO_STRUC_ID	'MHBO'
Verze (číslo verze struktury)	MQMHBO_VERSION_1	1
Volby (volby řídící akci MQMHBUF)	MQMHBO_PROPERTIES_IN_MQRFH2	

Notes:

1. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyku C se jedná o proměnnou makra.MQMHBO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQMHBO MyMHBO = {MQMHBO_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQMHBO

```
typedef struct tagMQMHBO MQMHBO;
struct tagMQMHBO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQMHBUF */
};
```

Deklarace jazyka COBOL pro MQMHBO

```
**  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.
```

Prohlášení PL/I pro MQMHBO

```
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 */
```

Prohlášení High Level Assembler pro MQMHBO

```
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)
```

StrucId (MQCHAR4) pro MQMHBO

Jedná se o identifikátor struktury popisovače zprávy do struktury voleb vyrovnávací paměti. Vždy se jedná o vstupní pole. Jeho hodnota je MQMHBO_STRUC_ID.

Hodnota musí být:

MQMHBO_STRUC_ID
Identifikátor pro strukturu voleb popisovače zprávy do vyrovnávací paměti.

Pro programovací jazyk C je definována také konstanta MQMHBO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQMHBO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQMHBO

Struktura volby popisovače zprávy do vyrovnávací paměti-pole Verze

Toto je číslo verze struktury. Hodnota musí být:

MQMHBO_VERSION_1
Číslo verze pro strukturu voleb popisovače zprávy do vyrovnávací paměti.

Následující konstanta určuje číslo verze aktuální verze:

MQMHBO_CURRENT_VERSION
Aktuální verze popisovače zprávy do struktury voleb vyrovnávací paměti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQMHBO_VERSION_1.

Volby (MQLONG) pro MQMHBO

Struktura voleb popisovače zprávy do vyrovnávací paměti-pole Volby

Tyto volby řídí akci MQMHBUF.

Musíte uvést následující volbu:

MQMHBO_PROPERTIES_IN_MQRFH2
Při převodu vlastností z manipulátoru zprávy do vyrovnávací paměti je převeďte do formátu MQRFH2 .

Volitelně můžete také zadat následující volbu. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

MQMHBO_DELETE_PROPERTIES
Vlastnosti přidané do vyrovnávací paměti jsou odstraněny z manipulátoru zprávy. Pokud volání selže, nebudou odstraněny žádné vlastnosti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQMHBO_PROPERTIES_IN_MQRFH2.

MQOD-Popisovač objektu

Struktura MQOD se používá k určení objektu podle názvu. Struktura je vstupní/výstupní parametr volání MQOPEN a MQPUT1 .
Následující typy objektů jsou platné:

- Fronta nebo distribuční seznam
- Seznam názvů
- Definice procesu
- Správce front
- Téma

Dostupnost

Všechny systémy IBM MQ plus IBM MQ MQI clients připojené k těmto systémům.

Verze

Aktuální verze produktu MQOD je MQOD_VERSION_4. Aplikace, které chcete portovat mezi několika prostředími, musí zajistit, aby požadovaná verze produktu MQOD byla podporována ve všech příslušných prostředích. Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech, které následují.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi produktu MQOD podporovanou prostředím, ale s počáteční hodnotou pole *Version* nastavenou na hodnotu MQOD_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole *Version* na číslo verze požadované verze.

Chcete-li otevřít distribuční seznam, *Version* musí být MQOD_VERSION_2 nebo vyšší.

Znaková sada a kódování

Data v produktu MQOD musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódováním lokálního správce front daným proměnnou MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	ID_STRUC_MQOD_STRUC_ID	'OD¬¬'
Verze (číslo verze struktury)	MQOD_VERSION_1	1
ObjectType (typ objektu)	MQOT_Q	1
ObjectName (název objektu)	Není	Prázdný řetězec nebo mezery
ObjectQMgrNázev (název správce front objektů)	Není	Prázdný řetězec nebo mezery

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
DynamicQName (název dynamické fronty)	Není	'CSQ.*' na z/OS ; 'AMQ.*' jinak
AlternateUserID (alternativní identifikátor uživatele)	Není	Prázdný řetězec nebo mezery
Poznámka: Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než hodnota MQOD_VERSION_2.		
RecsPresent (počet přítomných záznamů objektů)	Není	0
KnownDest (počet úspěšně otevřených lokálních front)	Není	0
UnknownDestPočet (počet úspěšně otevřených vzdálených front)	Není	0
InvalidDestPočet (počet front, které se nepodařilo otevřít)	Není	0
ObjectRecOffset (posun prvního záznamu objektu od začátku MQOD)	Není	0
ResponseRecOffset (posunutí prvního záznamu odpovědi od začátku MQOD)	Není	0
ObjectRecPtr (adresa prvního záznamu objektu)	Není	Ukazatel Null nebo bajty s hodnotou Null
ResponseRecPtr (adresa prvního záznamu odpovědi)	Není	Ukazatel Null nebo bajty s hodnotou Null
Poznámka: Zbývající pole jsou ignorována, pokud je *Version* menší než MQOD_VERSION_3.		
AlternateSecurityID (alternativní identifikátor zabezpečení)	MQSID_NONE	Hodnoty null
ResolvedQName (vyřešený název fronty)	Není	Prázdný řetězec nebo mezery
ResolvedQMgrNázev (vyřešený název správce front)	Není	Prázdný řetězec nebo mezery
Poznámka: Zbývající pole jsou ignorována, pokud je *Version* menší než MQOD_VERSION_4.		
ObjectString (dlouhý název objektu)	VÝCHOZÍ	Podle definice pro MQCHARV
SelectionString (řetězec výběru)	VÝCHOZÍ	Podle definice pro MQCHARV
ResObjectString (interpretovaný dlouhý název objektu)	VÝCHOZÍ	Podle definice pro MQCHARV
ResolvedType (vyřešený typ objektu)	MQOT_NONE	0

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQOD_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQOD MyOD = {MQOD_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQOD

```
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 */
};
```

Deklarace jazyka COBOL pro MQOD

```
** 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.
```

Deklarace PL/I pro MQOD

```
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 */
```

Deklarace High Level Assembler pro MQOD

```
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)
```

Vizuální základní deklarace pro MQOD

```
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
```

StrucId (MQCHAR4) pro MQOD

Jedná se o identifikátor struktury struktury deskriptoru objektu. Vždy se jedná o vstupní pole. Jeho hodnota je MQOD_STRUC_ID.

Hodnota musí být:

ID_STRUC_MQOD_STRUC_ID
Identifikátor struktury deskriptoru objektu.

Pro programovací jazyk C je definována také konstanta MQOD_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQOD_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQOD

Toto je číslo verze struktury; hodnota musí být jedna z následujících:

MQOD_VERSION_1
Version-1 struktura deskriptoru objektu.

MQOD_VERSION_2
Version-2 struktura deskriptoru objektu.

MQOD_VERSION_3
Struktura deskriptoru objektu Version-3 .

MQOD_VERSION_4
Struktura deskriptoru objektu Version-4 .

Všechny verze jsou podporovány ve všech prostředích IBM MQ V7.0 .

Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MQOD_CURRENT_VERSION
Aktuální verze struktury deskriptoru objektu.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQOD_VERSION_1.

ObjectType (MQLONG) pro MQOD

Typ objektu, který je pojmenován v deskriptoru objektu. Možné hodnoty jsou:

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta. Název objektu se nachází v poli *ObjectName* .

MQOT_Q
Fronta. Název objektu se nachází v poli *ObjectName* .

MQOT_NAMELIST
Seznam názvů. Název objektu se nachází v poli *ObjectName* .

MQOT_PROCESS
Definice procesu. Název objektu se nachází v poli *ObjectName* .

MQOT_Q_MGR
Správce front. Název objektu se nachází v poli *ObjectName* .

MQOT_TOPIC
. Úplný název tématu lze sestavit ze dvou různých polí: *ObjectName* a *ObjectString*.

Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQOT_Q.

ObjectName (MQCHAR48) pro MQOD

Jedná se o lokální název objektu, jak je definován ve správci front identifikovaném pomocí *ObjectQMgrName*. Název může obsahovat následující znaky:

- Velká písmena abecedy (A až Z)
- Malá písmena abecedy (a až z)
- Číselné číslice (0 až 9)
- Tečka (.), dopředné lomítko (/), podtržítko (_), procento (%)

Název nesmí obsahovat úvodní ani vložené mezery, ale může obsahovat koncové mezery. Použijte znak null, abyste označili konec důležitých dat v názvu; hodnota null a všechny následující znaky jsou považovány za mezery. V označených prostředích platí následující omezení:

- V systémech, které používají EBCDIC Katakana, nelze použít malá písmena.
- V systému z/OS:
  - Vyvarujte se názvů, které začínají nebo končí podtržítkem; nemohou být zpracovány operacemi a ovládacími panely.
  - Znak procenta má speciální význam pro RACF. Pokud se RACF používá jako externí správce zabezpečení, názvy nesmí obsahovat procenta. Pokud ano, nejsou tyto názvy zahrnuty do žádných kontrol zabezpečení, když se používají generické profily RACF .
- V systému IBM imusí být názvy obsahující malá písmena, dopředné lomítko nebo procenta uzavřeny v uvozovkách, jsou-li zadány v příkazech. Tyto uvozovky nesmí být uvedeny pro názvy, které se vyskytují jako pole ve strukturách nebo jako parametry ve voláních.

Úplný název tématu lze sestavit ze dvou různých polí: *ObjectName* a *ObjectString*. Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat.

Pro uvedené typy objektů platí následující body:

- Pokud je *ObjectName* název modelové fronty, správce front vytvoří dynamickou frontu s atributy modelové fronty a vrátí do pole *ObjectName* název vytvořené fronty. Modelová fronta může být určena pouze pro volání MQOPEN; modelová fronta není platná pro volání MQPUT1 .
- Pokud je *ObjectName* název alias fronty s TARGTYPE (TOPIC), kontrola zabezpečení se nejprve provede v pojmenované alias frontě; to je normální, když se používají alias fronty. Po úspěšném dokončení kontroly zabezpečení bude volání MQOPEN pokračovat a bude se chovat jako volání MQOPEN pro MQOT_TOPIC; to zahrnuje provedení kontroly zabezpečení pro objekt administrativního tématu.
- Pokud funkce *ObjectName* a *ObjectQMgrName* identifikují sdílenou frontu vlastněnou skupinou sdílení front, do které patří lokální správce front, nesmí v lokálním správci front existovat také definice fronty se stejným názvem. Pokud existuje taková definice (lokální fronta, alias fronta, vzdálená fronta nebo modelová fronta), volání selže s kódem příčiny MQRC_OBJECT_NOT_UNIQUE.
- Pokud je otevíraný objekt rozdělovník (to znamená, že *RecsPresent* je přítomen a větší než nula), *ObjectName* musí být prázdný nebo prázdný řetězec. Není-li tato podmínka splněna, volání selže s kódem příčiny MQRC_OBJECT_NAME_ERROR.
- Je-li *ObjectType* MQOT_Q_MGR, použijí se speciální pravidla; v tomto případě musí být název zcela prázdný až do prvního znaku null nebo konce pole.

Jedná se o vstupní/výstupní pole pro volání MQOPEN, když *ObjectName* je název modelové fronty a pole pouze pro vstup ve všech ostatních případech. Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ObjectQMgrNázev (MQCHAR48) pro MQOD

Jedná se o název správce front, v němž je definován objekt *ObjectName* . Znaky platné v názvu jsou stejné jako znaky pro *ObjectName* (viz “ObjectName (MQCHAR48) pro MQOD” na stránce 488). Název, který je zcela prázdný až do prvního znaku null nebo do konce pole, označuje správce front, ke kterému je aplikace připojena (lokální správce front).

Pro uvedené typy objektů platí následující body:

- Má-li parametr *ObjectType* hodnotu MQOT_TOPIC, MQOT_NAMELIST, MQOT_PROCESS nebo MQOT_Q_MGR, *ObjectQMgrName* musí být hodnota prázdná nebo název lokálního správce front.
- Pokud je *ObjectName* název modelové fronty, vytvoří správce front dynamickou frontu s atributy modelové fronty a v poli *ObjectQMgrName* vrátí název správce front, ve kterém je fronta vytvořena; jedná se o název lokálního správce front. Modelová fronta může být určena pouze pro volání MQOPEN; modelová fronta není platná pro volání MQPUT1 .
- Pokud je *ObjectName* název fronty klastru a *ObjectQMgrName* je prázdný, je místo určení zpráv odeslaných pomocí manipulátoru fronty vráceného voláním MQOPEN vybráno správcem front (nebo uživatelskou procedurou pracovní zátěže klastru, pokud je nainstalována) takto:
  - Je-li zadána volba MQOO_BIND_ON_OPEN, správce front vybere při zpracování volání MQOPEN konkrétní instanci fronty klastru a všechny zprávy vkládané s použitím tohoto manipulátoru fronty budou odeslány do této instance.
  - Je-li zadána volba MQOOO_BIND_NOT_FIXED, může správce front zvolit jinou instanci cílové fronty (umístěnou v jiném správci front v klastru) pro každé následné volání MQPUT, které používá tento manipulátor fronty.

  Pokud aplikace potřebuje odeslat zprávu *specifické* instanci fronty klastru (tj. instanci fronty, která se nachází v konkrétním správci front v klastru), musí v poli *ObjectQMgrName* zadat název tohoto správce front. To vynutí, aby lokální správce front odeslal zprávu určenému cílovému správci front.
- Pokud je *ObjectName* název sdílené fronty, která je vlastněna skupinou sdílení front, do které patří lokální správce front, *ObjectQMgrName* může být názvem skupiny sdílení front, názvem lokálního správce front nebo může být prázdná; zpráva se umístí do stejné fronty, podle toho, která z těchto hodnot je zadána.

  Skupiny sdílení front jsou podporovány pouze v systému z/OS.

- Pokud je *ObjectName* název sdílené fronty vlastněné vzdálenou skupinou sdílení front (tj. skupinou sdílení front, do které lokální správce front nepatří), musí být *ObjectQMgrName* název skupiny sdílení front. Můžete použít název správce front, který patří do této skupiny, ale to může zpozdit zprávu, pokud tento konkrétní správce front není k dispozici, když zpráva dorazí do skupiny sdílení front.
- Pokud je otevíraný objekt rozdělovník (to znamená, že *RecsPresent* je větší než nula), *ObjectQMgrName* musí být prázdný nebo prázdný řetězec. Není-li tato podmínka splněna, volání selže s kódem příčiny MQRC_OBJECT_Q_MGR_NAME_ERROR.

Jedná se o vstupní/výstupní pole pro volání MQOPEN, když *ObjectName* je název modelové fronty a pole pouze pro vstup ve všech ostatních případech. Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

DynamicQName (MQCHAR48) pro MQOD

Jedná se o název dynamické fronty, která má být vytvořena voláním MQOPEN. Toto má význam pouze v případě, že parametr *ObjectName* určuje název modelové fronty; ve všech ostatních případech je parametr *DynamicQName* ignorován.

Znaky, které jsou platné v názvu, jsou stejné jako znaky pro *ObjectName*, kromě toho, že hvězdička je také platná. Název, který je prázdný (nebo ve kterém se před prvním znakem null vyskytují pouze mezery), není platný, pokud *ObjectName* je název modelové fronty.

Je-li posledním neprázdným znakem v názvu hvězdička (*), správce front nahradí hvězdičku řetězcem znaků, který zaručuje, že název vygenerovaný pro frontu je v lokálním správci front jedinečný. Pro povolení dostatečného počtu znaků je hvězdička platná pouze na pozicích 1 až 33. Za hvězdičkou nesmí být žádné jiné znaky než mezery nebo znak null.

Je platné, aby se hvězdička vyskytovala na první znakové pozici. V takovém případě se název skládá pouze ze znaků generovaných správcem front.

V systému z/OSnepoužívejte název s hvězdičkou na pozici prvního znaku, protože ve frontě s úplným názvem, který je generován automaticky, nemohou být provedeny žádné kontroly zabezpečení.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je určena prostředím:

- V systému z/OSje hodnota `'CSQ.*'`.
- Na ostatních platformách je hodnota `'AMQ.*'`.

Hodnota je řetězec ukončený hodnotou null v jazyce C a řetězec vyplněný mezerou v jiných programovacích jazycích.

AlternateUserID (MQCHAR12) pro MQOD

Zadáte-li pro volání MQOPEN hodnotu MQOOO_ALTERNATE_USER_AUTHORITY nebo pro volání MQPUT1 hodnotu MQPMO_ALTERNATE_USER_AUTHORITY, bude toto pole obsahovat alternativní identifikátor uživatele, který slouží ke kontrole autorizace pro otevření namísto identifikátoru uživatele, pod kterým je aplikace aktuálně spuštěna. Některé kontroly jsou však stále prováděny s aktuálním identifikátorem uživatele (například kontextové kontroly).

Je-li zadána volba MQOOO_ALTERNATE_USER_AUTHORITY nebo MQPMO_ALTERNATE_USER_AUTHORITY a toto pole je prázdné až po první znak null nebo konec pole, může být otevření úspěšné pouze v případě, že pro otevření tohoto objektu s určenými volbami není potřebná žádná autorizace uživatele.

Není-li zadána volba MQOOO_ALTERNATE_USER_AUTHORITY ani volba MQPMO_ALTERNATE_USER_AUTHORITY, bude toto pole ignorováno.

V označených prostředích existují následující rozdíly:

- V systému z/OSse ke kontrole autorizace pro otevření použije pouze prvních 8 znaků *AlternateUserId* . Avšak aktuální identifikátor uživatele musí být autorizován k uvedení tohoto konkrétního alternativního identifikátoru uživatele; pro tuto kontrolu se použije všech 12 znaků

alternativního identifikátoru uživatele. Identifikátor uživatele musí obsahovat pouze znaky povolené externím správcem zabezpečení.

Je-li pro frontu zadána hodnota *AlternateUserId* , může být tato hodnota následně použita správcem front při vložení zpráv. Pokud volby MQPMO_ * _CONTEXT určené ve volání MQPUT nebo MQPUT1 způsobí, že správce front vygeneruje informace o kontextu identity, umístí hodnotu *AlternateUserId* do pole *UserIdentifier* v deskriptoru MQMD zprávy namísto aktuálního identifikátoru uživatele.

- V jiných prostředích se produkt *AlternateUserId* používá pouze pro kontroly řízení přístupu na objektu, který se otevírá. Pokud je objektem fronta, produkt *AlternateUserId* neovlivní obsah pole *UserIdentifier* v deskriptoru MQMD zpráv odeslaných pomocí tohoto popisovače fronty.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 12 prázdných znaků v jiných programovacích jazycích.

RecsPresent (MQLONG) pro MQOD

Jedná se o počet záznamů objektů MQOR, které byly poskytnuty aplikací. Je-li toto číslo větší než nula, znamená to, že se otevírá distribuční seznam, přičemž *RecsPresent* je počet cílových front v seznamu. Distribuční seznam může obsahovat pouze jeden cíl.

Hodnota *RecsPresent* nesmí být menší než nula, a pokud je větší než nula *ObjectType* , musí být MQOT_Q; volání selže s kódem příčiny MQRC_RECS_PRESENT_ERROR, pokud nejsou tyto podmínky splněny.

V systému z/OSmusí být toto pole nula.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_2.

Počet KnownDest(MQLONG) pro MQOD

Jedná se o počet front v rozdělovníku, které se interpretují na lokální fronty a které byly úspěšně otevřeny. Tento počet nezahrnuje fronty, které se interpretují na vzdálené fronty (i když se lokální přenosová fronta používá na počátku k uložení zprávy). Je-li toto pole přítomno, je také nastaveno při otevírání jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_1.

UnknownDestPočet (MQLONG) pro MQOD

Jedná se o počet front v rozdělovníku, které se interpretují na vzdálené fronty a které byly úspěšně otevřeny. Je-li toto pole přítomno, je také nastaveno při otevírání jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_1.

Počet InvalidDest(MQLONG) pro MQOD

Jedná se o počet front v rozdělovníku, které se nepodařilo úspěšně otevřít. Je-li toto pole přítomno, je také nastaveno při otevírání jedné fronty, která není v rozdělovníku.

Poznámka: Je-li toto pole přítomno, je nastaveno pouze v případě, že parametr **CompCode** ve volání MQOPEN nebo MQPUT1 je MQCC_OK nebo MQCC_WARNING; není nastaveno, pokud má parametr **CompCode** hodnotu MQCC_FAILED.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_1.

ObjectRecOffset (MQLONG) pro MQOD

Toto je posun v bajtech prvního záznamu objektu MQOR od začátku struktury MQOD. Posun může být kladný nebo záporný. *ObjectRecOffset* se používá pouze při otevírání rozdělovníku. Je-li hodnota *RecsPresent* nula, pole se ignoruje.

Při otevírání distribučního seznamu musí být zadáno pole jednoho nebo více záznamů objektů MQOR, aby bylo možné určit názvy cílových front v distribučním seznamu. To lze provést jedním ze dvou způsobů:

- Pomocí pole posunutí *ObjectRecOffset*.

  V tomto případě musí aplikace deklarovat svou vlastní strukturu obsahující MQOD následovanou polem záznamů MQOR (s tolika prvky pole, kolik je potřeba) a nastavit *ObjectRecOffset* na offset prvního prvku v poli od začátku MQOD. Ujistěte se, že toto posunutí je správné a má hodnotu, kterou lze umístit v rámci MQLONG (nejrestriktivnějším programovacím jazykem je COBOL, pro který je platný rozsah -999 999 999 999 až +999 999 999 999).

  Použijte *ObjectRecOffset* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele způsobem, který není přenositelný do různých prostředí (například programovací jazyk COBOL).

- Pomocí pole ukazatele *ObjectRecPtr*.

  V tomto případě může aplikace deklarovat pole struktur MQOR odděleně od struktury MQOD a nastavit *ObjectRecPtr* na adresu pole.

  ObjectRecPtr použijte pro programovací jazyky, které podporují datový typ ukazatele způsobem, který je přenositelný do různých prostředí (například programovací jazyk C).

Bez ohledu na techniku, kterou zvolíte, použijte jeden z *ObjectRecOffset* a *ObjectRecPtr* ; volání selže s kódem příčiny MQRC_OBJECT_RECORDS_ERROR, jsou-li obě nulové, nebo jsou-li nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_2.

ResponseRecOffset (MQLONG) pro MQOD

Toto je posun v bajtech prvního záznamu odpovědi MQRR od začátku struktury MQOD. Posun může být kladný nebo záporný. *ResponseRecOffset* se používá pouze při otevírání rozdělovníku. Je-li hodnota *RecsPresent* nula, pole se ignoruje.

Při otevírání distribučního seznamu můžete zadat pole jednoho nebo více záznamů odpovědí MQRR, abyste identifikovali fronty, které se nepodařilo otevřít (pole*CompCode* v MQRR), a příčinu každého selhání (pole*Reason* v MQRR). Data jsou vrácena v poli záznamů odpovědí ve stejném pořadí, v jakém se názvy front vyskytují v poli záznamů objektů. Správce front nastaví záznamy odpovědí pouze v případě, že je výsledek volání smíšený (tj. některé fronty byly úspěšně otevřeny, zatímco jiné selhaly, nebo všechny selhaly, ale z různých důvodů); kód příčiny MQRC_MULTIPLE_REASON z volání označuje tento případ. Pokud stejný kód příčiny platí pro všechny fronty, je tato příčina vrácena v parametru **Reason** volání MQOPEN nebo MQPUT1 a záznamy odezvy nejsou nastaveny. Záznamy odpovědí jsou volitelné, ale pokud jsou zadány, musí z nich být *RecsPresent* .

Záznamy odezvy mohou být poskytnuty stejným způsobem jako záznamy objektů, buď uvedením offsetu v *ResponseRecOffset*, nebo uvedením adresy v *ResponseRecPtr* ; podrobnosti o tom, jak to provést, viz “ObjectRecOffset (MQLONG) pro MQOD” na stránce 491. Nelze však použít více než jeden z parametrů *ResponseRecOffset* a *ResponseRecPtr* ; volání selže s kódem příčiny MQRC_RESPONSE_RECORDS_ERROR, pokud jsou obě nenulové.

Pro volání MQPUT1 se tyto záznamy odpovědí používají k vrácení informací o chybách, které se vyskytnou při odeslání zprávy do front v distribučním seznamu, a také o chybách, které se vyskytnou při otevření front. Kód dokončení a kód příčiny z operace vložení pro frontu nahradí kód příčiny z operace otevření pro tuto frontu pouze v případě, že kód dokončení z fronty byl MQCC_OK nebo MQCC_WARNING.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_2.

ObjectRecPtr (MQPTR) pro MQOD

Jedná se o adresu prvního záznamu objektu MQOR. *ObjectRecPtr* se používá pouze při otevírání rozdělovníku. Je-li hodnota *RecsPresent* nula, pole se ignoruje.

K určení záznamů objektů můžete použít buď *ObjectRecPtr* , nebo *ObjectRecOffset* , ale ne obojí; popis pole *ObjectRecOffset* viz "ObjectRecOffset (MQLONG) pro MQOD" na stránce 491. Pokud nepoužíváte *ObjectRecPtr*, nastavte jej na ukazatel Null nebo na nulový počet bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou typu all-null. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_2.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky, přičemž počáteční hodnota je bajtový řetězec s hodnotou typu all-null.

ResponseRecPtr (MQPTR) pro MQOD

Jedná se o adresu prvního záznamu odpovědi MQRR. *ResponseRecPtr* se používá pouze při otevírání rozdělovníku. Je-li hodnota *RecsPresent* nula, pole se ignoruje.

Chcete-li uvést záznamy odezvy, ale ne obojí, použijte buď *ResponseRecPtr* , nebo *ResponseRecOffset* ; podrobnosti viz "ResponseRecOffset (MQLONG) pro MQOD" na stránce 492. Pokud nepoužíváte *ResponseRecPtr*, nastavte jej na ukazatel Null nebo na nulový počet bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou typu all-null. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_2.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky, přičemž počáteční hodnota je bajtový řetězec s hodnotou typu all-null.

AlternateSecurityId (MQBYTE40) pro MQOD

Jedná se o identifikátor zabezpečení, který je předán s produktem *AlternateUserId* autorizační službě, aby bylo možné provést odpovídající kontroly autorizace. Parametr *AlternateSecurityId* se používá pouze v případě, že:

- Ve volání MQOPEN je určena volba MQOO_ALTERNATE_USER_AUTHORITY, nebo
- MQPMO_ALTERNATE_USER_AUTHORITY je zadáno ve volání MQPUT1 ,

a pole *AlternateUserId* není zcela prázdné až do prvního znaku null nebo konce pole.

V systému Windowslze *AlternateSecurityId* použít k dodání identifikátoru zabezpečení Windows (SID), který jedinečně identifikuje *AlternateUserId*. Identifikátor SID pro uživatele lze získat ze systému Windows pomocí volání rozhraní API produktu `LookupAccountName()` Windows .

V systému z/OSje toto pole ignorováno.

Pole *AlternateSecurityId* má následující strukturu:

- První bajt je binární celé číslo obsahující délku důležitých dat, která následují; hodnota vylučuje samotný bajt délky. Pokud není přítomen žádný identifikátor zabezpečení, je délka nula.
- Druhý bajt označuje typ identifikátoru zabezpečení, který je přítomen; jsou možné následující hodnoty:

  MQSIDT_NT_SECURITY_ID
  Windows identifikátor zabezpečení.

  MQSIDT_NONE
  Žádný identifikátor zabezpečení.
- Třetí a následující bajty až do délky definované prvním bajtem obsahují samotný identifikátor zabezpečení.
- Zbývající bajty v poli jsou nastaveny na binární nulu.

Můžete použít následující speciální hodnotu:

MQSID_NONE
Nebyl uveden žádný identifikátor zabezpečení.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQSID_NONE_ARRAY; má stejnou hodnotu jako MQSID_NONE, ale je to pole znaků místo řetězce.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_SECURITY_ID_LENGTH. Počáteční hodnota tohoto pole je MQSID_NONE. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_3.

ResolvedQName (MQCHAR48) pro MQOD

Jedná se o název cílové fronty po vyřešení názvu lokálním správcem front. Vrácený název je název fronty, která existuje ve správci front identifikovaném pomocí *ResolvedQMgrName*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou otevřenou pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Pokud je otevřený objekt některý z následujících, *ResolvedQName* je nastaven na mezery:

- Nejedná se o frontu
- Fronta, ale neotevřená pro procházení, vstup nebo výstup
- Distribuční seznam
- Alias fronty, která odkazuje na objekt tématu (místo toho viz ResObjectString).
- Alias fronty, která se interpretuje jako objekt tématu.

Toto je výstupní pole. Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_3.

ResolvedQMgrNázev (MQCHAR48) pro MQOD

Jedná se o název cílového správce front poté, co lokální správce front přeloží název. Vrácený název je název správce front, který vlastní frontu označenou jako *ResolvedQName*. *ResolvedQMgrName* může být název lokálního správce front.

Pokud je *ResolvedQName* sdílená fronta, kterou vlastní skupina sdílení front, do které lokální správce front patří, *ResolvedQMgrName* je název skupiny sdílení front. Pokud je fronta vlastněna jinou skupinou sdílení front, může být *ResolvedQName* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (povaha vrácené hodnoty je určena definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou otevřenou pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Pokud je otevřený objekt některý z následujících, *ResolvedQMgrName* je nastaven na mezery:

- Nejedná se o frontu
- Fronta, ale neotevřená pro procházení, vstup nebo výstup
- Fronta klastru se zadanou hodnotou MQOO_BIND_NOT_FIXED (nebo s hodnotou MQOO_BIND_AS_AS_Q_DEF v platnosti, když má atribut fronty **DefBind** hodnotu MQBND_BIND_NOT_FIXED)
- Distribuční seznam

Toto je výstupní pole. Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQOD_VERSION_3.

ObjectString (MQCHARV) pro MQOD

Pole ObjectString uvádí dlouhý název objektu.

Tato volba určuje dlouhý název objektu, který má být použit. Toto pole je odkazováno pouze pro určité hodnoty *ObjectType*a je ignorováno pro všechny ostatní hodnoty. Podrobnosti o tom, které hodnoty označují, že se toto pole používá, naleznete v popisu souboru *ObjectType* .

Pokud je parametr *ObjectString* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_OBJECT_STRING_ERROR.

Toto je vstupní pole. Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQCHARV.

Úplný název tématu lze sestavit ze dvou různých polí: *ObjectName* a *ObjectString*. Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat.

SelectionString (MQCHARV) pro MQOD

Jedná se o řetězec použitý k zadání kritérií výběru použitých při načítání zpráv z fronty.

Parametr *SelectionString* nesmí být uveden v následujících případech:

- Pokud *ObjectType* není MQOT_Q
- Pokud otevíraná fronta není otevřena pomocí jedné z voleb MQOO_BROWSE nebo MQOO_INPUT_ *

Pokud je v těchto případech uveden parametr *SelectionString* , volání selže s kódem příčiny MQRC_SELECTOR_INVALID_FOR_TYPE.

Pokud je parametr *SelectionString* zadán nesprávně, podle popisu způsobu použití struktury “MQCHARV-Řetězec délky proměnné” na stránce 296 , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_SELECTION_STRING_ERROR. Maximální délka parametru *SelectionString* je MQ_SELECTOR_LENGTH.

SelectionString použití je popsáno v části Selektory.

ResObjectŘetězec (MQCHARV) pro MQOD

Pole řetězce ResObjectje dlouhý název objektu poté, co správce front přeloží název uvedený v poli *ObjectName* .

Toto pole je vráceno pouze pro témata a aliasy fronty, které odkazují na objekt tématu.

Pokud je v souboru *ObjectString* uveden dlouhý název objektu a v souboru *ObjectName*není uveden žádný název, hodnota vrácená v tomto poli je stejná jako v souboru *ObjectString*.

Pokud je toto pole vynecháno (tj. ResObjectString.VSBufSize je nula), *ResObjectString* nebude vráceno, ale délka bude vrácena v ResObjectString.VSLength.

Je-li délka vyrovnávací paměti (uvedená v poli ResObjectStrng.VSBufSize) kratší než plná hodnota *ResObjectString*, řetězec bude oříznut a bude vrácen libovolný počet znaků nejvíce vpravo, který se vejde do poskytnuté vyrovnávací paměti.

Pokud je parametr *ResObjectString* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_RES_OBJECT_STRING_ERROR.

ResolvedType (MQLONG) pro MQOD

Typ interpretovaného (základního) objektu, který se otevírá.

Možné hodnoty jsou:

MQOT_Q
Vyřešený objekt je fronta. Tato hodnota se použije při přímém otevření fronty nebo při otevření alias fronty ukazující na frontu.

MQOT_TOPIC
Vyřešený objekt je téma. Tato hodnota se použije při přímém otevření tématu nebo při otevření alias fronty odkazující na objekt tématu.

MQOT_NONE
Vyřešený typ není ani fronta, ani téma.

MQOR-záznam objektu

Pomocí struktury MQOR zadejte název fronty a název správce front pro jednu cílovou frontu. MQOR je vstupní struktura pro volání MQOPEN a MQPUT1 .

Dostupnost

Struktura MQOR je k dispozici na následujících platformách:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Znaková sada a kódování

Data v MQOR musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pouľití

Poskytnutím pole těchto struktur ve volání MQOPEN můžete otevřít seznam front; tento seznam se nazývá rozdělovník. Každá zpráva vložená s použitím popisovače fronty vráceného tímto voláním MQOPEN je umístěna do každé z front v seznamu za předpokladu, že fronta byla úspěšně otevřena.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 505. Pole v MQOR pro MQOR

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
ObjectName (název objektu)	Není	Prázdný řetězec nebo mezery
ObjectQMgrNázev (název správce front objektů)	Není	Prázdný řetězec nebo mezery

Notes:

1. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyku C se jedná o proměnnou makra.MQOR_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQOR MyOR = {MQOR_DEFAULT};
```

Deklarace jazyka

Deklarace jazyka C pro objekt MQOR

```
typedef struct tagMQOR MQOR;
struct tagMQOR {
  MQCHAR48  ObjectName;      /* Object name */
  MQCHAR48  ObjectQMgrName;  /* Object queue manager name */
};
```

Deklarace jazyka COBOL pro objekt MQOR

```
**   MQOR structure
  10 MQOR.
**    Object name
   15 MQOR-OBJECTNAME     PIC X(48).
**    Object queue manager name
   15 MQOR-OBJECTQMGRNAME PIC X(48).
```

Deklarace PL/I pro MQOR

```
dcl
 1 MQOR based,
  3 ObjectName     char(48), /* Object name */
  3 ObjectQMgrName char(48); /* Object queue manager name */
```

Deklarace jazyka Visual Basic pro objekt MQOR

```
Type MQOR
  ObjectName     As String*48 'Object name'
  ObjectQMgrName As String*48 'Object queue manager name'
End Type
```

ObjectName (MQCHAR48) pro MQOR

Toto je stejné jako pole *ObjectName* ve struktuře MQOD (podrobnosti viz MQOD), kromě následujících:

- Musí to být název fronty.
- Nesmí se jednat o název modelové fronty.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ObjectQMgrNázev (MQCHAR48) pro objekt MQOR

Toto je stejné jako pole *ObjectQMgrName* ve struktuře MQOD (podrobnosti viz MQOD).

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

MQPD-deskriptor vlastnosti

Struktura **MQPD** se používá k definování atributů vlastnosti. Struktura je vstupní/výstupní parametr volání MQSETMP a výstupní parametr volání MQINQMP.

Dostupnost

Struktura **MQPD** je k dispozici na následujících platformách:

- AIX
- IBM i

- Linux
- Windows
- z/OS

a pro IBM MQ MQI clients.

Znaková sada a kódování

Data v souboru **MQPD** musí být ve znakové sadě aplikace a kódování aplikace (**MQENC_NATIVE**).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 506. Pole v MQPD

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	ID_STRUC_MQPD_STRUC_ID	'PD'
Verze (číslo verze struktury)	MQPD_VERSION_1	1
Volby (volby)	MQPD_NONE	0
Podpora (požadovaná podpora pro vlastnost zprávy)	MQPD_SUPPORT_OPTIONAL	0
Kontext (kontext zprávy, ke kterému vlastnost patří)	MQPD_NO_CONTEXT	0
CopyOptions (volby kopírování, ke které vlastnost patří)	MQCOPY_DEFAULT	0

Notes:

1. V programovacím jazyku C obsahuje proměnná makra MQPD_DEFAULT hodnoty uvedené v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQPD MyPD = {MQPD_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQPD

```
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 */
};
```

Deklarace jazyka COBOL pro MQPD

```
**   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.
```

Deklarace PL/I pro MQPD

```
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 */
```

Deklarace High Level Assembler pro MQPD

```
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)
```

StrucId (MQCHAR4) pro MQPD

Jedná se o identifikátor struktury struktury deskriptoru vlastností. Vždy se jedná o vstupní pole. Jeho hodnota je MQPD_STRUC_ID.

Hodnota musí být:

ID_STRUC_MQPD_STRUC_ID
Identifikátor pro strukturu deskriptoru vlastností.

Pro programovací jazyk C je definována také konstanta **MQPD_STRUC_ID_ARRAY** . Má stejnou hodnotu jako **MQPD_STRUC_ID**, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQPD

Toto je číslo verze struktury; hodnota musí být:

MQPD_VERSION_1
Version-1 struktura deskriptoru vlastností.

Následující konstanta určuje číslo verze aktuální verze:

MQPD_CURRENT_VERSION
Aktuální verze struktury deskriptoru vlastností.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQPD_VERSION_1**.

Volby (MQLONG) pro MQPD

Hodnota musí být:

MQPD_NONE
Nejsou uvedeny žádné volby

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQPD_NONE.

Podpora (MQLONG) pro MQPD

Toto pole popisuje, jaká úroveň podpory pro vlastnost zprávy je vyžadována pro správce front, aby mohla být zpráva obsahující tuto vlastnost vložena do fronty. Toto platí pouze pro vlastnosti definované pomocí IBM MQ; podpora všech ostatních vlastností je volitelná.

Pole je automaticky nastaveno na správnou hodnotu, pokud správce front zná vlastnost IBM MQ-defined. Není-li vlastnost rozpoznána, je přiřazen parametr MQPD_SUPPORT_OPTIONAL. Pokud správce front obdrží zprávu obsahující vlastnost definovanou proměnnou IBM MQ, kterou správce front rozpozná jako nesprávnou, opraví správce front hodnotu pole *Support* .

Při nastavení vlastnosti IBM MQ-defined pomocí volání MQSETMP na manipulátoru zprávy, kde byla nastavena volba MQCMHO_NO_VALIDATION, se *Support* stane vstupním polem. To umožňuje aplikaci vložit vlastnost IBM MQ-defined se správnou hodnotou, kde vlastnost není podporována připojeným správcem front, ale kde je zpráva určena ke zpracování v jiném správci front.

Hodnota MQPD_SUPPORT_OPTIONAL je vždy přiřazena k vlastnostem, které nejsou IBM MQdefinovanými vlastnostmi.

Pokud správce front IBM WebSphere MQ 7.0 , který podporuje vlastnosti zpráv, obdrží vlastnost, která obsahuje nerozpoznanou hodnotu *Support* , bude tato vlastnost považována za vlastnost:

- MQPD_SUPPORT_REQUIRED byla zadána, pokud je některá z nerozpoznaných hodnot obsažena v tabulce MQPD_REJECT_UNSUP_MASK.
- MQPD_SUPPORT_REQUIRED_IF_LOCAL byl zadán, pokud jsou některé z nerozpoznaných hodnot obsaženy v MQPD_ACCEPT_UNSUP_IF_XMIT_MASK
- Jinak bylo zadáno MQPD_SUPPORT_OPTIONAL.

Při použití volání MQSETMP pro manipulátor zprávy, kde je nastavena volba MQCMHO_NO_VALIDATION, je vrácena jedna z následujících hodnot voláním MQINQMP nebo lze zadat jednu z těchto hodnot:

MQPD_SUPPORT_OPTIONAL
Vlastnost je přijata správcem front i v případě, že není podporována. Vlastnost lze zrušit tak, aby zpráva směřovala do správce front, který nepodporuje vlastnosti zprávy. Tato hodnota je také přiřazena k vlastnostem, které nejsou IBM MQ-defined.

MQPD_SUPPORT_REQUIRED
Podpora pro vlastnost je povinná. Zpráva je odmítnuta správcem front, který nepodporuje vlastnost IBM MQ-defined. Volání MQPUT nebo MQPUT1 se nezdařilo s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_UNSUPPORTED_PROPERTY.

MQPD_SUPPORT_REQUIRED_IF_LOCAL
Zpráva je odmítnuta správcem front, který nepodporuje vlastnost IBM MQ-defined, pokud je zpráva určena pro lokální frontu. Volání MQPUT nebo MQPUT1 se nezdařilo s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_UNSUPPORTED_PROPERTY.

Volání MQPUT nebo MQPUT1 je úspěšné, pokud je zpráva určena pro vzdáleného správce front.

Jedná se o výstupní pole pro volání MQINQMP a vstupní pole pro volání MQSETMP, pokud byl popisovač zprávy vytvořen s nastavenou volbou MQCMHO_NO_VALIDATION. Počáteční hodnota tohoto pole je MQPD_SUPPORT_OPTIONAL.

Kontext (MQLONG) pro MQPD

Popisuje, do jakému kontextu zprávy vlastnost patří.

Pokud správce front obdrží zprávu obsahující vlastnost definovanou proměnnou IBM MQ, kterou správce front rozpozná jako nesprávnou, opraví správce front hodnotu pole *Context* .

Lze zadat následující volbu:

MQPD_USER_CONTEXT
Vlastnost je přidružena ke kontextu uživatele.

Pro nastavení vlastnosti přidružené ke kontextu uživatele pomocí volání MQSETMP není vyžadována žádná speciální autorizace.

Ve správci front IBM WebSphere MQ 7.0 je vlastnost přidružená ke kontextu uživatele uložena podle popisu pro MQOO_SAVE_ALL_CONTEXT. Volání MQPUT se zadaným parametrem MQPMO_PASS_ALL_CONTEXT způsobí zkopírování vlastnosti z uloženého kontextu do nové zprávy.

Pokud není dříve popsaná volba požadována, lze použít následující volbu:

MQPD_NO_CONTEXT
Vlastnost není přidružena ke kontextu zprávy.

Nerozpoznaná hodnota je odmítnuta s kódem *Reason* MQRC_PD_ERROR

Jedná se o vstupní/výstupní pole pro volání MQSETMP a výstupní pole pro volání MQINQMP. Počáteční hodnota tohoto pole je MQPD_NO_CONTEXT.

CopyOptions (MQLONG) pro MQPD

Popisuje, do kterého typu zpráv má být vlastnost zkopírována. Toto je pouze výstupní pole pro rozpoznané IBM MQ definované vlastnosti; IBM MQ nastavuje odpovídající hodnotu.

Pokud správce front obdrží zprávu obsahující vlastnost definovanou v souboru IBM MQ , kterou správce front rozpozná jako nesprávnou, správce front opraví hodnotu pole *CopyOptions* .

Můžete uvést jednu nebo více těchto voleb. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

MQCOPY_FORWARD
Tato vlastnost je zkopírována do předávané zprávy.

MQCOPY_PUBLISH
Tato vlastnost je zkopírována do zprávy přijaté odběratelem při publikování zprávy.

MQCOPY_REPLY
Tato vlastnost je zkopírována do zprávy odpovědi.

MQCOPY_REPORT
Tato vlastnost je zkopírována do zprávy sestavy.

MQCOPY_ALL
Tato vlastnost je zkopírována do všech typů následných zpráv.

Výchozí volba: Pro dodání výchozí sady voleb kopírování lze zadat následující volbu:

MQCOPY_DEFAULT
Tato vlastnost je zkopírována do předávané zprávy, do zprávy sestavy nebo do zprávy přijaté odběratelem při publikování zprávy.

Jedná se o ekvivalent k určení kombinace voleb MQCOPY_FORWARD, plus MQCOPY_REPORT, plus MQCOPY_PUBLISH.

Pokud není požadována žádná z dříve popsaných voleb, použijte následující volbu:

MQCOPY_NONE
Pomocí této hodnoty označíte, že nejsou zadány žádné další volby kopírování; programově neexistuje žádný vztah mezi touto vlastností a následnými zprávami. Tato hodnota je vždy vrácena pro vlastnosti deskriptoru zpráv.

Jedná se o vstupní/výstupní pole pro volání MQSETMP a výstupní pole pro volání MQINQMP. Počáteční hodnota tohoto pole je MQCOPY_DEFAULT.

MQPMO-Volby vložení zprávy

Struktura MQPMO umožňuje aplikaci určit volby, které řídí způsob umisťování zpráv do front nebo jejich publikování do témat. Struktura je vstupní/výstupní parametr volání MQPUT a MQPUT1 .

Verze

Aktuální verze MQPMO je MQPMO_VERSION_3. Určitá pole jsou k dispozici pouze v určitých verzích produktu MQPMO. Potřebujete-li portovat aplikace mezi několika prostředími, musíte se ujistit, že verze MQPMO je konzistentní ve všech prostředích. Pole, která existují pouze v konkrétních verzích struktury, jsou jako taková identifikována v tomto tématu a v popisech polí.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi MQPMO podporovanou prostředím, ale s počáteční hodnotou pole *Version* nastavenou na MQPMO_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole *Version* na číslo verze požadované verze.

Znaková sada a kódování

Data v MQPMO musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 507. Pole v MQPMO

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQPMO_STRUC_ID	'PMO¬'
Verze (číslo verze struktury)	MQPMO_VERSION_1	1
Volby (volby, které řídí akci MQPUT a MQPUT1)	MQPMO_NONE	0
Časový limit (vyhrazeno)	Není	-1
Kontext (popisovač objektu vstupní fronty)	Není	0
KnownDestKnownDest (počet zpráv úspěšně odeslaných do lokálních front)	Není	0
UnknownDestPočet (počet zpráv úspěšně odeslaných do vzdálených front)	Není	0
InvalidDestPočet (počet zpráv, které se nepodařilo odeslat)	Není	0
ResolvedQName (přeložený název cílové fronty)	Není	Prázdný řetězec nebo mezery
ResolvedQMgrNázev (vyřešený název správce cílových front)	Není	Prázdný řetězec nebo mezery
Poznámka: Zbývající pole jsou ignorována, pokud je *Version* menší než MQPMO_VERSION_2.		

Tabulka 507. Pole v MQPMO (pokračování)

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
RecsPresent (počet přítomných záznamů vložených zpráv nebo záznamů odpovědí)	Není	0
PutMsgRecFields (příznaky označující, která pole MQPMR jsou přítomna)	MQPMRF_NONE	0
PutMsgRecOffset (posun prvního záznamu vkládané zprávy od začátku MQPMO)	Není	0
ResponseRecOffset (posun prvního záznamu odpovědi od začátku MQPMO)	Není	0
PutMsgRecPtr (adresa záznamu první vkládané zprávy)	Není	Ukazatel Null nebo bajty s hodnotou Null
ResponseRecPtr (adresa prvního záznamu odpovědi)	Není	Ukazatel Null nebo bajty s hodnotou Null
Poznámka: Zbývající pole jsou ignorována, pokud je *Version* menší než MQPMO_VERSION_3.		
OriginalMsgpopisovač (původní popisovač zprávy)	MQHM_NONE	0
NewMsgHandle (nový popisovač zprávy)	MQHM_NONE	0
Akce (typ prováděné operace vložení a vztah mezi původní zprávou uvedenou v poli *OriginalMsgHandle* a novou zprávou uvedenou v poli *NewMsgHandle*).	MQACTP_NEW	0
PubLevel (úroveň odběru, na kterou je zaměřeno publikování)	Není	9

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQPMO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQPMO MyPMO = {MQPMO_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQPMO

```
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 */
};
```

Deklarace jazyka COBOL pro MQPMO

```
**   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.
```

Deklarace PL/I pro MQPMO

```
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 */
```

Deklarace High Level Assembler pro MQPMO

```
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)
```

Vizuální základní deklarace pro MQPMO

```
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
```

StrucId (MQCHAR4) pro MQPMO

Jedná se o identifikátor struktury struktury voleb vložení zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQPMO_STRUC_ID.

Hodnota musí být:

MQPMO_STRUC_ID
Identifikátor pro strukturu voleb vložení zprávy.

Pro programovací jazyk C je definována také konstanta MQPMO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQPMO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQPMO

Číslo verze struktury.

Hodnota musí být jedna z následujících:

MQPMO_VERSION_1
Version-1 struktura voleb vložení zprávy.

Tato verze je podporována ve všech prostředích.

MQPMO_VERSION_2
Version-2 struktura voleb vložení zprávy.

Tato verze je podporována v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

MQPMO_VERSION_3
Version-3 struktura voleb vložení zprávy.

Tato verze je podporována ve všech prostředích.

Pole, která existují pouze v novější verzi struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MQPMO_CURRENT_VERSION
Aktuální verze struktury voleb vložení zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQPMO_VERSION_1.

Volby (MQLONG) pro MQPMO

Pole Volby řídí činnost volání **MQPUT** a **MQPUT1** .

Volba rozsahu. Můžete zadat libovolnou volbu MQPMO nebo žádnou z voleb MQPMO. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace). Kombinace, které nejsou platné, jsou uvedeny; všechny ostatní kombinace jsou platné.

Následující volba řídí rozsah odesílaných publikací:

MQPMO_SCOPE_QMGR

Publikování je odesláno pouze odběratelům, kteří se přihlásili k odběru tohoto správce front. Publikování není předáno žádnému vzdálenému správci front publikování/odběru, který provedl odběr pro tohoto správce front, což potlačí jakékoli chování nastavené pomocí atributu tématu PUBSCOPE.

Poznámka: Není-li nastaven, je rozsah publikování určen atributem tématu PUBSCOPE.

Volby publikování. Následující volby řídí způsob publikování zpráv do tématu:

MQPMO_SUPPRESS_REPLYTO

Odběratelům nejsou předány žádné informace uvedené v polích *ReplyToQ* a *ReplyToQMgr* deskriptoru MQMD této publikace. Pokud je tato volba použita s volbou sestavy, která vyžaduje *ReplyToQ*, volání selže s MQRC_MISSING_REPLY_TO_Q.

MQPMO_RETAIN

Odesílaná publikace má být zachována správcem front. Toto uchování umožňuje odběrateli vyžádat si kopii tohoto publikování po jeho publikování pomocí volání MQSUBRQ. Umožňuje také odeslání publikace aplikacím, které po provedení tohoto publikování provádějí odběr (pokud se nerozhodnou, že nebudou odeslány pomocí volby MQSO_NEW_PUBLICATIONS_ONLY). Pokud je aplikaci odesláno publikování, které bylo zachováno, je označeno vlastností zprávy MQIsRetained tohoto publikování.

V každém uzlu stromu témat lze zachovat pouze jedno publikování. Proto, pokud již existuje zachované publikování pro toto téma, publikované jakoukoli jinou aplikací, je nahrazeno touto publikací. Proto je lepší vyhnout se tomu, aby více než jeden vydavatel uchovávající zprávy na stejné téma.

Pokud odběratel požaduje zachovaná publikování, může použitý odběr v tématu obsahovat zástupný znak. V takovém případě se může počet zachovaných publikování shodovat (na různých uzlech ve stromu témat) a do žádající aplikace může být odesláno několik publikování. Další podrobnosti viz popis volání “MQSUBRQ-Požadavek na odběr” na stránce 796 .

Informace o způsobu interakce zachovaných publikování s úrovněmi odběrů naleznete v tématu Zachování publikování.

Je-li použita tato volba a publikování nelze zachovat, zpráva nebude publikována a volání se nezdaří s hodnotou MQRC_PUT_NOT_ZACHOVÁNO.

MQPMO_NOT_OWN_SUBS

Informuje správce front o tom, že aplikace nechce odesílat žádná ze svých publikování na odběry, které vlastní. Odběry jsou považovány za vlastněné stejnou aplikací, pokud jsou manipulátory připojení stejné.

MQPMO_WARN_IF_NO_SUBS_MATCHED

Pokud publikování neodpovídá žádný odběr, vraťte kód dokončení (*CompCode*) MQCC_WARNING a kód příčiny MQRC_NO_SUBS_MATCHED.

Pokud operace vložení vrátí MQRC_NO_SUBS_MATCHED, publikování nebylo doručeno žádným odběrům. Je-li však pro operaci vložení určena volba MQPMO_RETAIN, zpráva se zachová a doručí do libovolného následně definovaného odpovídajícího odběru.

Odběr tématu odpovídá publikování, pokud je splněna některá z následujících podmínek:

- Zpráva je doručena do fronty odběrů.
- Zpráva by byla doručena do fronty odběrů, ale problém s frontou znamená, že zprávu nelze vložit do fronty, a následně byla umístěna do fronty nedoručených zpráv nebo vyřazena.
- Je definována uživatelská procedura směrování, která potlačuje doručení zprávy do odběru.

Odběr tématu neodpovídá publikování, pokud je splněna některá z následujících podmínek:

- Odběr má řetězec výběru, který neodpovídá publikování.
- Pro odběr byla určena volba MQSO_PUBLICATION_ON_REQUEST.
- Publikování nebylo doručeno, protože v operaci vložení byla zadána volba MQPMO_NOT_OWN_SUBS a odběr odpovídá identitě vydavatele.

Volby synchronizační položky. Následující volby se týkají účasti volání MQPUT nebo MQPUT1 v rámci pracovní jednotky:

MQPMO_SYNCPOINT
Požadavek je pracovat v rámci běžných protokolů jednotky práce. Zpráva není viditelná mimo jednotku práce, dokud není jednotka práce potvrzena. Pokud je jednotka práce odvolána, zpráva se odstraní.

Nejsou-li zadány hodnoty MQPMO_SYNCPOINT a MQPMO_NO_SYNCPOINT, je zahrnutí požadavku na vložení do protokolů jednotky práce určeno prostředím, ve kterém je spuštěn správce front, a nikoli prostředím, ve kterém je spuštěna aplikace. V systému z/OSje požadavek na vložení v rámci pracovní jednotky. Ve všech ostatních prostředích není požadavek na vložení v rámci pracovní jednotky.

Vzhledem k těmto rozdílům nesmí aplikace, kterou chcete portovat, povolit tuto volbu jako výchozí; výslovně uveďte buď MQPMO_SYNCPOINT, nebo MQPMO_NO_SYNCPOINT.

Neuvádějte MQPMO_SYNCPOINT s MQPMO_NO_SYNCPOINT.

MQPMO_NO_SYNCPOINT
Požadavek je pracovat mimo normální protokoly jednotky práce. Zpráva je k dispozici okamžitě a nelze ji odstranit zálohováním pracovní jednotky.

Nejsou-li zadány hodnoty MQPMO_NO_SYNCPOINT a MQPMO_SYNCPOINT, je zahrnutí požadavku na vložení do protokolů jednotky práce určeno prostředím, ve kterém je spuštěn správce front, a nikoli prostředím, ve kterém je spuštěna aplikace. V systému z/OSje požadavek na vložení v rámci pracovní jednotky. Ve všech ostatních prostředích není požadavek na vložení v rámci pracovní jednotky.

Vzhledem k těmto rozdílům nesmí aplikace, kterou chcete portovat, povolit tuto volbu jako výchozí; výslovně uveďte buď MQPMO_SYNCPOINT, nebo MQPMO_NO_SYNCPOINT.

Neuvádějte MQPMO_NO_SYNCPOINT s MQPMO_SYNCPOINT.

Volby Message-identifier a correlation-identifier. Následující volby požadují, aby správce front vygeneroval nový identifikátor zprávy nebo identifikátor korelace:

MQPMO_NEW_MSG_ID
Správce front nahradí obsah pole *MsgId* v deskriptoru MQMD novým identifikátorem zprávy. Tento identifikátor zprávy je odeslán spolu se zprávou a vrácen aplikaci při výstupu volání MQPUT nebo MQPUT1 .

Volbu MQPMO_NEW_MSG_ID lze zadat také při vložení zprávy do distribučního seznamu. Podrobnosti naleznete v popisu pole *MsgId* ve struktuře MQPMR.

Použití této volby zbavuje aplikaci potřeby resetovat pole *MsgId* na MQMI_NONE před každým voláním MQPUT nebo MQPUT1 .

MQPMO_NEW_CORREL_ID
Správce front nahradí obsah pole *CorrelId* v deskriptoru MQMD novým identifikátorem korelace. Tento identifikátor korelace je odeslán spolu se zprávou a vrácen aplikaci při výstupu volání MQPUT nebo MQPUT1 .

Volbu MQPMO_NEW_CORREL_ID lze zadat také při vložení zprávy do distribučního seznamu. Podrobnosti naleznete v popisu pole *CorrelId* ve struktuře MQPMR.

MQPMO_NEW_CORREL_ID je užitečné v situacích, kdy aplikace vyžaduje jedinečný identifikátor korelace.

Volby skupiny a segmentu. Následující volby se týkají zpracování zpráv ve skupinách a segmentech logických zpráv. Přečtěte si následující definice, které vám pomohou s porozuměním volbě.

Upozornění: Nemůžete použít segmentované nebo seskupené zprávy s publikováním/odběrem.

Fyzická zpráva
Jedná se o nejmenší jednotku informací, kterou lze umístit do fronty nebo z ní odebrat. Často odpovídá informacím zadaným nebo načteným v rámci jednoho volání MQPUT, MQPUT1nebo MQGET. Každá fyzická zpráva má svůj vlastní deskriptor zpráv (MQMD). Obecně se fyzické zprávy rozlišují podle různých hodnot pro identifikátor zprávy (pole*MsgId* v deskriptoru MQMD), ačkoli to správce front nevynucuje.

Logická zpráva
Logická zpráva je jedna jednotka informací o aplikaci pouze pro jiné platformy než z/OS . Při absenci systémových omezení je logická zpráva stejná jako fyzická zpráva. Jsou-li však logické zprávy extrémně velké, mohou být kvůli systémovým omezením vhodné nebo nezbytné rozdělit logickou zprávu do dvou nebo více fyzických zpráv, nazývaných *segmenty*.

Logická zpráva, která byla segmentována, se skládá ze dvou nebo více fyzických zpráv, které mají stejný nenulový identifikátor skupiny (pole*GroupId* v deskriptoru MQMD) a stejné pořadové číslo zprávy (pole*MsgSeqNumber* v deskriptoru MQMD). Segmenty jsou odlišeny odlišnými hodnotami pro posunutí segmentu (pole*Offset* v MQMD), které poskytuje posunutí dat ve fyzické zprávě od začátku dat v logické zprávě. Protože každý segment je fyzickou zprávou, segmenty v logické zprávě mají obvykle různé identifikátory zpráv.

Logická zpráva, která nebyla segmentována, ale pro kterou byla segmentace povolena odesílající aplikací, má také nenulový identifikátor skupiny, ačkoli v tomto případě existuje pouze jedna fyzická zpráva s tímto identifikátorem skupiny, pokud logická zpráva nepatří do skupiny zpráv. Logické zprávy, pro které byla segmentace blokována odesílající aplikací, mají identifikátor skupiny s hodnotou null (MQGI_NONE), pokud logická zpráva nepatří do skupiny zpráv.

Skupina zpráv
Skupina zpráv je sada jedné nebo více logických zpráv, které mají stejný nenulový identifikátor skupiny. Logické zprávy ve skupině se rozlišují podle různých hodnot pro pořadové číslo zprávy, což je celé číslo v rozsahu 1 až *n*, kde *n* je počet logických zpráv ve skupině. Je-li jedna nebo více logických zpráv segmentováno, existuje ve skupině více než *n* fyzických zpráv.

MQPMO_LOGICAL_ORDER
Tato volba informuje správce front o tom, jak aplikace vkládá zprávy do skupin a segmentů logických zpráv. Lze jej zadat pouze pro volání MQPUT; není platný pro volání MQPUT1 .

Je-li zadán parametr MQPMO_LOGICAL_ORDER, znamená to, že aplikace bude používat následná volání MQPUT, aby:

1. Vložila segmenty do každé logické zprávy kvůli zvýšení offsetu segmentu, počínaje 0, bez mezer.
2. Vložila všechny segmenty do jedné logické zprávy, a teprve pak vložila segment do další logické zprávy.
3. Vložila logické zprávy do každé skupiny zprávy, aby zvýšila pořadové číslo zprávy, počínaje 1, bez mezer. IBM MQ zvyšuje pořadové číslo zprávy automaticky.
4. Vložila všechny logické zprávy do jedné skupiny zpráv, a teprve pak vložila logické zprávy do další skupiny zpráv.

Podrobné informace o MQPMO_LOGICAL_ORDER naleznete v tématu Logické a fyzické řazení .

Volby kontextu. Zpracování kontextu zprávy řídí následující volby:

MQPMO_NO_CONTEXT
Identita i původní kontext jsou nastaveny tak, aby neoznačovaly žádný kontext. To znamená, že pole kontextu v deskriptoru MQMD jsou nastavena na:

- Mezery pro znaková pole
- Hodnoty Null pro bajtová pole
- Nuly pro číselná pole

MQPMO_DEFAULT_CONTEXT
Zpráva má mít k sobě přidružené výchozí informace o kontextu, a to jak pro identitu, tak pro původ. Správce front nastaví pole kontextu v deskriptoru zprávy následujícím způsobem:

Tabulka 508. Výchozí hodnoty informací o kontextu pro pole MQMD

Pole v deskriptoru MQMD	Použitá hodnota
UserIdentifier	Určuje se z prostředí, je-li to možné; jinak se nastaví na mezery.
AccountingToken	Určuje se z prostředí, je-li to možné; jinak nastavte na hodnotu MQACT_NONE.
ApplIdentityData	Nastavte na mezery.
PutApplType	Určeno z prostředí.
PutApplName	Určuje se z prostředí, je-li to možné; jinak se nastaví na mezery.
PutDate	Nastavit na datum, kdy je zpráva vložena.
PutTime	Nastavte na čas, kdy je zpráva vložena.
ApplOriginData	Nastavte na mezery.

Další informace o kontextu zprávy viz Kontext zprávy.

Jedná se o výchozí hodnoty a akce, pokud nejsou zadány žádné volby kontextu.

MQPMO_PASS_IDENTITY_CONTEXT
Zpráva má mít k sobě přidružené informace o kontextu. Kontext identity je převzat z popisovače fronty uvedeného v poli *Context* . Informace o původním kontextu jsou generovány správcem front stejným způsobem jako pro MQPMO_DEFAULT_CONTEXT (hodnoty naleznete v předchozí tabulce). Další informace o kontextu zprávy viz Kontext zprávy.

Pro volání MQPUT musí být fronta otevřena s volbou MQOO_PASS_IDENTITY_CONTEXT (nebo s volbou, která ji určuje). Pro volání MQPUT1 se provádí stejná kontrola autorizace jako pro volání MQOPEN s volbou MQOO_PASS_IDENTITY_CONTEXT.

MQPMO_PASS_ALL_CONTEXT
Zpráva má mít k sobě přidružené informace o kontextu. Kontext je převzat z popisovače fronty uvedeného v poli *Context* . Další informace o kontextu zprávy viz Řízení informací o kontextu.

Pro volání MQPUT musí být fronta otevřena s volbou MQOO_PASS_ALL_CONTEXT (nebo s volbou, která z ní vyplývá). Pro volání MQPUT1 se provádí stejná kontrola autorizace jako pro volání MQOPEN s volbou MQOO_PASS_ALL_CONTEXT.

MQPMO_SET_IDENTITY_CONTEXT
Zpráva má mít k sobě přidružené informace o kontextu. Aplikace určuje kontext identity ve struktuře MQMD. Informace o původním kontextu jsou generovány správcem front stejným způsobem jako pro MQPMO_DEFAULT_CONTEXT (hodnoty naleznete v předchozí tabulce). Další informace o kontextu zprávy viz Kontext zprávy.

Pro volání MQPUT musí být fronta otevřena s volbou MQOO_SET_IDENTITY_CONTEXT (nebo s volbou, která z ní vyplývá). Pro volání MQPUT1 se provádí stejná kontrola autorizace jako pro volání MQOPEN s volbou MQOO_SET_IDENTITY_CONTEXT.

MQPMO_SET_ALL_CONTEXT
Zpráva má mít k sobě přidružené informace o kontextu. Aplikace určuje identitu, původ a uživatelský kontext ve struktuře MQMD. Další informace o kontextu zprávy viz Kontext zprávy.

Pro volání MQPUT musí být fronta otevřena s volbou MQOO_SET_ALL_CONTEXT. Pro volání MQPUT1 se provádí stejná kontrola autorizace jako pro volání MQOPEN s volbou MQOO_SET_ALL_CONTEXT.

Můžete zadat pouze jednu z voleb kontextu MQPMO_ * _CONTEXT. Zadáte-li hodnotu none, bude se předpokládat hodnota MQPMO_DEFAULT_CONTEXT.

Volby vlastností. Následující volba se vztahuje k vlastnostem zprávy:

MQPMO_MD_FOR_OUTPUT_ONLY
Parametr deskriptoru zprávy musí být použit pouze pro výstup, aby vrátil deskriptor zprávy, která byla vložena. Pro vstup musí být použita pole deskriptoru zpráv přidružená k *NewMsgHandle*, *OriginalMsgHandle*nebo k oběma polím struktury **MQPMO** .

Není-li zadán platný popisovač zprávy, volání selže s kódem příčiny **MQRC_MD_ERROR**.

Volby odezvy vložení. Následující volby řídí odezvu vrácenou na volání MQPUT nebo MQPUT1 . Můžete uvést pouze jednu z těchto voleb. Pokud nejsou zadány parametry MQPMO_ASYNC_RESPONSE a MQPMO_SYNC_RESPONSE, předpokládá se MQPMO_RESPONSE_AS_Q_DEF nebo MQPMO_RESPONSE_AS_TOPIC_DEF.

MQPMO_ASYNC_RESPONSE
Volba MQPMO_ASYNC_RESPONSE požaduje, aby byla dokončena operace MQPUT nebo MQPUT1 , aniž by aplikace čekala na dokončení volání správce front. Použití této volby může zlepšit výkon systému zpráv, zejména pro aplikace používající vazby klienta. Aplikace může pomocí příkazu MQSTAT pravidelně kontrolovat, zda během předchozích asynchronních volání nedošlo k chybě.

Při použití této volby je zaručeno, že v deskriptoru MQMD budou vyplněna pouze následující pole;

- ApplIdentityData
- PutApplType
- PutApplName
- ApplOriginData

Kromě toho, pokud jsou jako volby zadány jedna nebo obě hodnoty MQPMO_NEW_MSG_ID nebo MQPMO_NEW_CORREL_ID, budou vráceny také hodnoty MsgId a CorrelId . (Parametr MQPMO_NEW_MSG_ID lze implicitně zadat zadáním prázdného pole MsgId).

Vyplní se pouze předchozí uvedená pole. Další informace, které by normálně byly vráceny ve struktuře MQMD nebo MQPMO, nejsou definovány.

Při vyžádání asynchronní odezvy vložení pro produkt MQPUT1nejsou definovány názvy ResolvedQName a ResolvedQMgrvrácené ve struktuře MQOD.

Při vyžádání asynchronní odezvy vložení pro MQPUT nebo MQPUT1nemusí CompCode a příčina MQCC_OK a MQRC_NONE nutně znamenat, že zpráva byla úspěšně vložena do fronty. Při vývoji aplikace MQI, která používá asynchronní odezvu vložení a vyžaduje potvrzení, že zprávy byly vloženy do fronty, musíte zkontrolovat kód CompCode i kódy příčiny z operací vložení a také použít MQSTAT k dotazování na informace o asynchronní chybě.

Ačkoli úspěch nebo selhání každého jednotlivého volání MQPUT nebo MQPUT1 nelze okamžitě vrátit, první chybu, ke které došlo v rámci asynchronního volání, lze určit později prostřednictvím volání MQSTAT.

Pokud se nezdaří doručení trvalé zprávy v synchronizačním bodu pomocí asynchronní odezvy vložení a pokusíte se potvrdit transakci, potvrzení se nezdaří a transakce se odvolá s kódem dokončení MQCC_FAILED a příčinou MQRC_BACKED_OUT. Aplikace může provést volání MQSTAT s cílem určit příčinu předchozího selhání MQPUT nebo MQPUT1 .

MQPMO_SYNC_RESPONSE
Zadáním tohoto typu odezvy vložení zajistíte, že operace MQPUT nebo MQPUT1 bude vždy zadána synchronně. Je-li operace vložení úspěšná, jsou dokončena všechna pole v MQMD a MQPMO.

Tato volba zajišťuje synchronní odezvu bez ohledu na výchozí hodnotu odezvy vložení definovanou ve frontě nebo objektu tématu.

MQPMO_RESPONSE_AS_Q_DEF
Je-li tato hodnota určena pro volání MQPUT, je použitý typ odezvy vložení převzat z hodnoty DEFPRESP určené ve frontě při prvním otevření aplikací.

- Pokud se jedná o frontu klastru a tato hodnota je určena pro volání MQPUT, je použitý typ odezvy vložení převzat z atributu **DEFPRESP** definovaného ve správci front *destination* , který vlastní konkrétní instanci fronty, do které je zpráva umístěna.

  Pokud existuje více instancí fronty klastrů lišících se v tomto atributu, vybere se hodnota od jedné z nich bez toho, že by šlo předpovědět, která to bude. Proto byste měli ve všech instancích tento atribut nastavit na stejnou hodnotu. Není-li to tento případ, je do protokolů správce front generována chybová zpráva AMQ9407. Viz také Jak jsou vyřešeny atributy cílového objektu pro aliasy, vzdálené fronty a fronty klastru?
- Pokud fronta není frontou klastru a tato hodnota je určena pro volání MQPUT, je použitý typ odezvy vložení převzat z atributu **DEFPRESP** definovaného v *lokálním* správci front, a to i v případě, že je cílový správce front vzdálený.

Pokud je aplikace klienta připojena ke správci front na úrovni starší než IBM WebSphere MQ 7.0, chová se, jako by byla zadána volba MQPMO_SYNC_RESPONSE.

Je-li tato volba určena pro volání MQPUT1 , hodnota atributu DEFPRESP není známa před odesláním požadavku na server. Standardně, pokud volání MQPUT1 používá MQPMO_SYNCPOINT, chová se jako pro MQPMO_ASYNC_RESPONSE a pokud používá MQPMO_NO_SYNCPOINT, chová se jako pro MQPMO_SYNC_RESPONSE. Toto výchozí chování však můžete přepsat nastavením vlastnosti Put1DefaultAlwaysSync v konfiguračním souboru klienta, viz CHANNELS stanza konfiguračního souboru klienta.

MQPMO_RESPONSE_AS_TOPIC_DEF
MQPMO_RESPONSE_AS_TOPIC_DEF je synonymem pro MQPMO_RESPONSE_AS_Q_DEF pro použití s objekty tématu.

Další volby. Následující volby řídí kontrolu autorizace, co se stane, když je správce front uveden do klidového stavu, a interpretaci názvů front a správců front:

Oprávnění MQPMO_ALTERNATE_USER_AUTHORITY
MQPMO_ALTERNATE_USER_AUTHORITY označuje, že pole *AlternateUserId* v parametru **ObjDesc** volání MQPUT1 obsahuje identifikátor uživatele, který má být použit k ověření oprávnění pro vložení zpráv do fronty. Volání může být úspěšné pouze v případě, že má produkt *AlternateUserId* oprávnění k otevření fronty s uvedenými volbami, bez ohledu na to, zda je k tomu autorizován identifikátor uživatele, pod kterým je aplikace spuštěna. (Toto neplatí pro zadané kontextové volby, které jsou však vždy kontrolovány proti identifikátoru uživatele, pod kterým je aplikace spuštěna.)

Tato volba je platná pouze s voláním MQPUT1 .

MQPMO_FAIL_IF_QUIESCING
Tato volba vynutí selhání volání MQPUT nebo MQPUT1 , pokud je správce front ve stavu uvedení do klidového stavu.

V systému z/OStato volba také vynutí selhání volání MQPUT nebo MQPUT1 , pokud je připojení (pro aplikaci CICS nebo IMS) v klidovém stavu.

Volání vrací kód dokončení MQCC_FAILED s kódem příčiny MQRC_Q_MGR_QUIESCING nebo MQRC_CONNECTION_QUIESCING.

MQPMO_RESOLVE_LOCAL_Q
Pomocí této volby můžete vyplnit *ResolvedQName* ve struktuře MQPMO názvem lokální fronty, do které je zpráva vložena, a *ResolvedQMgrName* názvem lokálního správce front, který je hostitelem lokální fronty. Další informace o MQPMO_RESOLVE_LOCAL_Q naleznete v tématu MQOO_RESOLVE_LOCAL_Q.

Máte-li oprávnění k vložení do fronty, máte požadované oprávnění k zadání tohoto příznaku ve volání MQPUT; není potřeba žádné speciální oprávnění.

Výchozí volba. Pokud nepotřebujete žádnou z popsaných voleb, použijte následující volbu:

MQPMO_NONE
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty. MQPMO_NONE je definována tak, aby pomáhala dokumentaci

programu; není zamýšleno, aby tato volba byla použita s jinou, ale protože její hodnota je nula, nelze takové použití zjistit.

MQPMO_NONE je vstupní pole. Počáteční hodnota pole *Options* je MQPMO_NONE.

Časový limit (MQLONG) pro MQPMO

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je -1.

Kontext (MQHOBJ) pro MQPMO

Je-li zadána volba MQPMO_PASS_IDENTITY_CONTEXT nebo MQPMO_PASS_ALL_CONTEXT, musí toto pole obsahovat manipulátor vstupní fronty, z něhož jsou převzaty informace o kontextu, které mají být přidruženy k vkládané zprávě.

Není-li zadána volba MQPMO_PASS_IDENTITY_CONTEXT ani MQPMO_PASS_ALL_CONTEXT, bude toto pole ignorováno.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

KnownDestPočet (MQLONG) pro MQPMO

Jedná se o počet zpráv, které aktuální volání MQPUT nebo MQPUT1 úspěšně odeslalo do front v distribučním seznamu, které jsou lokálními frontami. Tento počet nezahrnuje zprávy odeslané do front, které se interpretují do vzdálených front (i když se lokální přenosová fronta používá na počátku k uložení zprávy). Toto pole je také nastaveno při vkládání zprávy do jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole není nastaveno, pokud je hodnota *Version* menší než MQPMO_VERSION_1.

Toto pole není v systému z/OS definováno, protože rozdělovníky nejsou podporovány.

UnknownDestPočet (MQLONG) pro MQPMO

Jedná se o počet zpráv, které aktuální volání MQPUT nebo MQPUT1 úspěšně odeslalo do front v distribučním seznamu, které se interpretují do vzdálených front. Zprávy, které správce front dočasně uchovává ve formě distribučního seznamu, se počítají jako počet jednotlivých cílů, které tyto distribuční seznamy obsahují. Toto pole je také nastaveno při vkládání zprávy do jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole není nastaveno, pokud je hodnota *Version* menší než MQPMO_VERSION_1.

Toto pole není v systému z/OS definováno, protože rozdělovníky nejsou podporovány.

Počet InvalidDest(MQLONG) pro MQPMO

Jedná se o počet zpráv, které nebylo možné odeslat do front v rozdělovníku. Počet zahrnuje fronty, které se nepodařilo otevřít, stejně jako fronty, které byly úspěšně otevřeny, ale pro které operace vložení selhala. Toto pole je také nastaveno při vkládání zprávy do jedné fronty, která není v rozdělovníku.

Poznámka: Toto pole je nastaveno, pokud je parametr **CompCode** ve volání MQPUT nebo MQPUT1 nastaven na hodnotu MQCC_OK nebo MQCC_WARNING; může být nastaven, pokud je parametr **CompCode** nastaven na hodnotu MQCC_FAILED, ale nespoléhejte se na to v kódu aplikace.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole není nastaveno, pokud je hodnota *Version* menší než MQPMO_VERSION_1.

Toto pole není v systému z/OS definováno, protože rozdělovníky nejsou podporovány.

ResolvedQName (MQCHAR48) pro MQPMO

Jedná se o název cílové fronty po provedení rozlišování názvů lokálním správcem front. Vrácený název je název fronty, která existuje ve správci front identifikovaném pomocí *ResolvedQMgrName*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou; pokud je objekt distribučním seznamem nebo tématem, vrácená hodnota není definována.

Toto je výstupní pole. Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ResolvedQMgrNázev (MQCHAR48) pro MQPMO

Jedná se o název cílového správce front po provedení překladu názvů lokálním správcem front. Vrácený název je název správce front, který vlastní frontu identifikovanou pomocí *ResolvedQName*, a může se jednat o název lokálního správce front.

Pokud je *ResolvedQName* sdílená fronta, kterou vlastní skupina sdílení front, do které lokální správce front patří, *ResolvedQMgrName* je název skupiny sdílení front. Pokud je fronta vlastněna jinou skupinou sdílení front, může být *ResolvedQName* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (povaha vrácené hodnoty je určena definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou; pokud je objekt distribučním seznamem nebo tématem, vrácená hodnota není definována.

Toto je výstupní pole. Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

RecsPresent (MQLONG) pro MQPMO

Jedná se o počet záznamů vložených zpráv MQPMR nebo záznamů odpovědí MQRR, které byly poskytnuty aplikací. Toto číslo může být větší než nula pouze v případě, že je zpráva vkládána do rozdělovníku. Záznamy vložených zpráv a záznamy odpovědí jsou volitelné; aplikace nemusí poskytovat žádné záznamy, nebo se může rozhodnout poskytnout záznamy pouze jednoho typu. Pokud však aplikace poskytuje záznamy obou typů, musí poskytnout záznamy *RecsPresent* každého typu.

Hodnota *RecsPresent* nemusí být stejná jako počet cílů v rozdělovníku. Pokud je zadáno příliš mnoho záznamů, přebytečné záznamy se nepoužívají; pokud je zadáno příliš málo záznamů, použijí se výchozí hodnoty pro vlastnosti zpráv pro ta místa určení, která nevložila záznamy zpráv (viz *PutMsgRecOffset*).

Pokud je hodnota *RecsPresent* menší než nula nebo je větší než nula, ale zpráva není vložena do distribučního seznamu, volání selže s kódem příčiny MQRC_RECS_PRESENT_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQPMO_VERSION_2.

PutMsgRecFields (MQLONG) pro MQPMO

Toto pole obsahuje příznaky označující, která pole MQPMR jsou přítomna v záznamech vložených zpráv poskytnutých aplikací. Volbu *PutMsgRecFields* použijte pouze v případě, že je zpráva vkládána do rozdělovníku. Pole je ignorováno, pokud je hodnota *RecsPresent* nula, nebo jsou hodnoty *PutMsgRecOffset* i *PutMsgRecPtr* nula.

Pro pole, která jsou přítomna, používá správce front pro každé místo určení hodnoty z polí v odpovídajícím záznamu vkládané zprávy. Pro pole, která chybí, použije správce front hodnoty ze struktury MQMD.

Použijte jeden nebo více následujících příznaků, abyste označili, která pole jsou přítomna v záznamech vložených zpráv:

MQPMRF_MSG_ID
Pole identifikátoru zprávy je přítomno.

MQPMRF_CORREL_ID
Pole identifikátoru korelace je přítomno.

MQPMRF_GROUP_ID
Pole identifikátoru skupiny je přítomno.

MQPMRF_ZPĚTNÁ vazba
Pole zpětné vazby je přítomno.

MQPMRF_ACCOUNTING_TOKEN
Pole tokenu evidence je přítomno.

Zadáte-li tento příznak, zadejte do pole *Options* buď MQPMO_SET_IDENTITY_CONTEXT, nebo MQPMO_SET_ALL_CONTEXT; není-li tato podmínka splněna, volání selže s kódem příčiny MQRC_PMO_RECORD_FLAGS_ERROR.

Nejsou-li uvedena žádná pole MQPMR, lze zadat následující:

MQPMRF_NONE
Nejsou přítomna žádná pole záznamu vložená zpráva.

Je-li určena tato hodnota, *RecsPresent* musí být nula nebo obě hodnoty *PutMsgRecOffset* a *PutMsgRecPtr* musí být nula.

MQPMRF_NONE je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby tato konstanta byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Pokud *PutMsgRecFields* obsahuje neplatné příznaky nebo jsou poskytnuty záznamy vložených zpráv, ale *PutMsgRecFields* má hodnotu MQPMRF_NONE, volání se nezdaří s kódem příčiny MQRC_PMO_RECORD_FLAGS_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQPMRF_NONE. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQPMO_VERSION_2.

PutMsgRecOffset (MQLONG) pro MQPMO

Jedná se o posun v bajtech prvního záznamu zprávy vložení MQPMR od začátku struktury MQPMO. Posun může být kladný nebo záporný. *PutMsgRecOffset* se používá pouze v případě, že je zpráva vkládána do rozdělovníku. Je-li hodnota *RecsPresent* nula, pole se ignoruje.

Když je zpráva vkládána do distribučního seznamu, lze zadat pole jednoho nebo více záznamů vložených zpráv MQPMR, aby bylo možné určit určité vlastnosti zprávy pro každé místo určení jednotlivě; tyto vlastnosti jsou:

- Identifikátor zprávy
- Identifikátor korelace
- Identifikátor skupiny
- Hodnota zpětné vazby
- Token evidence

Nemusíte zadávat všechny tyto vlastnosti, ale bez ohledu na podmnožinu, kterou zvolíte, zadejte pole ve správném pořadí. Další podrobnosti viz popis struktury MQPMR.

Obvykle musí existovat tolik záznamů vložených zpráv, kolik existuje záznamů objektů určených produktem MQOD při otevření distribučního seznamu; každý záznam vložených zpráv dodává vlastnosti zpráv pro frontu určenou odpovídajícím záznamem objektu. Fronty v rozdělovníku, které se nepodařilo otevřít, musí mít stále přidělené záznamy zpráv na příslušných pozicích v poli, ačkoli vlastnosti zprávy jsou v tomto případě ignorovány.

Počet záznamů vložených zpráv se může lišit od počtu záznamů objektů. Pokud existuje méně záznamů vložených zpráv než záznamů objektů, vlastnosti zpráv pro místa určení, která nemají vloženy záznamy zpráv, jsou převzaty z odpovídajících polí v deskriptoru zpráv MQMD. Pokud existuje více záznamů vkládaných zpráv než záznamů objektů, přebytečné záznamy se nepoužívají (i když musí být stále možné k nim přistupovat). Záznamy vložených zpráv jsou volitelné, ale pokud jsou zadány, musí být *RecsPresent* z nich.

Zadejte záznamy vložených zpráv podobným způsobem jako záznamy objektů v produktu MQOD, a to buď zadáním offsetu v souboru *PutMsgRecOffset*, nebo zadáním adresy v souboru *PutMsgRecPtr* ; Podrobnosti o tom, jak to provést, viz pole *ObjectRecOffset* popsané v části“MQOD-Popisovač objektu” na stránce 482.

Nelze použít více než jeden z parametrů *PutMsgRecOffset* a *PutMsgRecPtr* ; volání selže s kódem příčiny MQRC_PUT_MSG_RECORDS_ERROR, pokud jsou obě nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQPMO_VERSION_2.

ResponseRecOffset (MQLONG) pro MQPMO

Jedná se o posun v bajtech prvního záznamu odpovědi MQRR od začátku struktury MQPMO. Posun může být kladný nebo záporný. *ResponseRecOffset* se používá pouze v případě, že je zpráva vkládána do rozdělovníku. Je-li hodnota *RecsPresent* nula, pole se ignoruje.

Při vkládání zprávy do distribučního seznamu můžete zadat pole jednoho nebo více záznamů odpovědí MQRR, abyste identifikovali fronty, do kterých nebyla zpráva úspěšně odeslána (pole*CompCode* v MQRR), a příčinu každého selhání (pole*Reason* v MQRR). Je možné, že zpráva nebyla odeslána buď proto, že se nezdařilo otevřít frontu, nebo proto, že se nezdařila operace vložení. Správce front nastaví záznamy odpovědí pouze v případě, že je výsledek volání smíšený (to znamená, že některé zprávy byly úspěšně odeslány, zatímco jiné selhaly, nebo všechny selhaly, ale z různých důvodů); kód příčiny MQRC_MULTIPLE_REASON z volání označuje tento případ. Pokud se stejný kód příčiny vztahuje na všechny fronty, vrátí se tato příčina v parametru **Reason** volání MQPUT nebo MQPUT1 a záznamy odezvy nejsou nastaveny.

Obvykle existuje tolik záznamů odpovědí, kolik je záznamů objektů určených produktem MQOD při otevření distribučního seznamu; v případě potřeby je každý záznam odezvy nastaven na kód dokončení a kód příčiny pro vložení do fronty identifikované odpovídajícím záznamem objektu. Fronty v rozdělovníku, které se nepodařilo otevřít, musí mít stále přidělené záznamy odpovědí na příslušných pozicích v poli, i když jsou nastaveny na kód dokončení a kód příčiny, který je výsledkem operace otevření, spíše než operace vložení.

Počet záznamů odpovědí se může lišit od počtu záznamů objektů. Pokud je méně záznamů odpovědí než záznamů objektů, aplikace nemusí být schopna identifikovat všechna místa určení, pro která operace vložení selhala, nebo příčiny selhání. Pokud existuje více záznamů odpovědí než záznamů objektů, přebytečné záznamy se nepoužívají (i když musí být stále možné k nim přistupovat). Záznamy odpovědí jsou volitelné, ale pokud jsou zadány, musí z nich být *RecsPresent* .

Poskytněte záznamy odezvy podobným způsobem jako záznamy objektů v produktu MQOD, buď uvedením offsetu v produktu *ResponseRecOffset*, nebo uvedením adresy v produktu *ResponseRecPtr* ; Podrobnosti o tom, jak to provést, viz pole *ObjectRecOffset* popsané v části “MQOD-Popisovač objektu” na stránce 482. Avšak nepoužívejte více než jeden z *ResponseRecOffset* a *ResponseRecPtr* ; volání selže s kódem příčiny MQRC_RESPONSE_RECORDS_ERROR, pokud jsou obě nenulové.

Pro volání MQPUT1 musí být toto pole nula. Důvodem je skutečnost, že informace o odezvě (je-li požadována) jsou vráceny v záznamech odezvy určených deskriptorem objektu MQOD.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQPMO_VERSION_2.

PutMsgRecPtr (MQPTR) pro MQPMO

Jedná se o adresu prvního záznamu zprávy vložení MQPMR. Volbu *PutMsgRecPtr* použijte pouze v případě, že je zpráva vkládána do rozdělovníku. Je-li hodnota *RecsPresent* nula, pole se ignoruje.

Můžete použít buď *PutMsgRecPtr* , nebo *PutMsgRecOffset* , chcete-li uvést záznamy vkládané zprávy, ale ne obojí; podrobnosti viz “PutMsgRecOffset (MQLONG) pro MQPMO” na stránce 515. Pokud nepoužíváte *PutMsgRecPtr*, nastavte jej na ukazatel Null nebo na nulový počet bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou typu all-null. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQPMO_VERSION_2.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky, přičemž počáteční hodnota je bajtový řetězec s hodnotou typu all-null.

ResponseRecPtr (MQPTR) pro MQPMO

Jedná se o adresu prvního záznamu odpovědi MQRR. *ResponseRecPtr* se používá pouze v případě, že je zpráva vkládána do rozdělovníku. Je-li hodnota *RecsPresent* nula, pole se ignoruje.

Chcete-li uvést záznamy odezvy, ale ne obojí, použijte buď *ResponseRecPtr* , nebo *ResponseRecOffset* ; podrobnosti viz "ResponseRecOffset (MQLONG) pro MQPMO" na stránce 516. Pokud parametr *ResponseRecPtr* nepoužijete, nastavte jej na ukazatel Null nebo na bajty s hodnotou null.

Pro volání MQPUT1 musí být toto pole ukazatelem Null nebo nulovým počtem bajtů. Důvodem je skutečnost, že informace o odezvě (je-li požadována) jsou vráceny v záznamech odezvy určených deskriptorem objektu MQOD.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou typu all-null. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQPMO_VERSION_2.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky, přičemž počáteční hodnota je bajtový řetězec s hodnotou typu all-null.

OriginalMsgManipulátor (MQHMSG) pro MQPMO

Jedná se o volitelný popisovač zprávy. Je možné, že byl dříve načten z fronty. Použití tohoto popisovače je podmíněno hodnotou pole *Action* ; viz také NewMsg.

Obsah původního popisovače zprávy nebude změněn voláním **MQPUT** nebo **MQPUT1** .

Toto je vstupní pole. Počáteční hodnota tohoto pole je **MQHM_NONE**. Toto pole je ignorováno, pokud je verze menší než **MQPMO_VERSION_3**.

NewMsgManipulátor (MQHMSG) pro MQPMO

Jedná se o nepovinný popisovač pro vkládanou zprávu, který je předmětem hodnoty pole Akce. Definuje vlastnosti zprávy a přepisuje hodnoty *OriginalMsgHandle*, pokud jsou zadány.

Při návratu z volání **MQPUT** nebo **MQPUT1** obsah popisovače odráží zprávu, která byla skutečně vložena.

Toto je vstupní pole. Počáteční hodnota tohoto pole je **MQHM_NONE**. Toto pole je ignorováno, pokud je verze menší než **MQPMO_VERSION_3**.

Akce (MQLONG) pro MQPMO

Tato volba určuje typ prováděné operace vložení a vztah mezi původní zprávou určenou v poli manipulátoru OriginalMsga novou zprávou určenou v poli manipulátoru NewMsg. Vlastnosti zprávy jsou vybrány správcem front na základě zadané hodnoty akce.

Obsah deskriptoru zprávy můžete zadat pomocí parametru MsgDesc ve volání MQPUT nebo MQPUT1 . Alternativně je možné nezadat parametr MsgDesc nebo určit, že se jedná pouze o výstup, a to zahrnutím MQPMO_MD_FOR_OUTPUT_ONLY do pole Volby struktury MQPMO.

Není-li zadán parametr MsgDesc nebo je-li zadán pouze pro výstup, je deskriptor zprávy pro novou zprávu naplněn daty z polí popisovače zprávy MQPMO v souladu s pravidly popsanými v tomto tématu.

Nastavení kontextu a předávání aktivit popsaných v tématu Řízení informací o kontextu se projeví po vytvoření deskriptoru zprávy.

Je-li zadána nesprávná hodnota akce, volání se nezdaří s kódem příčiny MQRC_ACTION_ERROR.

Lze zadat libovolnou z následujících akcí:

MQACTP_NEW
Probíhá vložení nové zprávy a program nezadává žádný vztah k předchozí zprávě. Deskriptor zprávy je sestaven takto:

- Je-li ve volání MQPUT nebo MQPUT1 zadána hodnota MsgDesc a MQPMO_MD_FOR_OUTPUT_ONLY není v produktu MQPMO.Options, toto se používá jako neupravený deskriptor zprávy.
- Pokud není zadána hodnota MsgDesc , nebo je MQPMO_MD_FOR_OUTPUT_ONLY v produktu MQPMO.Options pak správce front vygeneruje deskriptor zprávy pomocí kombinace vlastností z popisovače OriginalMsga popisovače NewMsg. Všechna pole deskriptoru zpráv explicitně nastavená v popisovači nové zprávy mají přednost před poli v původním popisovači zprávy.

Data zprávy jsou převzata z parametru vyrovnávací paměti MQPUT nebo MQPUT1 .

MQACTP_FORWARD
Probíhá předávání dříve načtené zprávy. Původní popisovač zprávy uvádí zprávu, která byla dříve načtena.

Nový popisovač zprávy určuje veškeré úpravy vlastností (včetně všech v deskriptoru zprávy) v původním popisovači zprávy.

Deskriptor zprávy je sestaven takto:

- Je-li ve volání MQPUT nebo MQPUT1 zadána hodnota MsgDesc a MQPMO_MD_FOR_OUTPUT_ONLY není v produktu MQPMO.Options, toto se používá jako neupravený deskriptor zprávy.
- Pokud není zadána hodnota MsgDesc , nebo je MQPMO_MD_FOR_OUTPUT_ONLY v produktu MQPMO.Options pak správce front vygeneruje deskriptor zprávy pomocí kombinace vlastností z popisovače OriginalMsga popisovače NewMsg. Všechna pole deskriptoru zpráv explicitně nastavená v popisovači nové zprávy mají přednost před poli v původním popisovači zprávy.
- Je-li v produktu MQPMO.Options, pak jsou splněny.

Vlastnosti zprávy jsou sestaveny takto:

- Všechny vlastnosti z původního popisovače zprávy, které mají v produktu MQPD.CopyOptions
- Všechny vlastnosti z nového popisovače zprávy. Pro každou vlastnost v popisovači nové zprávy, která má stejný název jako vlastnost v původním popisovači zprávy, je hodnota převzata z nového popisovače zprávy. Jedinou výjimkou z tohoto pravidla je speciální případ, kdy má vlastnost v popisovači nové zprávy stejný název jako vlastnost v původním popisovači zprávy, ale hodnota vlastnosti je null. V tomto případě je vlastnost odebrána ze zprávy.

Data zprávy, která mají být předána, jsou převzata z parametru vyrovnávací paměti MQPUT nebo MQPUT1 .

MQACTP_REPLY
Byla provedena odpověď na dříve načtenou zprávu. Původní popisovač zprávy uvádí zprávu, která byla dříve načtena.

Nový popisovač zprávy určuje veškeré úpravy vlastností (včetně všech v deskriptoru zprávy) v původním popisovači zprávy.

Deskriptor zprávy je sestaven takto:

- Je-li ve volání MQPUT nebo MQPUT1 zadána hodnota MsgDesc a MQPMO_MD_FOR_OUTPUT_ONLY není v produktu MQPMO.Options, toto se používá jako neupravený deskriptor zprávy.
- Pokud není zadána hodnota MsgDesc , nebo je MQPMO_MD_FOR_OUTPUT_ONLY v produktu MQPMO.Options, pak se pole deskriptoru počáteční zprávy vybírají takto:

Tabulka 509. Transformace popisovače zprávy odpovědi

Pole v deskriptoru MQMD	**Použitá hodnota**
Sestava	Je-li MQRO_PASS_DISCARD_AND_EXPIRAČNÍ a MQRO_DISCARD_MSG jsou nastaveny: MQRO_DISCARD_MSG jinak MQRO_NONE

Tabulka 509. Transformace popisovače zprávy odpovědi (pokračování)	
Pole v deskriptoru MQMD	**Použitá hodnota**
MsgType	MQMT_REPLY
Vypršení	Je-li MQRO_PASS_DISCARD_AND_EXPIRAČNÍ je nastaveno: Zkopírováno ze vstupní zprávy jinak MQEI_UNLIMITED
Zpětná vazba	MQFB_NONE
MsgId	Je-li nastaveno MQPMO_NEW_MSG_ID: Vygeneruje se nový identifikátor zprávy jinak, je-li nastaveno MQRO_PASS_MSG_ID: Zkopírováno ze vstupní zprávy jinak MQMI_NONE
CorrelId	Je-li nastaveno MQPMO_NEW_CORREL_ID: Vygeneruje se nový identifikátor korelace. jinak, je-li nastaveno MQRO_COPY_MSG_ID_TO_CORREL_ID: Zkopírováno z pole MsgId Vstupní zpráva jinak, je-li nastaveno MQRO_PASS_CORREL_ID: Zkopírováno z pole CorrelId Vstupní zpráva jinak MQCI_NONE
BackoutCount	0
ReplyToQ	Mezery
ReplyToQMgr	Mezery
GroupId	MQGI_NONE
MsgSeqNumber	1
Offset	0
MsgFlags	MQMF_NONE
OriginalLength	MQOL_UNDEFINED

- Deskriptor zprávy je poté upraven novým popisovačem zprávy-všechna pole deskriptoru zprávy explicitně nastavená jako vlastnosti v novém popisovači zprávy mají přednost před poli deskriptoru zprávy, jak bylo popsáno výše.

Vlastnosti zprávy jsou sestaveny takto:

- Všechny vlastnosti z původního popisovače zprávy, které mají v produktu MQPD.CopyOptions
- Všechny vlastnosti z nového popisovače zprávy. Pro každou vlastnost v popisovači nové zprávy, která má stejný název jako vlastnost v původním popisovači zprávy, je hodnota převzata z nového popisovače zprávy. Jedinou výjimkou z tohoto pravidla je speciální případ, kdy má vlastnost v popisovači nové zprávy stejný název jako vlastnost v původním popisovači zprávy, ale hodnota vlastnosti je null. V tomto případě je vlastnost odebrána ze zprávy.

Data zprávy, která mají být postoupena, jsou převzata z parametru vyrovnávací paměti MQPUT/MQPUT1 .

MQACTP_REPORT
Sestava je generována jako výsledek dříve načtené zprávy. Původní popisovač zprávy uvádí zprávu, která způsobuje generování sestavy.

Nový popisovač zprávy určuje veškeré úpravy vlastností (včetně všech v deskriptoru zprávy) v původním popisovači zprávy.

Deskriptor zprávy je sestaven takto:

- Je-li ve volání MQPUT nebo MQPUT1 zadána hodnota MsgDesc a MQPMO_MD_FOR_OUTPUT_ONLY není v produktu MQPMO.Options, toto se používá jako neupravený deskriptor zprávy.
- Pokud není zadána hodnota MsgDesc , nebo je MQPMO_MD_FOR_OUTPUT_ONLY v produktu MQPMO.Options pak pole deskriptoru počáteční zprávy jsou zvolena takto:

Tabulka 510. Transformace popisovače zprávy sestavy

Pole v deskriptoru MQMD	**Použitá hodnota**
Sestava	Pokud je hodnota MQRO_PASS_DISCARD_AND_EXPIRAČNÍ a MQRO_DISCARD_MSG jsou nastaveny: MQRO_DISCARD_MSG jinak MQRO_NONE
MsgType	MQMT_REPORT
Vypršení	Je-li MQRO_PASS_DISCARD_AND_EXPIRAČNÍ je nastaveno: Zkopírováno ze vstupní zprávy jinak MQEI_UNLIMITED
MsgId	Je-li nastaveno MQPMO_NEW_MSG_ID: Vygeneruje se nový identifikátor zprávy jinak, je-li nastaveno MQRO_PASS_MSG_ID: Zkopírováno ze vstupní zprávy jinak MQMI_NONE
CorrelId	Je-li nastaveno MQPMO_NEW_CORREL_ID: Vygeneruje se nový identifikátor korelace. jinak, je-li nastaveno MQRO_COPY_MSG_ID_TO_CORREL_ID: Zkopírováno z pole MsgId Vstupní zpráva jinak, je-li nastaveno MQRO_PASS_CORREL_ID: Zkopírováno z pole CorrelId Vstupní zpráva jinak MQCI_NONE
BackoutCount	0
ReplyToQ	Mezery
ReplyToQMgr	Mezery

Tabulka 510. Transformace popisovače zprávy sestavy (pokračování)	
Pole v deskriptoru MQMD	**Použitá hodnota**
OriginalLength	Nastavit na *BufferLength*

- Deskriptor zprávy je poté upraven novým popisovačem zprávy-všechna pole deskriptoru zprávy explicitně nastavená jako vlastnosti v novém popisovači zprávy mají přednost před poli deskriptoru zprávy, jak bylo popsáno výše.

Vlastnosti zprávy jsou sestaveny takto:

- Všechny vlastnosti z původního manipulátoru zprávy, které mají v produktu MQPD.CopyOptions
- Všechny vlastnosti z nového popisovače zprávy. Pro každou vlastnost v popisovači nové zprávy, která má stejný název jako vlastnost v původním popisovači zprávy, je hodnota převzata z nového popisovače zprávy. Jedinou výjimkou z tohoto pravidla je speciální případ, kdy má vlastnost v popisovači nové zprávy stejný název jako vlastnost v původním popisovači zprávy, ale hodnota vlastnosti je null. V tomto případě je vlastnost odebrána ze zprávy.

Pole Feedback ve výsledném deskriptoru MQMD představuje sestavu, která má být vygenerována. Hodnota zpětné vazby MQFB_NONE způsobí selhání volání MQPUT nebo MQPUT1 s kódem příčiny MQRC_FEEDBACK_ERROR.

Chcete-li zvolit uživatelská data zprávy sestavy, produkt IBM MQ konzultuje pole Sestava a Zpětná vazba ve výsledném deskriptoru MQMD a parametry Vyrovnávací paměť a BufferLength volání MQPUT nebo MQPUT1 .

- Je-li zpětná vazba MQFB_COA, MQFB_COD nebo MQFB_EXPIRATION, zkontroluje se hodnota sestavy.
- Má-li některý z následujících případů hodnotu true, použijí se úplná data zprávy z vyrovnávací paměti pro délku BufferLength .
  - Zpětná vazba je MQFB_EXPIRATION a sestava obsahuje MQRO_EXPIRATION_WITH_FULL_DATA
  - Zpětná vazba je MQFB_COD a sestava obsahuje MQRO_COD_WITH_FULL_DATA
  - Zpětná vazba je MQFB_COA a sestava obsahuje MQRO_COA_WITH_FULL_DATA
- Je-li některý z následujících případů pravdivý, použije se prvních 100 bajtů zprávy (nebo BufferLength , je-li menší než 100) z vyrovnávací paměti.
  - Zpětná vazba je MQFB_EXPIRATION a sestava obsahuje MQRO_EXPIRATION_WITH_DATA
  - Zpětná vazba je MQFB_COD a sestava obsahuje MQRO_COD_WITH_DATA
  - Zpětná vazba je MQFB_COA a sestava obsahuje MQRO_COA_WITH_DATA
- Je-li zpětná vazba MQFB_EXPIRATION, MQFB_COD nebo MQFB_COA a sestava neobsahuje volby * _WITH_FULL_DATA nebo * _WITH_DATA relevantní pro tuto hodnotu Feeback, nebudou do zprávy zahrnuta žádná uživatelská data.
- Pokud zpětná vazba přebírá jinou hodnotu, než je uvedeno výše, použijí se vyrovnávací paměť a BufferLength jako obvykle.

Odvození uživatelských dat popsaných v předchozím seznamu je také uvedeno v následující tabulce:

Tabulka 511. Zdroj uživatelských dat			
	MQFB_COA	**MQFB_COD**	**MQFB_EXPIRATION**
MQRO_EXPIRATION _WITH_FULL_DATA	Není	Není	Vyrovnávací paměť (délka vyrovnávací paměti)
MQRO_COD_WITH _FULL_DATA	Není	Vyrovnávací paměť (délka vyrovnávací paměti)	Není

Tabulka 511. Zdroj uživatelských dat (pokračování)			
	MQFB_COA	**MQFB_COD**	**MQFB_EXPIRATION**
MQRO_COA_WITH_FULL_DATA	Vyrovnávací paměť (délka vyrovnávací paměti)	Není	Není
MQRO_EXPIRATION_WITH_DATA	Není	Není	Vyrovnávací paměť (prvních 100 bajtů)
MQRO_COD_WITH_DATA	Není	Vyrovnávací paměť (prvních 100 bajtů)	Není
MQRO_COA_WITH_DATA	Vyrovnávací paměť (prvních 100 bajtů)	Není	Není

PubLevel (MQLONG) pro MQPMO

Počáteční hodnota tohoto pole je 9. Úroveň odběru, na kterou se zaměřuje tato publikace. Toto publikování obdrží pouze ty odběry s nejvyšší SubLevel , která je menší nebo rovna této hodnotě. Tato hodnota musí být v rozsahu nula až 9; nula je nejnižší úroveň. Pokud však bylo publikování zachováno, není již k dispozici pro odběratele na vyšších úrovních, protože je znovu publikována na PubLevel 1.

Další informace naleznete v tématu Zachycení publikací.

MQPMR-záznam vkládající zprávy

Pomocí struktury MQPMR můžete určit různé vlastnosti zprávy pro jedno místo určení při vkládání zprávy do distribučního seznamu. MQPMR je vstupní/výstupní struktura pro volání MQPUT a MQPUT1 .

Dostupnost

Struktura MQPMR je k dispozici na následujících platformách:

- AIX
- IBM i
- Linux
- Windows

a pro klienty IBM MQ připojené k těmto systémům.

Znaková sada a kódování

Data v MQPMR musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pouľití

Poskytnutím pole těchto struktur ve volání MQPUT nebo MQPUT1 můžete zadat různé hodnoty pro každou cílovou frontu v distribučním seznamu. Některá pole jsou pouze vstupní, jiná jsou vstupní/výstupní.

Poznámka: Tato struktura je neobvyklá v tom, že nemá pevné rozložení. Pole v této struktuře jsou volitelná a přítomnost nebo nepřítomnost jednotlivých polí je označena příznaky v poli *PutMsgRecFields* v MQPMO. Pole, která jsou přítomna v produktu ***, se musí vyskytovat v následujícím pořadí*** :

- *MsgId*
- *CorrelId*

- *GroupId*
- *Feedback*
- *AccountingToken*

Pole, která chybí, nezabírají v záznamu žádné místo.

Protože MQPMR nemá pevné rozvržení, není v záhlaví, COPY a INCLUDE pro podporované programovací jazyky uvedena žádná jeho definice. Programátor aplikace musí vytvořit deklaraci obsahující pole, která jsou vyžadována aplikací, a nastavit příznaky v souboru *PutMsgRecFields* tak, aby označovaly pole, která jsou přítomna.

Pole

Pro tuto strukturu nejsou definovány žádné počáteční hodnoty, protože v záhlaví nejsou uvedeny žádné deklarace struktury, soubory COPYa INCLUDE pro podporované programovací jazyky. Ukázkové deklarace ukazují, jak deklarovat strukturu, pokud jsou vyžadována všechna pole.

Tabulka 512. Pole v MQPMR

Název pole	**Popis pole**
MsgId	Identifikátor zprávy
CorrelId	Identifikátor korelace
GroupId	Identifikátor skupiny
Zpětná vazba	Zpětná vazba nebo kód příčiny
AccountingToken	Token evidence

Deklarace jazyka

C prohlášení pro MQPMR

```
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 */
};
```

Deklarace jazyka COBOL pro MQPMR

```
**   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).
```

Deklarace PL/I pro MQPMR

```
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 */
```

Vizuální základní deklarace pro MQPMR

```
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
```

MsgId (MQBYTE24) pro MQPMR

Jedná se o identifikátor zprávy, který má být použit pro zprávu odeslanou do fronty s názvem, který byl určen odpovídajícím prvkem v poli struktur MQOR ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *MsgId* v deskriptoru MQMD pro vložení do jedné fronty.

Pokud toto pole není v záznamu MQPMR přítomno nebo existuje méně záznamů MQPMR než místa určení, hodnota v MQMD se použije pro místa určení, která nemají záznam MQPMR obsahující pole *MsgId* . Je-li tato hodnota MQMI_NONE, vygeneruje se nový identifikátor zprávy pro *každý* z těchto míst určení (tj. žádné dva z těchto míst určení nemají stejný identifikátor zprávy).

Je-li uvedeno MQPMO_NEW_MSG_ID, vygenerují se nové identifikátory zpráv pro všechna místa určení v distribučním seznamu bez ohledu na to, zda mají záznamy MQPMR. Liší se od způsobu zpracování MQPMO_NEW_CORREL_ID (viz pole *CorrelId*).

Toto je vstupní/výstupní pole.

CorrelId (MQBYTE24) pro MQPMR

Jedná se o identifikátor korelace, který má být použit pro zprávu odeslanou do fronty s názvem určeným příslušným prvkem v poli struktur MQOR poskytnutém ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *CorrelId* v deskriptoru MQMD pro vložení do jedné fronty.

Pokud toto pole není v záznamu MQPMR přítomno nebo existuje méně záznamů MQPMR než místa určení, hodnota v MQMD se použije pro místa určení, která nemají záznam MQPMR obsahující pole *CorrelId* .

Je-li zadáno MQPMO_NEW_CORREL_ID, vygeneruje se *jeden* nový identifikátor korelace a použije se pro všechna místa určení v distribučním seznamu bez ohledu na to, zda mají záznamy MQPMR. Liší se od způsobu zpracování MQPMO_NEW_MSG_ID (viz pole *MsgId*).

Toto je vstupní/výstupní pole.

GroupId (MQBYTE24) pro MQPMR

GroupId je identifikátor skupiny, který má být použit pro zprávu odeslanou do fronty s názvem určeným odpovídajícím prvkem v poli struktur MQOR zadaných ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *GroupId* v deskriptoru MQMD pro vložení do jedné fronty.

Pokud toto pole není v záznamu MQPMR přítomno nebo existuje méně záznamů MQPMR než místa určení, hodnota v MQMD se použije pro místa určení, která nemají záznam MQPMR obsahující pole *GroupId* . Hodnota je zpracována tak, jak je dokumentováno v Fyzickém pořadí ve frontě, ale s následujícími rozdíly:

- GroupId se vytvoří z názvu QMName a časového razítka. Chcete-li tedy zachovat jedinečné názvy správce front GroupId , zachovejte také jedinečné názvy správců front. Také nenastavujte hodiny zpět na počítači správců front.
- V případech, kdy by byl použit nový identifikátor skupiny, správce front vygeneruje pro každé místo určení jiný identifikátor skupiny (tj. žádné dva cíle nemají stejný identifikátor skupiny).
- V těch případech, kdy by se použila hodnota v poli, volání selže s kódem příčiny MQRC_GROUP_ID_ERROR

Toto je vstupní/výstupní pole.

Zpětná vazba (MQLONG) pro MQPMR

Jedná se o kód zpětné vazby, který má být použit pro zprávu odeslanou do fronty s názvem určeným příslušným prvkem v poli struktur MQOR poskytnutém ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *Feedback* v deskriptoru MQMD pro vložení do jedné fronty.

Pokud toto pole není přítomno, použije se hodnota v MQMD.

Toto je vstupní pole.

AccountingToken (MQBYTE32) pro MQPMR

Jedná se o token evidence, který má být použit pro zprávu odeslanou do fronty s názvem určeným příslušným prvkem v poli struktur MQOR určeném ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *AccountingToken* v deskriptoru MQMD pro vložení do jedné fronty. Informace o obsahu tohoto pole naleznete v popisu položky *AccountingToken* v souboru "MQMD-Deskriptor zpráv" na stránce 424 .

Pokud toto pole není přítomno, použije se hodnota v MQMD.

Toto je vstupní pole.

MQRFH-Pravidla a formátování záhlaví

Struktura MQRFH definuje rozvržení pravidel a záhlaví formátování. Pomocí tohoto záhlaví odešlete řetězcová data ve formě dvojic název-hodnota.

Dostupnost

Všechny systémy IBM MQ a IBM MQ MQI clients připojené k těmto systémům.

Název formátu

MQFMT_RF_HEADER

Znaková sada a kódování

Pole ve struktuře MQRFH (včetně *NameValueString*) jsou ve znakové sadě a kódování dané poli *CodedCharSetId* a *Encoding* ve struktuře záhlaví, která předchází MQRFH, nebo těmito poli ve struktuře MQMD, pokud je MQRFH na začátku dat zprávy aplikace.

Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech front.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 513. Pole v MQRFH pro MQRFH

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQRFH_STRUC_ID	'RFH¬'
Verze (číslo verze struktury)	MQRFH_VERSION_1	1
StrucLength (délka struktury MQRFH v bajtech)	MQRFH_STRUC_LENGTH_FIXED	32
Kódování (číselné kódování dat, která následují *NameValueString*)	MQENC_NATIVE	Závisí na prostředí

Tabulka 513. Pole v MQRFH pro MQRFH (pokračování)

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
CodedCharSetId (uvádí identifikátor znakové sady dat, která následují *NameValueString*)	MQCCSI_UNDEFINED	0
Formát (název formátu dat, která následují *NameValueString*)	MQFMT_NONE	Mezery
Příznaky (příznaky)	MQRFH_NONE	0
NameValueString (znakový řetězec proměnné délky obsahující dvojice název-hodnota)	Není	Není

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQRFH_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQRFH MyRFH = {MQRFH_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQRFH

```
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 */
};
```

Deklarace jazyka COBOL pro MQRFH

```
**   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.
```

Deklarace PL/I pro MQRFH

```
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 */
```

Deklarace High Level Assembler pro MQRFH

```
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)
```

Vizuální základní deklarace pro MQRFH

```
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
```

StrucId (MQCHAR4) pro MQRFH

Jedná se o identifikátor struktury pravidel a struktury záhlaví formátování. Vždy se jedná o vstupní pole. Jeho hodnota je MQRFH_STRUC_ID.

Hodnota musí být:

MQRFH_STRUC_ID
Identifikátor pro pravidla a strukturu záhlaví formátování.

Pro programovací jazyk C je definována také konstanta MQRFH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQRFH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQRFH

Toto je číslo verze struktury; hodnota musí být:

MQRFH_VERSION_1
Version-1 pravidla a struktura záhlaví formátování.

Počáteční hodnota tohoto pole je MQRFH_VERSION_1.

StrucLength (MQLONG) pro MQRFH

Jedná se o délku struktury MQRFH v bajtech, včetně pole *NameValueString* na konci struktury. Délka nezahrnuje žádná uživatelská data, která následují za polem *NameValueString* .

Chcete-li se vyhnout problémům při převodu uživatelských dat v některých prostředích, *StrucLength* musí být násobkem čtyř.

Následující konstanta udává délku *pevné* části struktury, tj. délku bez pole *NameValueString* :

MQRFH_STRUC_LENGTH_FIXED
Délka pevné části struktury MQRFH.

Počáteční hodnota tohoto pole je MQRFH_STRUC_LENGTH_FIXED.

Kódování (MQLONG) pro MQRFH

Tato volba určuje číselné kódování dat, která následují za *NameValueString* ; se nevztahuje na numerická data v samotné struktuře MQRFH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG) pro MQRFH

Tato volba určuje identifikátor znakové sady dat, která následují za položkou *NameValueString* ; se nevztahuje na znaková data v samotné struktuře MQRFH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

MQCCSI_INHERIT
Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota MQCCSI_INHERIT vrácena voláním MQGET.

MQCCSI_INHERIT nelze použít, pokud hodnota pole *PutApplType* v MQMD je MQAT_BROKER.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8) pro MQRFH

Tato volba určuje název formátu dat, která následují za položkou *NameValueString*.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *Format* v MQMD.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQRFH

Lze uvést následující:

MQRFH_NONE
Žádné příznaky.

Počáteční hodnota tohoto pole je MQRFH_NONE.

NameValueŘetězec (MQCHARn) pro MQRFH

Jedná se o znakový řetězec proměnné délky obsahující dvojice název-hodnota ve tvaru:

```
name1 value1 name2 value2 name3 value3 ...
```

Každý název nebo hodnota musí být odděleny od sousedního názvu nebo hodnoty jedním nebo více prázdnými znaky; tyto mezery nejsou významné. Název nebo hodnota může obsahovat významné mezery pomocí předpony a přípony názvu nebo hodnoty s dvojitými uvozovkami; všechny znaky mezi otevřenými dvojitými uvozovkami a odpovídajícími uzavřenými dvojitými uvozovkami jsou považovány za významné. V následujícím příkladu je název FAMOUS_WORDSa hodnota je `Hello World`:

```
FAMOUS_WORDS "Hello World"
```

Název nebo hodnota může obsahovat jakékoli jiné znaky než znak null (který se chová jako oddělovač pro *NameValueString*). V zájmu usnadnění interoperability však může aplikace omezit názvy na následující znaky:

- První znak: velká nebo malá písmena abecedy (A až Z nebo a až z) nebo podtržítko.
- Následné znaky: velká nebo malá písmena abecedy, desetinná číslice (0 až 9), podtržítko, pomlčka nebo tečka.

Pokud název nebo hodnota obsahuje jednu nebo více dvojitých uvozovek, název nebo hodnota musí být uzavřeny v dvojitých uvozovkách a každá dvojitá uvozovka v řetězci musí být dvojitá:

```
Famous_Words "The program displayed ""Hello World"""
```

Názvy a hodnoty rozlišují velká a malá písmena, to znamená, že malá písmena nejsou považována za stejná jako velká písmena. Například FAMOUS_WORDS a `Famous_Words` jsou dva různé názvy.

Délka v bajtech *NameValueString* je rovna *StrucLength* minus MQRFH_STRUC_LENGTH_FIXED. Chcete-li se vyhnout problémům při převodu uživatelských dat v některých prostředích, učiňte tuto délku násobkem čtyř. Tuto délku vyplněte znaky *NameValueString* mezerami nebo ji ukončete dříve tak, že za poslední významný znak v řetězci umístíte znak null. Znak null a bajty, které za ním následují, až do uvedené délky *NameValueString*, jsou ignorovány.

Poznámka: Protože délka tohoto pole není pevná, pole je vynecháno z deklarací struktury, které jsou poskytnuty pro podporované programovací jazyky.

MQRFH2 -Pravidla a formátování záhlaví 2

Záhlaví MQRFH2 je založeno na záhlaví MQRFH , ale umožňuje přenos řetězců Unicode bez překladu a může nést číselné datové typy.Struktura MQRFH2 definuje formát pravidel version-2 a záhlaví formátování. Toto záhlaví použijete k odeslání dat, která byla zakódována pomocí syntaxe podobné XML. Zpráva může obsahovat dvě nebo více struktur MQRFH2 v řadě, přičemž uživatelská data mohou volitelně sledovat poslední strukturu MQRFH2 v řadě.

Dostupnost

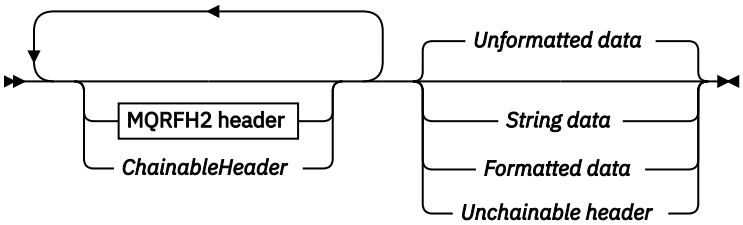
Všechny systémy IBM MQ a IBM MQ MQI clients připojené k těmto systémům.

Název formátu

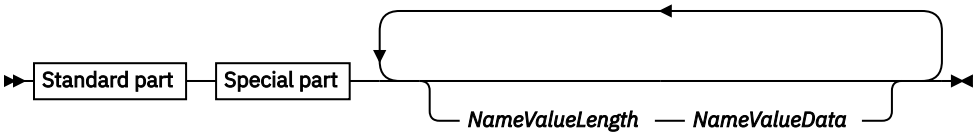
MQFMT_RF_HEADER_2

Syntax

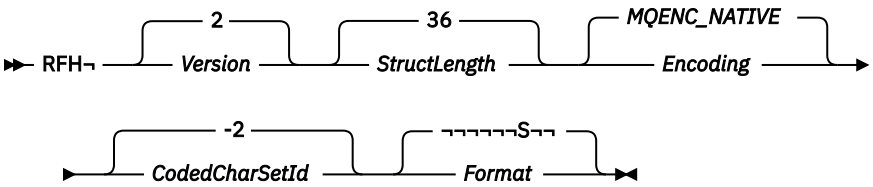
IBM MQ Message



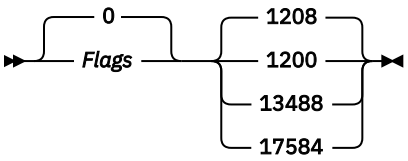
MQRFH2 header



Standard part



Special part



Znaková sada a kódování

Pro znakovou sadu a kódování použité pro strukturu MQRFH2 platí speciální pravidla:

- Jiná pole než *NameValueData* jsou ve znakové sadě a kódování dané poli *CodedCharSetId* a *Encoding* ve struktuře záhlaví, která předchází MQRFH2, nebo těmito poli ve struktuře MQMD , pokud je MQRFH2 na začátku dat zprávy aplikace.

  Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech front.

  Je-li ve volání MQGET zadán parametr MQGMO_CONVERT , správce front převede pole MQRFH2 jiná než *NameValueData*na požadovanou znakovou sadu a kódování.
- *NameValueData* je ve znakové sadě dané polem *NameValueCCSID* . Pouze uvedené znakové sady Unicode jsou platné pro *NameValueCCSID* ; Podrobnosti naleznete v popisu souboru *NameValueCCSID* .

  Některé znakové sady mají reprezentaci, která závisí na kódování. Pokud je *NameValueCCSID* jednou z těchto znakových sad, *NameValueData* musí být ve stejném kódování jako ostatní pole v MQRFH2.

  Je-li ve volání MQGET zadána volba MQGMO_CONVERT , převede správce front *NameValueData* na požadované kódování, ale nezmění svou znakovou sadu.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 514. Pole v MQRFH2 pro MQRFH2

Název pole	Název konstanty	Hodnota konstanty
StrucId (identifikátor struktury)	MQRFH_STRUC_ID	'RFH¬'
Verze (číslo verze struktury)	MQRFH_VERSION_2	2
StrucLength (délka struktury MQRFH2 v bajtech)	MQRFH_STRUC_LENGTH_FIXED_2	36
Kódování (číselné kódování dat, která následují za posledním *NameValueData* polem)	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId (identifikátor znakové sady dat, která následují za posledním polem *NameValueData*)	MQCCSI_INHERIT	-2
Formát (název formátu dat, která následují za posledním *NameValueData* polem)	MQFMT_NONE	Mezery
Příznaky (příznaky)	MQRFH_NONE	0
NameValueCCSID (identifikátor kódované znakové sady dat v poli *NameValueData*)	Není	1208
NameValueDélka (délka dat v bajtech v poli *NameValueData*)	Není	None
NameValueData (dvojice název-hodnota vlastností zprávy)	Není	Není

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQRFH2_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQRFH2 MyRFH2 = {MQRFH2_DEFAULT};
```

Deklarace jazyka

Deklarace jazyka C pro MQRFH2

```
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 */
};
```

Deklarace jazyka COBOL pro MQRFH2

```
**   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.
```

Deklarace PL/I pro MQRFH2

```
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 */
```

Deklarace High Level Assembler pro MQRFH2

```
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)
```

Deklarace jazyka Visual Basic pro MQRFH2

```
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
```

StrucId (MQCHAR4) pro MQRFH2

Jedná se o identifikátor struktury pravidel a formátování záhlaví dvě struktury. Vždy se jedná o vstupní pole. Jeho hodnota je MQRFH2_STRUC_ID.

Hodnota musí být:

MQRFH2_STRUC_ID
Identifikátor pro pravidla a formátování záhlaví dvě struktury.

Pro programovací jazyk C je definována také konstanta MQRFH2_STRUC_ID_ARRAY . Má stejnou hodnotu jako MQRFH2_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQRFH2

Toto je číslo verze struktury; hodnota musí být:

MQRFH_VERSION_2
Version-2 -pravidla a struktura formátování záhlaví.

Počáteční hodnota tohoto pole je MQRFH_VERSION_2.

StrucLength (MQLONG) pro MQRFH2

Jedná se o délku struktury MQRFH2 v bajtech, včetně polí *NameValueLength* a *NameValueData* na konci struktury. Je platné, aby na konci struktury bylo více dvojic polí *NameValueLength* a *NameValueData* v pořadí:

```
length1, data1, length2, data2, ...
```

StrucLength nezahrnuje žádná uživatelská data, která by mohla následovat za posledním *NameValueData* polem na konci struktury.

Chcete-li se vyhnout problémům s převodem uživatelských dat v některých prostředích, *StrucLength* musí být násobkem čtyř.

Následující konstanta udává délku *pevné* části struktury, tj. délku bez polí *NameValueLength* a *NameValueData* :

MQRFH_STRUC_LENGTH_FIXED_2
Délka pevné části struktury MQRFH2 .

Počáteční hodnota tohoto pole je MQRFH_STRUC_LENGTH_FIXED_2.

Kódování (MQLONG) pro MQRFH2

Tato volba určuje číselné kódování dat, která následují za posledním polem *NameValueData* ; nevztahuje se na číselná data v samotné struktuře MQRFH2 .

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG) pro MQRFH2

Určuje identifikátor znakové sady dat, která následují za posledním polem *NameValueData* ; nevztahuje se na znaková data v samotné struktuře MQRFH2 .

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

MQCCSI_INHERIT
Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota MQCCSI_INHERIT vrácena voláním MQGET.

MQCCSI_INHERIT nelze použít, pokud hodnota pole *PutApplType* v MQMD je MQAT_BROKER.

Počáteční hodnota tohoto pole je MQCCSI_INHERIT.

Formát (MQCHAR8) pro MQRFH2

Tato volba určuje název formátu dat, která následují za posledním polem *NameValueData* .

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *Format* v MQMD.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQRFH2

Počáteční hodnota tohoto pole je MQRFH_NONE. MQRFH_NONE musí být zadaný.

MQRFH_NONE
Žádné příznaky.

MQRFH_INTERNAL
Záhlaví MQRFH2 obsahuje interně nastavené vlastnosti.

MQRFH_INTERNAL je určen pro použití správcem front.

Horních 16 bitů, MQRFH_FLAGS_RESTRICTED_MASK, je vyhrazeno pro příznaky sad správců front. Příznaky, které může uživatel nastavit, jsou definovány v posledních 16 bitech.

NameValueCCSID (MQLONG) pro MQRFH2

Uvádí identifikátor kódované znakové sady dat v poli *NameValueData* . Liší se od znakové sady ostatních řetězců ve struktuře MQRFH2 a může se lišit od znakové sady dat (pokud existuje), která následuje za posledním polem *NameValueData* na konci struktury.

NameValueCCSID musí mít jednu z následujících hodnot:

CCSID	**Význam**
1200	UTF-16, nejnovější podporovaná verze Unicode

CCSID	Význam
13488	UTF-16, verze Unicode 2.0 dílčí sada
17584	Podmnožina UTF-16, Unicode verze 3.0 (obsahuje symbol Euro)
1208	UTF-8, nejnovější podporovaná verze Unicode

Pro znakové sady UTF-16 musí být kódování (pořadí bajtů) *NameValueData* stejné jako kódování ostatních polí ve struktuře MQRFH2 .

Znaky mimo základní vícejazyčnou rovinu Unicode (nad znaky U + FFFF) reprezentované v kódu UTF-16 náhradními kódovými body (X'D800'až X'DFFF') nebo čtyřmi bajty v kódování UTF-8nejsou podporovány.

Poznámka: Pokud produkt *NameValueCCSID* nemá žádnou z výše uvedených hodnot a struktura MQRFH2 vyžaduje převod volání MQGET, volání se dokončí s kódem příčiny MQRC_SOURCE_CCSID_ERROR a zpráva se vrátí nepřevedená.

Počáteční hodnota tohoto pole je 1208.

NameValueDélka (MQLONG) pro MQRFH2

Délka odpovídajícího pole NameValueData

Určuje délku dat v bajtech v poli *NameValueData* . *NameValueLength* musí být násobkem čtyř.

Poznámka: Pole *NameValueLength* a *NameValueData* jsou volitelná, ale pokud jsou přítomna, musí se vyskytovat jako dvojice a sousedící. Dvojici polí lze opakovat tolikrát, kolikrát je třeba, například:

```
length1 data1 length2 data2 length3 data3
```

Protože jsou tato pole volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.

NameValueData (MQCHARn) pro MQRFH2

NameValueData je pole s proměnnou délkou, které obsahuje složku obsahující dvojice název-hodnota vlastností zprávy. Složka je znakový řetězec proměnné délky obsahující data kódovaná pomocí syntaxe podobné XML. Délka znakového řetězce v bajtech je dána polem *NameValueLength* , které předchází poli *NameValueData* . Délka musí být násobkem čtyř.

Pole *NameValueLength* a *NameValueData* jsou volitelná, ale pokud jsou přítomna, musí se vyskytovat jako dvojice a sousedící. Dvojici polí lze opakovat tolikrát, kolikrát je třeba, například:

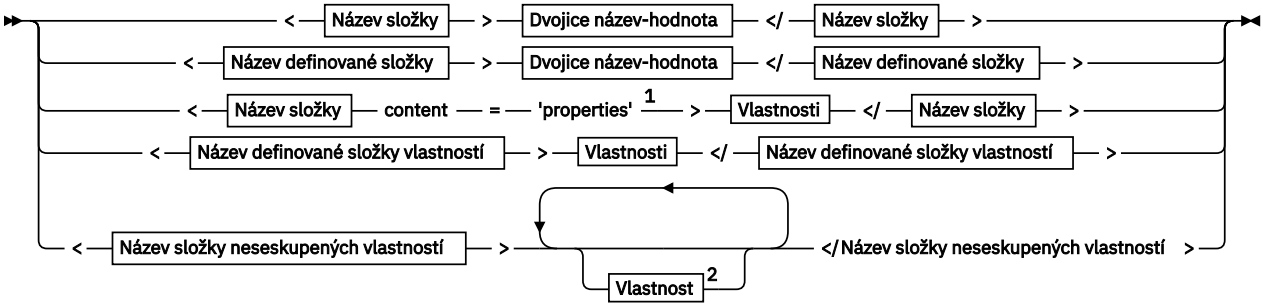
```
length1 data1 length2 data2 length3 data3
```

NameValueData není převedeno na znakovou sadu uvedenou ve volání MQGET . I když je zpráva načtena s volbou MQGMO_CONVERT v platnosti *NameValueData* , zůstane v původní znakové sadě. Hodnota *NameValueData* se však převede na kódování zadané ve volání MQGET .

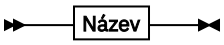
Notes:

- Protože jsou tato pole volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.
- Termíny "definované" a "vyhrazené" se používají v syntaktickém diagramu. "Definováno" znamená, že název používá IBM MQ. "Vyhrazeno" znamená, že název je vyhrazen pro budoucí použití produktem IBM MQ.

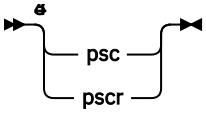
NameValueData syntaxe



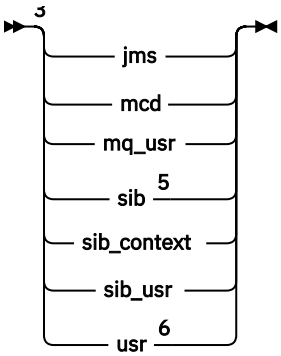
Název složky



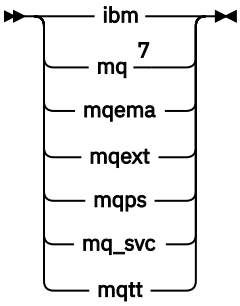
Název definované složky



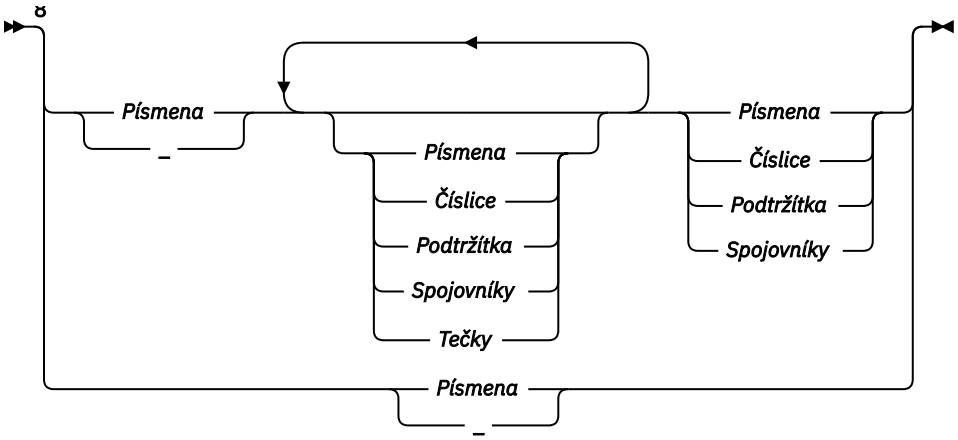
Název definované složky vlastností



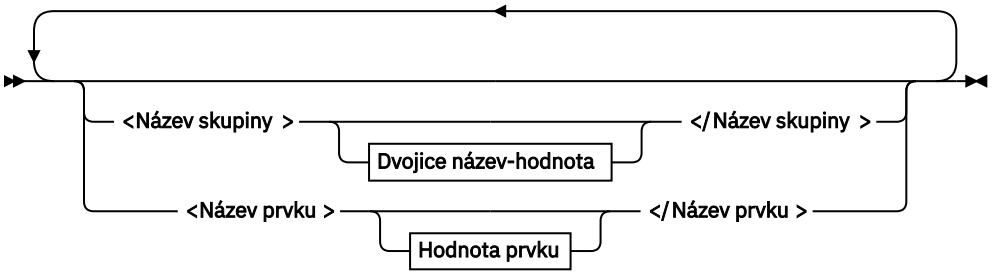
Název složky neseskupených vlastností



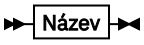
Název



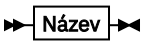
Dvojice název-hodnota



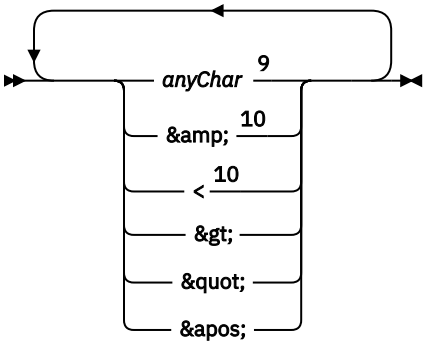
Název skupiny



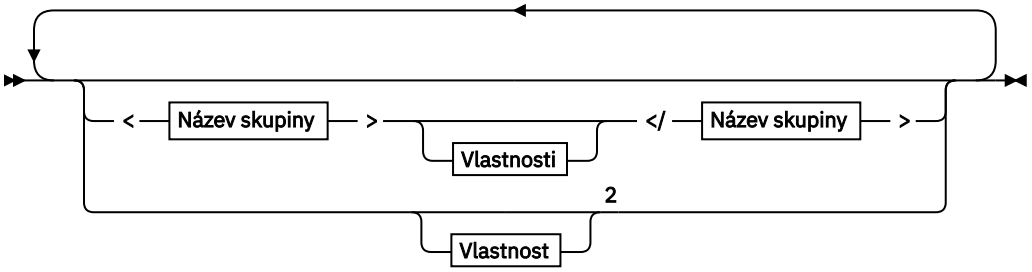
Název prvku



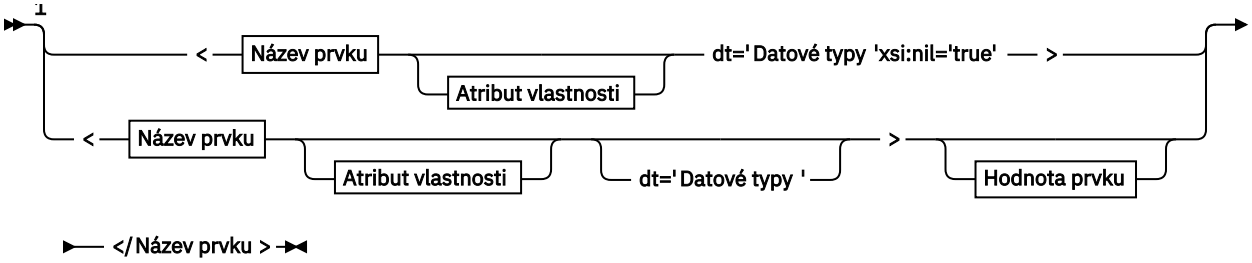
Hodnota prvku



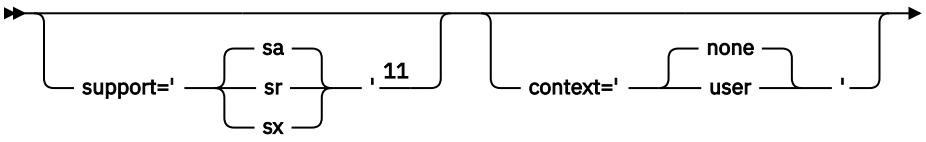
Vlastnosti

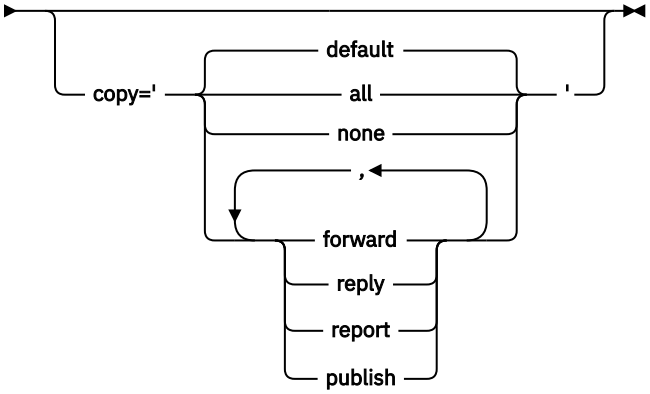


Vlastnost

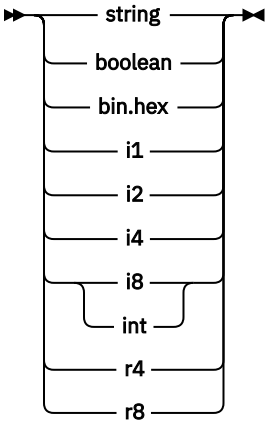


Atribut vlastnosti





Datové typy



Poznámky:

[1] Dvojité nebo jednoduché uvozovky jsou platné.

[2] Nepoužívejte neplatný název vlastnosti; viz “Neplatný název vlastnosti” na stránce 549. Název vyhrazené vlastnosti použijte pouze pro její definovaný účel; viz “Definované názvy vlastností” na stránce 549.

[3] Název musí být uveden malými písmeny.

[4] Podporována je pouze jedna složka `psc` a `pscr` .

[5] WebSphere Application Server Služba Integration Bus ignoruje složky sib, sib_contexta sib_usr v následných záhlavích `MQRFH2` a významné jsou pouze vlastnosti v prvním záhlaví `MQRFH2` .

[6] V souboru `MQRFH2`nesmí být přítomna více než jedna složka `usr` . Vlastnosti ve složce `usr` se nesmí vyskytovat více než jednou.

[7] Důležité jsou pouze vlastnosti v první složce mq . Je-li složka UTF-8, jsou podporovány pouze jednobajtové znaky UTF-8 . Jediný znak mezery je Unicode U+0020.
[8] Platné znaky jsou definovány ve specifikaci XML W3C a skládají se v podstatě z kategorií Unicode Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm, a Nd ; viz Kategorie znaků Unicode.
[9] Všechny znaky jsou významné. Počáteční a koncové mezery jsou součástí hodnoty prvku.
[10] Nepoužívejte neplatný znak; viz “Neplatné znaky” na stránce 549. Místo těchto neplatných znaků použijte řídicí posloupnost.
[11] Atribut vlastnosti podpory je platný pouze ve složce mq .

Název složky

NameValueData obsahuje jednu složku. Chcete-li vytvořit více složek, vytvořte více polí *NameValueData* . Můžete vytvořit více polí *NameValueData* v jednom záhlaví MQRFH2 v rámci zprávy. Případně můžete vytvořit více zřetězených záhlaví MQRFH2 , z nichž každé obsahuje více polí *NameValueData* .

Pořadí záhlaví MQRFH2 a pořadí polí *NameValueData* se nijak neliší od logického obsahu složky. Pokud je stejná složka ve zprávě přítomna více než jednou, je složka analyzována jako celek. Pokud se stejná vlastnost vyskytuje v více instancích stejné složky, je analyzována jako seznam.

Správná analýza MQRFH2 není ovlivněna alternativními způsoby, jak může být složka fyzicky uložena ve zprávě.

Čtyři složky toto pravidlo nedodržují. Analyzuje se pouze první instance složky mq, sib, sib_contexta sib_usr .

Pokud se stejná vlastnost vyskytuje více než jednou v kombinovaném obsahu zřetězených záhlaví MQRFH2 , je analyzována pouze první instance vlastnosti. Pokud je vlastnost nastavena pomocí volání rozhraní API, jako např. MQSETMP, a je přidána do produktu MQRFH2 přímo aplikací, má volání rozhraní API přednost.

Název složky je název složky obsahující dvojice název-hodnota nebo skupiny. Skupiny a dvojice název-hodnota mohou být smíchány na stejné úrovni ve stromu složek; viz Obrázek 1 na stránce 539. Nekombinujte název skupiny a název prvku; viz Obrázek 2 na stránce 539

```
<group1><nvp1>value</nvp1></group1><group2><nvp2>value</nvp2></group2>
<group3><nvp1>value</nvp1></group3><nvp3>value</nvp3>
```

Obrázek 1. Správné použití skupin a dvojic název-hodnota

```
<group1><nvp1> value </nvp1> value </group1>
```

Obrázek 2. Nesprávné použití skupin a dvojic název-hodnota

Nepoužívejte neplatný nebo vyhrazený název složky; viz “Neplatný název cesty” na stránce 549 a “Název složky vyhrazené složky nebo složky vlastností” na stránce 548. Definovaný název složky použijte pouze pro svůj definovaný účel; viz “Název definované složky” na stránce 540.

Pokud přidáte atribut 'content=properties' do značky názvu složky, složka se stane složkou vlastností; viz Obrázek 3 na stránce 540.

```
<myFolder></myfolder>
<myPropertyFolder contents='properties'></myPropertyFolder>
```

Obrázek 3. Příklad složky a složky vlastností

V názvech složek se rozlišují malá a velká písmena. Názvy složek a názvy složek vlastností sdílejí stejný obor názvů. Musí mít různá jména. `Folder1` v Obrázek 4 na stránce 540 musí být jiný název než `Folder2` v Obrázek 5 na stránce 540.

```
< Folder1 ><NVP1> value </NVP1></ Folder1 >
```

Obrázek 4. `Folder1` prostor jmen

```
< Folder2 content='properties'>< Property1 > value </ Property1 ></ Folder2 >
```

Obrázek 5. `Folder2` prostor jmen

Skupiny, vlastnosti a dvojice název-hodnota v různých složkách mají různé obory názvů. `Property1` in Obrázek 5 na stránce 540 je jiná vlastnost než `Property1` in Obrázek 6 na stránce 540.

```
<Folder3 content='properties'>< Property1 > value </ Property1 ></Folder3>
```

Obrázek 6. `Folder3` prostor jmen

Složky vlastností se liší od složek nevlastnických ve dvou důležitých ohledech:

1. Složky vlastností obsahují vlastnosti a složky bez vlastností obsahují dvojice název-hodnota. Složky se mírně liší, syntakticky.
2. K přístupu k vlastnostem zprávy použijte definovaná rozhraní, například vlastnosti MQI nebo vlastnosti zprávy JMS . Rozhraní zajišťují, že složky vlastností v souboru `MQRFH2` jsou dobře formované. Správně utvořená složka vlastností je interoperabilní mezi správci front na různých platformách a různých verzích.

Vlastnost zprávy MQI je robustní způsob, jak číst a psát `MQRFH2`, a vyhýbá se potížím se správnou syntaktickou analýzou `MQRFH2` .

Název definované složky

Definovaný název složky je název složky, která je vyhrazena pro použití produktem IBM MQnebo jiným produktem. Nevytvářejte složku se stejným názvem a nepřidávejte do složek vlastní dvojice název-hodnota. Definované složky jsou `psc` a `pscr`.

`psc` a `pscr` jsou používány ve frontě publikování/odběru.

Segmentovaná zpráva vložená s `MQMF_SEGMENT` nebo `MQMF_SEGMENTATION_ALLOWED` nemůže obsahovat `MQRFH2` s definovaným názvem složky. `MQPUT` selže s kódem příčiny 2443, `MQRC_SEGMENTATION_NOT_ALLOWED`.

Název definované složky vlastností

Definovaný název složky vlastností je název složky vlastností, kterou používá produkt IBM MQnebo jiný produkt. Názvy složek a jejich obsah viz Složky vlastností. Definované názvy složek vlastností jsou podmnožinou všech názvů složek vyhrazených pro IBM MQ; viz “Název složky vyhrazené složky nebo složky vlastností” na stránce 548.

Libovolný prvek uložený v definované složce vlastností je vlastností. Prvek uložený v definované složce vlastností nesmí mít atribut `content='properties'` .

Vlastnosti můžete přidávat pouze do definovaných složek vlastností `usr`, `mq_usr`a `sib_usr`. V jiných složkách vlastností, jako např. `mq` a `sib`, IBM MQ ignoruje nebo zahodí vlastnosti, které nerozpozná.

Popis každé definované složky vlastností uvádí vlastnosti, které produkt IBM MQ definoval a které mohou používat aplikační programy. K některým vlastnostem se přistupuje nepřímo nastavením nebo získáním vlastnosti JMS a k některým z nich se přistupuje přímo pomocí volání `MQSETMP` a `MQINQMP` MQI.

Definované složky vlastností také obsahují další vlastnosti, které IBM MQ má vyhrazené, ale k nimž aplikace nemají přístup. Názvy vyhrazených vlastností nejsou uvedeny. Ve složkách vlastností `usr`, `mq_usr`a `sib_usr` nejsou žádné vyhrazené vlastnosti. Nevytvářejte však vlastnosti s neplatnými názvy vlastností; viz “Neplatný název vlastnosti” na stránce 549.

Složky vlastností

jms

Produkt `jms` obsahuje pole záhlaví JMS a vlastnosti JMSX, které nelze plně vyjádřit v produktu `MQMD`. Složka `jms` je vždy přítomna v prostředí JMS `MQRFH2`.

Tabulka 515. název vlastnosti jms, synonymum, datový typ a složka

Synonymum vlastnosti	Název vlastnosti	Datový typ	Složka
JMSDestination	jms.Dst	string	<jms><Dst> *destination* </Dst></jms>
JMSExpiration	jms.Exp	i8	<jms><Exp> *expiration* </Exp></jms>
KorelaceJMSCorrelation	jms.Cid	string	<jms><Cid> *correlationId* </Cid></jms>
JMSDelivery	jms.Dlv	i4	<jms><Dlv> *delivery* </Dlv></jms>
JMSPriority.	jms.Pri	i4	<jms><Pri> *priority* </Pri></jms>
JMSReplyTo	jms.Rto	string	<jms><Rto> *replyToURI* </Rto></jms>
JMSTimeStamp	jms.Tms	i8	<jms><Tms> *timestamp* </Tms></jms>
JMSXGroupID	jms.Gid	string	<jms><Gid> *groupId* </Gid></jms>
JMSXGroupSeq	jms.Seq	i4	<jms><Seq> *messageSequenceNo* </Seq></jms>

Nepřidávejte své vlastní vlastnosti do složky `jms`.

mcd

mcd obsahuje vlastnosti, které popisují formát zprávy. Například vlastnost Msd domény služby zpráv identifikuje zprávu jako typu JMSTextMessage, JMSBytesMessage, JMSStreamMessage, JMSMapMessage, JMSObjectMessage nebo null.

Složka mcd je vždy přítomna ve zprávě JMS obsahující MQRFH2.

Vždy se vyskytuje ve zprávě obsahující MQRFH2 odeslané z IBM Integration Bus. Popisuje doménu, formát, typ a sadu zpráv příslušné zprávy.

Tabulka 516. mcd - název vlastnosti, synonymum, datový typ a složka

Synonymum vlastnosti	Název vlastnosti	Datový typ	Složka
	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>

Nepřidávejte své vlastní vlastnosti do složky mcd.

mq_usr

mq_usr obsahuje vlastnosti definované aplikací, které nejsou vystaveny jako JMS vlastnosti definované uživatelem. Do této složky lze umístit vlastnosti, které nesplňují požadavky produktu JMS .

Vlastnosti můžete vytvořit ve složce mq_usr . Vlastnosti, které vytvoříte v souboru mq_usr , jsou podobné vlastnostem, které vytvoříte v nových složkách s atributem content='properties' .

sib

sib obsahuje WebSphere Application Server vlastnosti systémové zprávy sběrnice pro integraci služeb (WAS/SIB). Vlastnosti sib nejsou vystaveny jako JMS vlastnosti pro aplikace IBM MQ JMS , protože nejsou podporovaného typu. Například některé vlastnosti sib nelze vystavit jako vlastnosti JMS , protože se jedná o bajtová pole. Některé vlastnosti sib jsou vystaveny aplikacím WAS/SIB jako vlastnosti JMS_IBM_* ; patří mezi ně vlastnosti cest dopředného a zpětného směrování.

Nepřidávejte své vlastní vlastnosti do složky sib.

sib_context

sib_context obsahuje vlastnosti systémové zprávy WAS/SIB, které nejsou vystaveny uživatelským aplikacím WAS/SIB nebo jako vlastnosti JMS . sib_context obsahuje vlastnosti zabezpečení a transakčních vlastností, které se používají pro webové služby.

Nepřidávejte své vlastní vlastnosti do složky sib_context.

sib_usr

sib_usr obsahuje vlastnosti zprávy uživatele WAS/SIB, které nejsou vystaveny jako JMS vlastnosti uživatele, protože nejsou podporovaného typu. Produkt sib_usr je vystaven aplikacím WAS/SIB v rozhraní SIMessage ; viz Vývoj integrace služeb.

Typ vlastnosti sib_usr musí být bin.hexa hodnota musí být ve správném formátu. Pokud aplikace IBM MQ zapíše prvek typu bin.hex do složky v nesprávném formátu, obdrží aplikace prvek IOException. Pokud datový typ vlastnosti není bin.hex , aplikace obdrží ClassCastException.

Nepokoušejte se zpřístupnit uživatelské vlastnosti produktu JMS pro soubor WAS/SIB pomocí této složky; místo toho použijte složku usr .

Vlastnosti můžete vytvořit ve složce `sib_usr` .

usr

`usr` obsahuje vlastnosti JMS definované aplikací přidružené ke zprávě. Složka `usr` je k dispozici pouze v případě, že aplikace nastavila vlastnost definovanou aplikací.

`usr` je výchozí složka vlastností. Je-li vlastnost nastavena bez názvu složky, je umístěna do složky `usr` .

Tabulka 517. název vlastnosti usr, synonymum, datový typ a složka

Synonymum vlastnosti	Název vlastnosti	Datový typ	Složka
	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>

Vlastnosti můžete vytvořit ve složce `usr` .

Segmentovaná zpráva vložená s hodnotou `MQMF_SEGMENT` nebo `MQMF_SEGMENTATION_ALLOWED` nemůže obsahovat `MQRFH2` s definovaným názvem složky vlastností. `MQPUT` selže s kódem příčiny 2443, `MQRC_SEGMENTATION_NOT_ALLOWED`.

Název složky neseskupených vlastností

ibm

`ibm` obsahuje vlastnosti, které používá pouze IBM MQ.

Tabulka 518. `ibm` - název vlastnosti, synonymum, datový typ a složka

Synonymum vlastnosti	Název vlastnosti	Datový typ	Složka
	ibm.rfp	string	<ibm><rfp>fingerprint</rfp></ibm>

Nepřidávejte své vlastní vlastnosti do složky `ibm`.

mq

`mq` obsahuje vlastnosti, které používá pouze IBM MQ.

Na vlastnosti ve složce `mq` se vztahují následující omezení:

- Pouze vlastnosti v první významné složce `mq` ve zprávě jsou produktem MQovlivněny; vlastnosti v jakékoli jiné složce `mq` ve zprávě jsou ignorovány.
- Ve složce jsou povoleny pouze jednobajtové znaky UTF-8 . Vícebajtový znak ve složce může způsobit selhání syntaktické analýzy a odmítnutí zprávy.

- Ve složce nepoužívejte únikové řetězce. Řídicí řetězec je považován za skutečnou hodnotu prvku.
- Pouze znak Unicode U+0020 je ve složce považován za mezeru. Všechny ostatní znaky jsou považovány za významné a mohou způsobit selhání syntaktické analýzy složky a odmítnutí zprávy.

Pokud se syntaktická analýza složky mq nezdaří nebo pokud složka nedodržuje tato omezení, bude zpráva odmítnuta s kódem příčiny 2527, MQRC_RFH_RESTRICTED_FORMAT_ERR.

Nepřidávejte své vlastní vlastnosti do složky mq.

mqema

mqema obsahuje vlastnosti, které používá pouze WebSphere Application Server. Složka byla nahrazena složkou mqext.

Nepřidávejte své vlastní vlastnosti do složky mqema.

mqext

Produkt mqext obsahuje následující typy vlastností:

- Vlastnosti, které používá pouze WebSphere Application Server.
- Vlastnosti související se zpožděným doručováním zpráv.

Složka je přítomna, pokud má aplikace nastavenu minimálně jednu definovanou vlastnost IBM nebo používá prodlevu doručení.

Tabulka 519. mqext - název vlastnosti, synonymum, datový typ a složka

Synonymum vlastnosti	Název vlastnosti	Datový typ	Složka
JMSArmCorrelator	mqext.Arm	string	<mqext><Arm>*armCorrelator*</Arm></mqext>
JMSRMCorrelator	mqext.Wrm	string	<mqext><Wrm>*wrmCorrelator*</Wrm></mqext>
JMSDeliveryTime	mqext.Dlt	i8	<mqext><Dlt>*DeliveryTime*</Dlt></mqext>
JMSDeliveryDelay	mqext.Dly	i8	<mqext><Dly>*DeliveryTime*</Dly></mqext>

Nepřidávejte své vlastní vlastnosti do složky mqext.

mqps

Produkt mqps obsahuje vlastnosti, které jsou používány pouze v rámci publikování/odběru produktu IBM MQ. Tato složka je přítomna pouze, když má aplikace nastavenu minimálně jednu z integrovaných vlastností publikování/odběru.

Tabulka 520. mqps - název vlastnosti, synonymum, datový typ a složka

Synonymum vlastnosti	Název vlastnosti	Datový typ	Složka
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>

Tabulka 520. mqps - název vlastnosti, synonymum, datový typ a složka (pokračování)			
Synonymum vlastnosti	**Název vlastnosti**	**Datový typ**	**Složka**
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>

Nepřidávejte své vlastní vlastnosti do složky mqps.

mq_svc

mq_svc obsahuje vlastnosti používané SupportPac MA93.

Nepřidávejte své vlastní vlastnosti do složky mq_svc.

mqtt

mqtt obsahuje vlastnosti používané produktem MQ Telemetry

Tabulka 521. Název vlastnosti mqtt, synonymum, datový typ a složka			
Synonymum vlastnosti	**Název vlastnosti**	**Datový typ**	**Složka**
	mqtt.clientId	string	<mqtt><clientId> *topicString* </clientId></mqtt>
	mqtt.qos	i4	<mqtt><qos> *qualityOfService* </qos></mqtt>
	mqtt.msgid	string	<mqtt><msgid> *messageIdentifier* </msgid></mqtt>

Nepřidávejte své vlastní vlastnosti do složky mqtt.

Segmentovaná zpráva vložená s hodnotou MQMF_SEGMENT nebo MQMF_SEGMENTATION_ALLOWED nemůže obsahovat MQRFH2 s neseskupeným názvem složky vlastností. MQPUT selže s kódem příčiny 2443, MQRC_SEGMENTATION_NOT_ALLOWED.

Dvojice název-hodnota

V syntaktickém diagramu "Dvojice název-hodnota" popisuje obsah běžné složky. Běžná složka obsahuje skupiny a prvky. Prvek je dvojice název-hodnota. Skupina obsahuje prvky a další skupiny.

Pokud jde o stromy, prvky jsou koncové uzly a skupiny jsou interní uzly. Interní uzel a složka, která je kořenovým uzlem, mohou obsahovat kombinaci interních uzlů a koncových uzlů. Uzel nemůže být současně vnitřním i koncovým uzlem; viz Obrázek 2 na stránce 539.

Vlastnosti

V syntaktickém diagramu "Vlastnosti" popisuje obsah složky vlastností. Složka vlastností obsahuje skupiny a vlastnosti. Vlastnost je dvojice název-hodnota s volitelným atributem datového typu. Skupina obsahuje vlastnosti a další skupiny.

Pokud jde o stromy, vlastnosti jsou koncové uzly a skupiny jsou interní uzly. Interní uzel a složka vlastností, která je kořenovým uzlem, mohou obsahovat kombinaci interních uzlů a koncových uzlů. Uzel nemůže být současně vnitřním i koncovým uzlem; viz Obrázek 2 na stránce 539.

Vlastnost

Vlastnost zprávy je dvojice název-hodnota ve složce vlastností. Volitelně může obsahovat atribut datového typu a atribut vlastnosti; příklad viz následující kód. Pokud je atribut datového typu vynechán, typ vlastnosti je `string`.

```
<pf><p1 dt='i8' > value </p1></pf>
```

Název vlastnosti zprávy je úplný název cesty, se syntaxí XML, <> , nahrazenou tečkami. Například `myPropertyFolder1.myGroup1.myGroup2.myProperty1` je mapován na řetězec *NameValueData* následujícím způsobem. Řetězec je formátován pro snadnější čtení.

```
<myPropertyFolder1>
    <myGroup1>
        <myGroup2>
            <myProperty1>value</myProperty1>
        </myGroup2>
    </myGroup1>
</myPropertyFolder1>
```

Složka vlastností může obsahovat více vlastností. Například vlastnosti v souboru Obrázek 7 na stránce 546 jsou mapovány na složku vlastností v části Změny pro izolaci prodejní fronty v novém klastru a oddělení přenosových front klastru brány.

```
 myPropertyFolder1.myProperty4
myPropertyFolder1.myGroup1.myGroup2.myProperty1
myPropertyFolder1.myGroup1.myGroup2.myProperty2
myPropertyFolder1.myGroup1.myProperty3
```

Obrázek 7. Více vlastností se stejným kořenovým názvem

```
<myPropertyFolder1>
    <myProperty4>value</myProperty4>
    <myGroup1>
        <myGroup2>
            <myProperty1>value</myProperty1>
            <myProperty2>value</myProperty2>
        </myGroup2>
        <myProperty3>value</myProperty3>
    </myGroup1>
</myPropertyFolder1>
```

Obrázek 8. Mapování názvu více vlastností

Název

Název musí začínat *písmenem* nebo *podtržítkem*. Nesmí obsahovat *Dvojtečku*, nesmí končit *tečkou* a obsahovat pouze *písmena*, *číslice*, *podtržítka*, *spojovníky* a *tečky*. Platné znaky jsou definovány ve specifikaci XML W3C a skládají se v podstatě z kategorií Unicode `Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm,` a `Nd` ; viz Kategorie znaků Unicode.

Úplná cesta k vlastnosti nebo dvojici název-hodnota nesmí porušit pravidlo popsané v části "Neplatný název cesty" na stránce 549. Cesty jsou omezeny na 4095 bajtů, nesmí obsahovat znaky kompatibility Unicode a nesmí začínat řetězcem XML.

Název skupiny

Název skupiny má stejnou syntaxi jako název. Názvy skupin jsou volitelné. Vlastnosti a dvojice název-hodnota mohou být umístěny v kořenovém adresáři složky. Použijte skupiny, pokud pomáhají uspořádat vlastnosti a dvojice název-hodnota.

Název prvku

Název prvku má stejnou syntaxi jako název.

Hodnota prvku

Hodnota prvku zahrnuje všechny mezery mezi značkou < *Element name* > a značkou < */Element name* >. V hodnotě nepoužívejte dva znaky < a & . Poté nahraďte < a `&`.

Atributy vlastností

Pole deskriptoru vlastností mapování atributů vlastností: Mapování jsou následující:

Podpora

sa (výchozí)
`MQPD_SUPPORT_OPTIONAL`

sr
`MQPD_SUPPORT_REQUIRED`

sx
`MQPD_SUPPORT_REQUIRED_IF_LOCAL`

Kontext

žádný (výchozí)
`MQPD_NO_CONTEXT`

uživatel
`MQPD_USER _CONTEXT`

CopyOptions

objekt forward
`MQPD_COPY_FORWARD`

reply
`MQPD_COPY_REPLY`

sestava
`MQPD_COPY_REPORT`

publikovat
`MQPD_COPY_PUBLISH`

vše
`MQPD_COPY_ALL`

Nepoužívejte `all` v kombinaci s dalšími volbami.

default
`MQPD_COPY_DEFAULT`

Nepoužívejte volbu `default` v kombinaci s jinými volbami. `default` je stejné jako `forward` + `report` + `publish`.

Není
`MQPD_COPY_NONE`

Nepoužívejte volbu `none` v kombinaci s jinými volbami.

Atributy vlastnosti Podpora lze použít pouze pro vlastnosti ve složce `mq` .

Atributy vlastností Context a CopyOptions jsou použitelné pro všechny složky vlastností.

Datové typy

Datové typy MQRFH2 se mapuje na typy vlastností zprávy takto:

Tabulka 522. mapování datových typů

MQRFH2 datový typ	**Typ vlastnosti zprávy**
bin.hex	MQBYTE[]
boolean	MQBOOL
i1	MQINT8
i2	MQINT16
i4	MQINT32
i8	MQINT64
r4	MQFLOAT32
r8	MQFLOAT64
string	MQCHAR[]

Předpokládá se, že jakýkoli prvek bez datového typu je typu `string`.

Atribut prvku `xsi:nil='true'` označuje hodnotu null. Nepoužívejte atribut `xsi:nil='false'` pro nenulové hodnoty. Například následující vlastnost má hodnotu null:

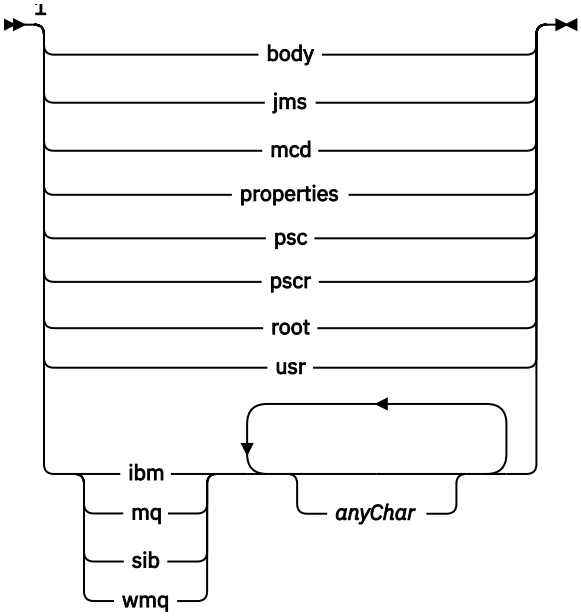
```
<NullProperty
xsi:nil='true'></NullProperty>
```

Vlastnost bajtového nebo znakového řetězce může mít prázdnou hodnotu. Prázdná hodnota je reprezentována prvkem MQRFH2 s hodnotou prvku s nulovou délkou. Například následující vlastnost má prázdnou hodnotu:

```
<EmptyProperty></EmptyProperty>
```

Název složky vyhrazené složky nebo složky vlastností

Omezit název složky nebo složky vlastností tak, aby nezačínaal žádným z následujících řetězců. Předpony jsou vyhrazeny pro názvy složek nebo vlastností vytvořené produktem IBM.

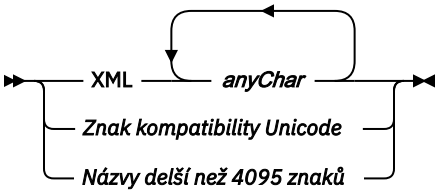


Poznámky:

[1] Vyhrazený název složky nebo vlastnosti obsahuje libovolnou kombinaci malých a velkých písmen.

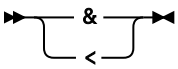
Neplatný název cesty

Omezte úplnou cestu dvojice název-hodnota nebo vlastnost tak, aby neobsahovala žádný z následujících řetězců.



Neplatné znaky

Vždy použijte řídicí posloupnosti & a < místo literálů "&" a "<".

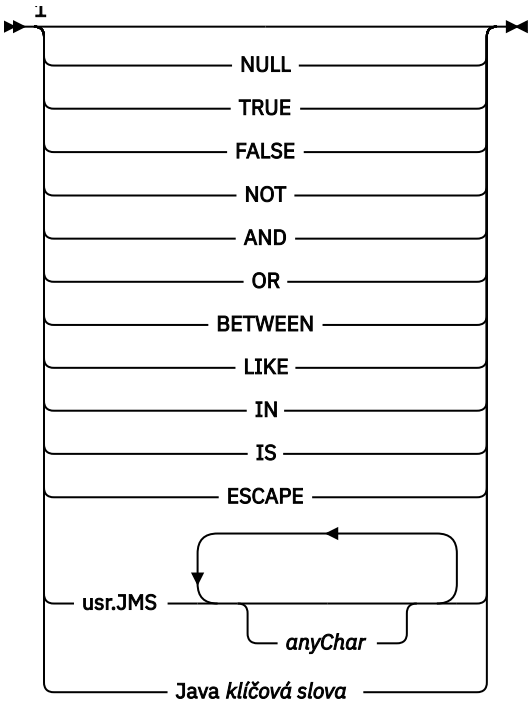


Definované názvy vlastností

Definované názvy vlastností jsou názvy vlastností, které jsou definovány produktem IBM MQnebo jinými produkty a používány produktem IBM MQ a uživatelskými aplikacemi. Definované vlastnosti existují pouze v definovaných složkách vlastností. Definované názvy vlastností jsou popsány v popisu složek vlastností; viz Složky vlastností.

Neplatný název vlastnosti

Nekonstruujte názvy vlastností, které odpovídají následujícímu pravidlu. Pravidlo platí pro úplnou cestu k vlastnosti, která pojmenovává vlastnost, a nikoli pouze pro název prvku vlastnosti.



Poznámky:

[1] Neplatný název vlastnosti může obsahovat libovolnou kombinaci velkých a malých písmen.

Neplatné atributy

Složky a vlastnosti vlastností mohou obsahovat pouze podporované položky “Atributy vlastností” na stránce 547 a “Datové typy” na stránce 548.

Všechny nepodporované atributy typu XML, například názvy s řetězcovými hodnotami v uvozovkách, které jsou obsaženy ve složkách vlastností nebo vlastnostech, mohou být odebrány.

Atributy podobné XML obsažené ve složkách bez vlastností nebo v prvcích bez vlastností, které zůstávají v záhlavích MQRFH2 .

MQRMH-Záhlaví referenční zprávy

Struktura MQRMH definuje formát záhlaví referenční zprávy. Toto záhlaví se používá u uživatelských procedur kanálu zpráv napsaných uživatelem k odesílání extrémně velkého množství dat (nazývaných *hromadná data*). z jednoho správce front do jiného. Rozdíl ve srovnání s normálním systémem zpráv spočívá v tom, že hromadná data nejsou uložena ve frontě; místo toho je ve frontě uložen pouze *odkaz* na hromadná data. To snižuje možnost vyčerpání prostředků IBM MQ malým počtem extrémně velkých zpráv.

Dostupnost

Struktura MQRMH je k dispozici na následujících platformách:

- AIX
- IBM i
- Linux
- Windows

a pro klienty IBM MQ připojené k těmto systémům.

Název formátu

MQFMT_REF_MSG_HEADER

Znaková sada a kódování

Znaková data v MQRMH a řetězce adresované poli offsetu musí být ve znakové sadě lokálního správce front; to je dáno atributem správce front **CodedCharSetId** . Číselná data v MQRMH musí být v nativním kódování počítače; toto je dáno hodnotou MQENC_NATIVE pro programovací jazyk C.

Nastavte znakovou sadu a kódování MQRMH do polí *CodedCharSetId* a *Encoding* v:

- MQMD (pokud je struktura MQRMH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQRMH (všechny ostatní případy).

Použití

Aplikace vloží zprávu sestávající z MQRMH, ale vynechá hromadná data. Když agent kanálu zpráv (MCA) čte zprávu z přenosové fronty, je vyvolána uživatelská procedura pro zpracování záhlaví referenční zprávy. Uživatelská procedura může k referenční zprávě připojit hromadná data identifikovaná strukturou MQRMH před tím, než agent MCA odešle zprávu prostřednictvím kanálu dalšímu správci front.

Na přijímacím konci musí existovat uživatelská procedura pro zprávy, která čeká na referenční zprávy. Při přijetí referenční zprávy musí uživatelská procedura vytvořit objekt z hromadných dat, která následují za MQRMH ve zprávě, a poté předat referenční zprávu bez hromadných dat. Referenční zprávu může později načíst aplikace, která načte referenční zprávu (bez hromadných dat) z fronty.

Za normálních okolností je struktura MQRMH vše, co je ve zprávě. Pokud se však zpráva nachází v přenosové frontě, jedno nebo více dalších záhlaví předchází struktuře MQRMH.

Referenční zprávu lze také odeslat do rozdělovníku. V tomto případě struktura MQDH a související záznamy předcházejí struktuře MQRMH, když je zpráva v přenosové frontě.

Poznámka: Neposílejte referenční zprávu jako segmentovanou zprávu, protože uživatelská procedura zprávy ji nemůže správně zpracovat.

Převod dat

Pro účely převodu dat zahrnuje převod struktury MQRMH převod dat zdrojového prostředí, název zdrojového objektu, data cílového prostředí a název cílového objektu. Jakékoli jiné bajty v rozsahu *StrucLength* bajtů od začátku struktury jsou buď vyřazeny, nebo mají nedefinované hodnoty po převodu dat. Hromadná data se převedou za předpokladu, že všechny následující příkazy jsou pravdivé:

- Hromadná data jsou přítomna ve zprávě, když se provádí převod dat.
- Pole *Format* v MQRMH má jinou hodnotu než MQFMT_NONE.
- Uživatelská procedura pro převod dat zapsaná uživatelem existuje se zadaným názvem formátu.

Mějte však na paměti, že hromadná data nejsou ve zprávě obvykle přítomna, když je zpráva ve frontě, a že v důsledku toho jsou hromadná data převedena pomocí volby MQGMO_CONVERT.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 523. Pole v MQRMH pro MQRMH

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQRMH_STRUC_ID	'RMH¬'

Tabulka 523. Pole v MQRMH pro MQRMH (pokračování)

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
Verze (číslo verze struktury)	MQRMH_VERSION_1	1
StrucLength (celková délka MQRMH, včetně řetězců na konci pevných polí, ale ne hromadných dat)	Není	0
Kódování (číselné kódování hromadných dat)	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId (identifikátor znakové sady hromadných dat)	MQCCSI_UNDEFINED	0
Formát (název formátu hromadných dat)	MQFMT_NONE	Mezery
Příznaky (příznaky referenční zprávy)	MQRMHF_NOT_LAST	0
ObjectType (typ objektu)	Není	Mezery
ObjectInstanceId (identifikátor instance objektu)	MQOII_NONE	Hodnoty null
SrcEnvSrcEnv (délka zdrojových dat prostředí)	Není	0
SrcEnvOffset (posun zdrojových dat prostředí)	Není	0
SrcNameDélka (délka názvu zdrojového objektu)	Není	0
SrcNameOffset (offset názvu zdrojového objektu)	Není	0
DestEnvDestEnv (délka dat cílového prostředí)	Není	0
DestEnvOffset (posun dat cílového prostředí)	Není	0
DestNameDélka (délka názvu cílového objektu)	Není	0
DestNameOffset (posunutí názvu cílového objektu)	Není	0
DataLogicalDélka (délka hromadných dat)	Není	0
DataLogicalOffset (nízké posunutí hromadných dat)	Není	0
DataLogicalOffset2 (vysoké posunutí hromadných dat)	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQRMH_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQRMH MyRMH = {MQRMH_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQRMH

```
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 */
};
```

Deklarace jazyka COBOL pro MQRMH

```
**   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.
```

Deklarace PL/I pro MQRMH

```
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 */
```

Deklarace High Level Assembler pro MQRMH

```
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)
```

Vizuální základní deklarace pro MQRMH

```
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
```

StrucId (MQCHAR4) pro MQRMH

Jedná se o identifikátor struktury struktury záhlaví referenční zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQRMH_STRUC_ID.

Hodnota musí být:

MQRMH_STRUC_ID
Identifikátor pro strukturu záhlaví referenční zprávy.

Pro programovací jazyk C je také definována konstanta MQRMH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQRMH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQRMH

Číslo verze struktury. Hodnota musí být:

MQRMH_VERSION_1
Struktura záhlaví zprávy odkazu Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

MQRMH_CURRENT_VERSION
Aktuální verze struktury záhlaví referenční zprávy.

Počáteční hodnota tohoto pole je MQRMH_VERSION_1.

StrucLength (MQLONG) pro MQRMH

Celková délka MQRMH, včetně řetězců na konci pevných polí, ale ne hromadných dat.

Počáteční hodnota tohoto pole je nula.

Kódování (MQLONG) pro MQRMH

Tato volba určuje číselné kódování hromadných dat; nevztahuje se na číselná data v samotné struktuře MQRMH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG) pro MQRMH

Tato volba určuje identifikátor znakové sady pro hromadná data; nevztahuje se na znaková data v samotné struktuře MQRMH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

MQCCSI_INHERIT
Znaková data v datech následujících za touto strukturou jsou ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota MQCCSI_INHERIT vrácena voláním MQGET.

Nepoužívejte MQCCSI_INHERIT, pokud je hodnota pole `PutApplType` v MQMD MQAT_BROKER.

Tato hodnota je podporována v následujících prostředích:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

a pro klienty IBM MQ připojené k těmto systémům.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8) pro MQRMH

Tato volba určuje název formátu hromadných dat.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *`Format`* v MQMD.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQRMH

Jedná se o příznaky referenční zprávy. Jsou definovány následující příznaky:

MQRMHF_LAST
Tento příznak označuje, že referenční zpráva představuje nebo obsahuje poslední část odkazovaného objektu.

MQRMHF_NOT_LAST
Referenční zpráva neobsahuje ani nepředstavuje poslední část objektu. MQRMHF_NOT_LAST pomáhá s dokumentací programu. Není zamýšleno, aby tato volba byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Počáteční hodnota tohoto pole je MQRMHF_NOT_LAST.

ObjectType (MQCHAR8) pro MQRMH

Jedná se o název, který může uživatelská procedura zpráv použít k rozpoznání typů referenčních zpráv, které podporuje. Název musí odpovídat stejným pravidlům jako pole *`Format`* , viz “Formát (MQCHAR8) pro MQRMH” na stránce 556.

Počáteční hodnota tohoto pole je 8 mezer.

ObjectInstanceId (MQBYTE24) pro MQRMH

Toto pole použijte k identifikaci specifické instance objektu. Není-li potřeba, nastavte ji na následující hodnotu:

MQOII_NONE
Nebyl uveden žádný identifikátor instance objektu. Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQOII_NONE_ARRAY; má stejnou hodnotu jako MQOII_NONE, ale je to pole znaků místo řetězce.

Délka tohoto pole je dána hodnotou MQ_OBJECT_INSTANCE_ID_LENGTH. Počáteční hodnota tohoto pole je MQOII_NONE.

SrcEnvDélka (MQLONG) pro MQRMH

Délka zdrojových dat prostředí. Je-li toto pole nula, nejsou k dispozici žádná data zdrojového prostředí a parametr *SrcEnvOffset* je ignorován.

Počáteční hodnota tohoto pole je 0.

SrcEnvOffset (MQLONG)

Toto pole určuje posun zdrojových dat prostředí od začátku struktury MQRMH. Data zdrojového prostředí mohou být určena tvůrcem referenční zprávy, pokud jsou tato data tvůrci známa. Například v systému Windows mohou být zdrojová data prostředí cestou k adresáři objektu, který obsahuje hromadná data. Pokud však tvůrce nezná data zdrojového prostředí, uživatelská procedura pro zprávy musí určit všechny potřebné informace o prostředí.

Délka zdrojových dat prostředí je dána *SrcEnvLength* ; Je-li tato délka nulová, nejsou k dispozici žádná data zdrojového prostředí a parametr *SrcEnvOffset* je ignorován. Pokud jsou k dispozici, data zdrojového prostředí musí být od začátku struktury zcela v rozsahu *StrucLength* bajtů.

Aplikace nesmí předpokládat, že data prostředí se spustí okamžitě po posledním pevném poli ve struktuře, nebo že jsou souvislá s libovolnými daty adresovanými poli *SrcNameOffset*, *DestEnvOffset*a *DestNameOffset* .

Počáteční hodnota tohoto pole je 0.

SrcNameDélka (MQLONG) pro MQRMH

Délka názvu zdrojového objektu. Je-li toto pole nulové, neexistuje žádný název zdrojového objektu a parametr *SrcNameOffset* je ignorován.

Počáteční hodnota tohoto pole je 0.

SrcNameOffset (MQLONG) pro MQRMH

Toto pole určuje posun názvu zdrojového objektu od začátku struktury MQRMH. Název zdrojového objektu může určit tvůrce referenční zprávy, pokud jsou tato data tvůrci známa. Pokud však tvůrce nezná název zdrojového objektu, uživatelská procedura pro zprávy musí identifikovat objekt, ke kterému má být přistupováno.

Délka názvu zdrojového objektu je dána *SrcNameLength* ; Je-li tato délka nula, neexistuje žádný název zdrojového objektu a parametr *SrcNameOffset* je ignorován. Je-li uveden, musí být název zdrojového objektu zcela v rozmezí *StrucLength* bajtů od začátku struktury.

Aplikace nesmí předpokládat, že název zdrojového objektu je souvislý s žádným z dat adresovaných poli *SrcEnvOffset*, *DestEnvOffset*a *DestNameOffset* .

Počáteční hodnota tohoto pole je 0.

DestEnvDélka (MQLONG) pro MQRMH

Jedná se o délku dat cílového prostředí. Je-li toto pole nulové, nejsou k dispozici žádná data cílového prostředí a parametr *DestEnvOffset* je ignorován.

DestEnvOffset (MQLONG) pro MQRMH

Toto pole určuje posun dat cílového prostředí od začátku struktury MQRMH. Data cílového prostředí mohou být určena tvůrcem referenční zprávy, pokud jsou tato data tvůrci známa. Například v systému Windows mohou být data cílového prostředí cestou k adresáři objektu, kde se mají hromadná data uložit. Pokud však tvůrce nezná data cílového prostředí, je odpovědností uživatelské procedury pro zprávy dodané uživatelem určit potřebné informace o prostředí.

Délku dat cílového prostředí uvádí *DestEnvLength* ; Je-li tato délka nula, nejsou k dispozici žádná data cílového prostředí a parametr *DestEnvOffset* je ignorován. Jsou-li k dispozici, musí být data cílového prostředí od začátku struktury zcela v rozsahu *StrucLength* bajtů.

Aplikace nesmí předpokládat, že data cílového prostředí jsou souvislá s jakýmikoli daty adresovanými poli *SrcEnvOffset*, *SrcNameOffset*a *DestNameOffset* .

Počáteční hodnota tohoto pole je 0.

DestNameDélka (MQLONG) pro MQRMH

Délka názvu cílového objektu. Je-li toto pole nula, neexistuje žádný název cílového objektu a parametr *DestNameOffset* je ignorován.

DestNameOffset (MQLONG) pro MQRMH

Toto pole určuje posun názvu cílového objektu od začátku struktury MQRMH. Název cílového objektu může určit tvůrce referenční zprávy, jsou-li tato data známa tvůrci. Pokud však tvůrce nezná název cílového objektu, je odpovědností uživatelské procedury pro zprávy určené k identifikaci objektu, který má být vytvořen nebo upraven.

Délka názvu cílového objektu je dána *DestNameLength* ; Je-li tato délka nulová, neexistuje žádný název cílového objektu a parametr *DestNameOffset* je ignorován. Je-li uveden, název cílového objektu musí být od začátku struktury zcela v rozmezí *StrucLength* bajtů.

Aplikace nesmí předpokládat, že název cílového objektu je souvislý se všemi daty adresovanými poli *SrcEnvOffset*, *SrcNameOffset*a *DestEnvOffset* .

Počáteční hodnota tohoto pole je 0.

DataLogicalDélka (MQLONG) pro MQRMH

Pole *DataLogicalLength* určuje délku hromadných dat, na která odkazuje struktura MQRMH.

Pokud jsou ve zprávě skutečně přítomna hromadná data, data začínají s posunem *StrucLength* bajtů od začátku struktury MQRMH. Délka celé zprávy minus *StrucLength* udává délku přítomných hromadných dat.

Pokud jsou ve zprávě přítomna data, *DataLogicalLength* uvádí množství příslušných dat. Normální případ znamená, že *DataLogicalLength* má mít stejnou hodnotu jako délka dat přítomných ve zprávě.

Pokud struktura MQRMH představuje zbývající data v objektu (počínaje zadaným logickým offsetem), můžete pro parametr *DataLogicalLength*použít hodnotu nula za předpokladu, že hromadná data nejsou ve zprávě skutečně přítomna.

Pokud nejsou k dispozici žádná data, konec MQRMH se shoduje s koncem zprávy.

Počáteční hodnota tohoto pole je 0.

DataLogicalOffset (MQLONG) pro MQRMH

Toto pole určuje nízkou odchylku hromadných dat od začátku objektu, jehož část hromadných dat tvoří. Posun hromadných dat od začátku objektu se nazývá *logický posun*. Nejedná se o fyzické posunutí hromadných dat od začátku struktury MQRMH; toto posunutí je dáno *StrucLength*.

Chcete-li povolit odesílání velkých objektů pomocí referenčních zpráv, logický posun je rozdělen do dvou polí a skutečný logický posun je dán součtem těchto dvou polí:

- *DataLogicalOffset* představuje zbytek získaný při dělení logického posunu o 1 000 000 000. Jedná se tedy o hodnotu v rozsahu 0 až 999 999 999 999.
- *DataLogicalOffset2* představuje výsledek získaný vydělením logického posunu 1 000 000 000. Je to tedy počet úplných násobků 1 000 000 000, které existují v logickém posunu. Počet násobků je v rozsahu 0 až 999 999 999 999.

Počáteční hodnota tohoto pole je 0.

DataLogicalOffset2 (MQLONG) pro MQRMH

Toto pole určuje vysoké posunutí hromadných dat od začátku objektu, jehož část hromadných dat tvoří. Jedná se o hodnotu v rozsahu 0 až 999 999 999 999. Podrobnosti viz *DataLogicalOffset* .

Počáteční hodnota tohoto pole je 0.

MQRR-záznam odezvy

Použijte strukturu MQRR k přijetí kódu dokončení a kódu příčiny, který je výsledkem operace otevření nebo vložení pro jednu cílovou frontu, když je cílem distribuční seznam. MQRR je výstupní struktura pro volání MQOPEN, MQPUT a MQPUT1 .

Dostupnost

Struktura MQRR je k dispozici na následujících platformách:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

a pro klienty IBM MQ připojené k těmto systémům.

Znaková sada a kódování

Data v MQRR musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pouľití

Poskytnutím pole těchto struktur ve voláních MQOPEN a MQPUT nebo ve volání MQPUT1 můžete určit kódy dokončení a kódy příčiny pro všechny fronty v rozdělovníku, když je výsledek volání smíšený, tj. když je volání pro některé fronty v seznamu úspěšné, ale pro jiné selže. Kód příčiny MQRC_MULTIPLE_REASON z volání označuje, že záznamy odezvy (jsou-li poskytovány aplikací) byly nastaveny správcem front.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 524. Pole v MQRR

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
CompCode (kód dokončení pro frontu)	MQCC_OK	0
Příčina (kód příčiny pro frontu)	MQRC_NONE	0

Notes:

1. V programovacím jazyku C se jedná o proměnnou makra.MQRR_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQRR MyRR = {MQRR_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQRR

```
typedef struct tagMQRR MQRR;
struct tagMQRR {
  MQLONG  CompCode;  /* Completion code for queue */
  MQLONG  Reason;    /* Reason code for queue */
};
```

Deklarace jazyka COBOL pro MQRR

```
**   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.
```

Deklarace PL/I pro MQRR

```
dcl
 1 MQRR based,
  3 CompCode fixed bin(31), /* Completion code for queue */
  3 Reason   fixed bin(31); /* Reason code for queue */
```

Deklarace jazyka Visual Basic pro MQRR

```
Type MQRR
  CompCode As Long 'Completion code for queue'
  Reason   As Long 'Reason code for queue'
End Type
```

CompCode (MQLONG) pro MQRR

Jedná se o kód dokončení, který je výsledkem operace otevření nebo vložení pro frontu s názvem určeným příslušným prvkem v poli struktur MQOR poskytnutém ve volání MQOPEN nebo MQPUT1 .

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je MQCC_OK.

Příčina (MQLONG) pro MQRR

Jedná se o kód příčiny, který je výsledkem operace otevření nebo vložení pro frontu s názvem, který byl určen odpovídajícím prvkem v poli struktur MQOR poskytnutém ve volání MQOPEN nebo MQPUT1 .

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je MQRC_NONE.

Volby konfigurace MQSCO-SSL/TLS

Struktura MQSCO ve spojení s poli TLS ve struktuře MQCD umožňuje aplikaci spuštěné jako IBM MQ MQI client určit volby konfigurace, které řídí použití TLS pro připojení klienta, když je kanálový protokol TCP/IP. Struktura je vstupní parametr volání MQCONNX.

Dostupnost

Struktura MQSCO je k dispozici na následujících klientech:

- AIX
- IBM i
- Linux

- Windows Windows

Není-li pro kanál klienta použit protokol TCP/IP, bude struktura MQSCO ignorována.

Znaková sada a kódování

Data v MQSCO musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a v kódování lokálního správce front dané proměnnou MQENC_NATIVE.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 525. Pole v souboru MQSCO

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQSCO_STRUC_ID	'SCO¬'
Verze (číslo verze struktury)	MQSCO_CURRENT_VERSION	1
KeyRepository (umístění úložiště klíčů)	Není	Prázdný řetězec nebo mezery
CryptoHardware (podrobnosti o šifrovacím hardwaru)	Není	Prázdný řetězec nebo mezery
AuthInfoRecCount (počet přítomných záznamů MQAIR)	Není	0
AuthInfoRecOffset (posunutí prvního záznamu MQAIR od začátku MQSCO)	Není	0
AuthInfoRecPtr (adresa prvního záznamu MQAIR)	Není	Ukazatel Null nebo bajty s hodnotou Null
Poznámka: Následující dvě pole jsou ignorována, pokud je hodnota *Version* menší než MQSCO_VERSION_2.		
KeyResetPočet (počet resetů tajného klíče TLS)	MQSCO_RESET_COUNT_DEFAULT	0
"FipsRequired (MQLONG) pro MQSCO" na stránce 567 (použití šifrovacích algoritmů s certifikací FIPS v IBM MQ)	MQSSL_FIPS_NO	0
Poznámka: Následující dvě pole jsou ignorována, pokud je hodnota *Version* menší než MQSCO_VERSION_3.		
EncryptionPolicySuiteB (používat pouze šifrovací algoritmy Suite B)	MQ_SUITE_B_NONE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE	1, 0, 0, 0

Tabulka 525. Pole v souboru MQSCO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
Poznámka: Následující dvě pole jsou ignorována, pokud je hodnota *Version* menší než MQSCO_VERSION_4.		
CertificateValPolicy (zásada ověření certifikátu)	MQ_CERT_VAL_POLICY_DEFAULT	0
Poznámka: Následující dvě pole jsou ignorována, pokud je hodnota *Version* menší než MQSCO_VERSION_5.		
CertificateLabel (podrobnosti o popisku certifikátu, který se používá)	Není	Prázdný řetězec nebo mezery
Poznámka: Zbývající pole se ignorují, pokud je *Version* menší než MQSCO_VERSION_6.		
V 9.3.0 V 9.3.0 KeyRepoPasswordPtr (adresa hesla úložiště klíčů TLS)	Není	Ukazatel Null nebo bajty s hodnotou Null
V 9.3.0 V 9.3.0 KeyRepoPasswordOffset (posun hesla úložiště klíčů TLS)	Není	0
V 9.3.0 V 9.3.0 KeyRepoPasswordLength (délka hesla úložiště klíčů TLS)	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQSCO_DEFAULT obsahuje hodnoty uvedené v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQSCO MySCO = {MQSCO_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQSCO

```
typedef struct tagMQSCO MQSCO;
struct tagMQSCO {
   MQCHAR4    StrucId;                  /* Structure identifier */
   MQLONG     Version;                  /* Structure version number */
   MQCHAR256  KeyRepository;            /* Location of TLS key */
                                        /* repository */
   MQCHAR256  CryptoHardware;           /* Cryptographic hardware */
                                        /* configuration string */
   MQLONG     AuthInfoRecCount;         /* Number of MQAIR records */
                                        /* present */
   MQLONG     AuthInfoRecOffset;        /* Offset of first MQAIR */
                                        /* record from start of */
                                        /* MQSCO structure */
   PMQAIR     AuthInfoRecPtr;           /* Address of first MQAIR */
                                        /* record */
/* Ver:1 */
   MQLONG     KeyResetCount;            /* Number of unencrypted */
                                        /* bytes sent/received */
                                        /* before secret key is */
                                        /* reset */
```

```
   MQLONG     FipsRequired;               /* Using FIPS-certified */
/* Ver:2 */
                                          /* algorithms */
   MQLONG     EncryptionPolicySuiteB[4];  /* Use only Suite B */
/* Ver:3 */
                                          /* cryptographic algorithms */
   MQLONG     CertificateValPolicy;       /* Certificate validation */
                                          /* policy */
/* Ver:4 */
   MQCHAR64   CertificateLabel;           /* Certificate label */
/* Ver:5 */
   MQPTR      KeyRepoPasswordPtr;         /* Address of key */
                                          /* repository password */
   MQLONG     KeyRepoPasswordOffset;      /* Offset of key repository */
                                          /* password */
   MQLONG     KeyRepoPasswordLength;      /* Length of key repository */
                                          /* password */
/* Ver:6 */
  };
```

Deklarace jazyka COBOL pro MQSCO

```
**   MQSCO structure
  10 MQSCO.
**    Structure identifier
   15 MQSCO-STRUCID                  PIC X(4).
**    Structure version number
   15 MQSCO-VERSION                  PIC S9(9) BINARY.
**    Location of TLS key repository
   15 MQSCO-KEYREPOSITORY            PIC X(256).
**    Cryptographic hardware configuration string
   15 MQSCO-CRYPTOHARDWARE           PIC X(256).
**    Number of MQAIR records present
   15 MQSCO-AUTHINFORECCOUNT         PIC S9(9) BINARY.
**    Offset of first MQAIR record from start of MQSCO structure
   15 MQSCO-AUTHINFORECOFFSET        PIC S9(9) BINARY.
**    Address of first MQAIR record
   15 MQSCO-AUTHINFORECPTR           POINTER.
** Version 1 **
**    Number of unencrypted bytes sent/received before secret key is
**    reset
   15 MQSCO-KEYRESETCOUNT            PIC S9(9) BINARY.
**    Using FIPS-certified algorithms
   15 MQSCO-FIPSREQUIRED             PIC S9(9) BINARY.
** Version 2 **
**    Use only Suite B cryptographic algorithms
   15 MQSCO-ENCRYPTIONPOLICYSUITEB   PIC S9(9) BINARY OCCURS 4.
** Version 3 **
**    Certificate validation policy setting
   15 MQSCO-CERTIFICATEVALPOLICY     PIC S9(9) BINARY.
** Version 4 **
**    SSL/TLS certificate label
   15 MQSCO-CERTIFICATELABEL         PIC X(64).
** Version 5 **
**    Add padding to ensure that pointers start on correct
**    boundaries
   15 FILLER                         PIC S9(9) BINARY VALUE 0.
**   Address of key repository password
   15 MQSCO-KEYREPOPASSWORDPTR       POINTER.
**   Offset of key repository password
   15 MQSCO-KEYREPOPASSWORDOFFSET    PIC S9(9) BINARY.
**   Length of key repository password
   15 MQSCO-KEYREPOPASSWORDLENGTH    PIC S9(9) BINARY.
** Version 6 **
```

Prohlášení PL/I pro MQSCO

```
dcl
 1 MQSCO based,
  3 StrucId                   char(4),       /* Structure identifier */
  3 Version                   fixed bin(31), /* Structure version number */
  3 KeyRepository             char(256),     /* Location of TLS key
                                                repository */
  3 CryptoHardware            char(256),     /* Cryptographic hardware
                                                configuration string */
  3 AuthInfoRecCount          fixed bin(31), /* Number of MQAIR records
                                                present */
```

```
  3 AuthInfoRecOffset          fixed bin(31), /* Offset of first MQAIR record
                                                 from start of MQSCO structure */
  3 AuthInfoRecPtr             pointer,       /* Address of first MQAIR record */
  3 KeyResetCount              fixed bin(31), /* Key reset count */
/* Version 1 */
  3 FipsRequired               fixed bin(31), /* FIPS required */
/* Version 2 */
  3 EncryptionPolicySuiteB (4) fixed bin(31), /* Suite B encryption policy */
/* Version 3 */
  3 CertificateValPolicy       fixed bin(31), /* Certificate validation policy */
/* Version 4 */
  3 CertificateLabel           char(64),      /* SSL/TLS certificate label */
/* Version 5 */
  3 KeyRepoPasswordPtr         pointer,       /* Address of key repository
                                                 password */
  3 KeyRepoPasswordOffset      fixed bin(31), /* Offset of key repository
                                                 password */
  3 KeyRepoPasswordLength      fixed bin(31); /* Length of key repository
                                                 password */
/* Version 6 */
```

Vizuální základní deklarace pro MQSCO

```
Type MQSCO
  StrucId           As String*4   'Structure identifier'
  Version           As Long       'Structure version number'
  KeyRepository     As String*256 'Location of TLS key repository'
  CryptoHardware    As String*256 'Cryptographic hardware configuration'
                                  '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
```

Související odkazy

“MQCNO-Volby připojení” na stránce 318

Struktura MQCNO umožňuje aplikaci určit volby související s připojením ke správci front. Struktura je vstupní/výstupní parametr volání MQCONNX.

StrucId (MQCHAR4) pro MQSCO

Toto je identifikátor struktury struktury voleb konfigurace SSL/TLS. Vždy se jedná o vstupní pole. Jeho hodnota je MQSCO_STRUC_ID.

Hodnota musí být:

MQSCO_STRUC_ID
Identifikátor struktury voleb konfigurace SSL/TLS.

Pro programovací jazyk C je definována také konstanta MQSCO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQSCO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQSCO

Toto je číslo verze struktury; hodnota musí být:

MQSCO_VERSION_1
Version-1 Struktura voleb konfigurace TLS.

MQSCO_VERSION_2
Version-2 Struktura voleb konfigurace TLS.

MQSCO_VERSION_3
Version-3 Struktura voleb konfigurace TLS.

MQSCO_VERSION_4
Struktura voleb konfigurace TLS Version-4 .

MQSCO_VERSION_5
Version-5 Struktura voleb konfigurace TLS.

V 9.3.0 V 9.3.0 **MQSCO_VERSION_6**
Version-6 Struktura voleb konfigurace TLS.

Následující konstanta určuje číslo verze aktuální verze:

MQSCO_CURRENT_VERSION
Aktuální verze struktury voleb konfigurace TLS.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSCO_VERSION_1.

Multi *KeyRepository (MQCHAR256) pro MQSCO*

Toto pole je relevantní pouze pro IBM MQ MQI clients spuštění na systémech IBM i, AIX, Linux, and Windows . Určuje umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty. Není-li přípona souboru uvedena, automaticky se přidá přípona `.kdb` .

Každý soubor databáze klíčů může mít přidružený *soubor pro uložení hesla*. Soubor pro dočasné ukládání obsahuje kódovaná hesla, která umožňují programový přístup k databázi klíčů. Soubor pro dočasné ukládání hesla musí být umístěn ve stejném adresáři a musí mít stejný systém souborů jako databáze klíčů a musí končit příponou `.sth`.

Pokud je například soubor databáze klíčů `/xxx/yyy/key.kdb`, pak soubor pro dočasné ukládání hesla musí být `/xxx/yyy/key.sth`, kde `xxx` a `yyy` představují názvy adresářů.

V 9.3.0 V 9.3.0 Heslo databáze klíčů lze také zadat pomocí pole *KeyRepoPasswordPtr* nebo *KeyRepoPasswordOffset* struktury MQSCO.

Pokud je hodnota kratší než délka pole, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku pole. Hodnota není zaškrtnuta. Pokud dojde k chybě při přístupu k úložišti klíčů, volání se nezdaří s kódem příčiny MQRC_KEY_REPOSITORY_ERROR.

Chcete-li spustit připojení TLS z IBM MQ MQI client, nastavte *KeyRepository* na platný název souboru databáze klíčů.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_SSL_KEY_REPOSITORY_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.

CryptoHardware (MQCHAR256) pro MQSCO

V tomto poli jsou uvedeny podrobnosti o konfiguraci šifrovacího hardwaru připojeného ke klientskému systému.

Nastavte pole na řetězec v následujícím formátu, nebo jej ponechte prázdný nebo null:

```
GSK_PKCS11=the PKCS #11 driver path and file name;the PKCS #11
token label;the PKCS #11 token password;symmetric cipher setting;
```

Chcete-li použít šifrovací hardware, který je v souladu s rozhraním PKCS #11 , například IBM 4960 nebo IBM 4764, musíte zadat cestu k ovladači PKCS #11 , popisek tokenu PKCS #11 a řetězce hesel tokenu PKCS #11 , které jsou ukončeny středníkem.

Cesta k ovladači PKCS #11 je absolutní cesta ke sdílené knihovně poskytující podporu pro kartu PKCS #11 . Název souboru ovladače PKCS #11 je název sdílené knihovny. Příklad hodnoty požadované pro cestu a název souboru PKCS #11 je:

```
/usr/lib/pkcs11/PKCS11_API.so
```

Popisek tokenu PKCS #11 se musí shodovat s popiskem, který jste nakonfigurovali pro hardware.

Pokud není požadována žádná konfigurace šifrovacího hardwaru, nastavte pole na prázdnou hodnotu nebo na hodnotu null.

Pokud je hodnota kratší než délka pole, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku pole. Pokud hodnota není platná nebo vede k selhání při použití ke konfiguraci šifrovacího hardwaru, volání selže s kódem příčiny MQRC_CRYPTO_HARDWARE_ERROR.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_SSL_CRYPTO_HARDWARE_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.

AuthInfoRecCount (MQLONG) pro MQSCO

Jedná se o počet záznamů ověřovacích informací (MQAIR) adresovaných poli *AuthInfoRecPtr* nebo *AuthInfoRecOffset* . Další informace viz"MQAIR-záznam ověřovacích informací" na stránce 272. Hodnota musí být nula nebo větší. Pokud hodnota není platná, volání selže s kódem příčiny MQRC_AUTH_INFO_REC_COUNT_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

AuthInfoRecOffset (MQLONG) pro MQSCO

Jedná se o posun v bajtech prvního záznamu ověřovacích informací od začátku struktury MQSCO. Posun může být kladný nebo záporný. Je-li hodnota *AuthInfoRecCount* nula, pole se ignoruje.

K určení záznamů MQAIR můžete použít buď *AuthInfoRecOffset* , nebo *AuthInfoRecPtr* , ale ne obojí. Podrobnosti naleznete v popisu pole *AuthInfoRecPtr* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

AuthInfoRecPtr (PMQAIR) pro MQSCO

Jedná se o adresu prvního záznamu ověřovacích informací. Je-li hodnota *AuthInfoRecCount* nula, pole se ignoruje.

Pole záznamů MQAIR můžete poskytnout jedním ze dvou způsobů:

- Pomocí pole ukazatele *AuthInfoRecPtr*

  V tomto případě může aplikace deklarovat pole záznamů MQAIR, které je oddělené od struktury MQSCO, a nastavit *AuthInfoRecPtr* na adresu pole.

  Zvažte použití produktu *AuthInfoRecPtr* pro programovací jazyky, které podporují datový typ ukazatele způsobem, který je přenositelný do různých prostředí (například programovací jazyk C).
- Pomocí pole offsetu *AuthInfoRecOffset*

  V tomto případě musí aplikace deklarovat složenou strukturu obsahující MQSCO následovanou polem záznamů MQAIR a nastavit *AuthInfoRecOffset* na offset prvního záznamu v poli od začátku struktury MQSCO. Ujistěte se, že je tato hodnota správná a má hodnotu, kterou lze umístit v rámci MQLONG (nejrestriktivnějším programovacím jazykem je COBOL, pro který je platný rozsah -999 999 999 999 až +999 999 999 999).

  Zvažte použití *AuthInfoRecOffset* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele způsobem, který není přenositelný do různých prostředí (například programovací jazyk COBOL).

Bez ohledu na zvolenou techniku lze použít pouze jeden z parametrů *AuthInfoRecPtr* a *AuthInfoRecOffset* ; volání selže s kódem příčiny MQRC_AUTH_INFO_REC_ERROR, pokud jsou obě nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou typu all-null.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

KeyResetPočet (MQLONG) pro MQSCO

Toto představuje celkový počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS, než je znovu vyjednán tajný klíč.

Počet bajtů zahrnuje řídicí informace odeslané MCA.

Zadáte-li počet resetů tajného klíče TLS v rozsahu 1 bajt až 32 kB, budou kanály TLS používat počet resetů tajného klíče 32 kB. Tím se vyhnete nákladům na zpracování nadměrných resetů klíčů, které by se vyskytly pro malé hodnoty resetu tajného klíče TLS.

Toto je vstupní pole. Hodnota je číslo v rozsahu 0 až 999 999 999 999, s výchozí hodnotou 0. Chcete-li označit, že tajné klíče nejsou nikdy znovu vyjednány, použijte hodnotu 0.

FipsRequired (MQLONG) pro MQSCO

Produkt IBM MQ lze konfigurovat s šifrovacím hardwarem tak, aby používané šifrovací moduly byly ty, které poskytuje hardwarový produkt. Tyto moduly mohou být certifikovány FIPS na konkrétní úroveň v závislosti na používaném šifrovacím hardwarovém produktu. Toto pole použijte, chcete-li uvést, že se použijí pouze algoritmy certifikované FIPS, pokud je šifrování poskytnuto v IBM MQ-poskytnutém softwaru.

Poznámka: V systému AIX, Linux, and Windowsposkytuje produkt IBM MQ kompatibilitu se standardem FIPS 140-2 prostřednictvím šifrovacího modulu IBM Crypto for C (ICC) . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C (ICC) certifikát a měli by si být vědomi všech doporučení poskytnutých NIST. Náhradní modul FIPS 140-3 momentálně probíhá a jeho stav lze zobrazit jeho vyhledáním v modulech NIST CMVP v seznamu procesů.

Když je nainstalován produkt IBM MQ , je také nainstalována implementace šifrování TLS, která poskytuje některé moduly s certifikací FIPS.

Hodnoty mohou být:

MQSSL_FIPS_NO
Toto je výchozí hodnota. Při nastavení na tuto hodnotu:

- Lze použít libovolnou specifikaci CipherSpec podporovanou na konkrétní platformě.
- Pokud se spustí bez použití šifrovacího hardwaru, specifikace CipherSpecs se spustí s použitím certifikovaného šifrování FIPS 140-2 na platformách IBM MQ .

  Seznam specifikací CipherSpecss certifikací FIPS naleznete v tabulce popsané v tématu Povolení CipherSpecs.

MQSSL_FIPS_YES
Při nastavení na tuto hodnotu, pokud k provedení šifrování nepoužíváte kryptografický hardware, můžete si být jisti, že

- V CipherSpec , které se používají pro toto připojení klienta, lze použít pouze šifrovací algoritmy s certifikací FIPS.
- Příchozí a odchozí připojení kanálu TLS jsou úspěšná pouze v případě, že jsou použity určité specifikace šifrování.

  Další informace viz Povolení CipherSpecs .

Poznámka: Pokud jsou konfigurovány pouze specifikace CipherSpecs standardu FIPS, klient MQI odmítne připojení, která určují jinou specifikaci než FIPS CipherSpec s volbou `MQRC_SSL_INITIALIZATION_ERROR`. Produkt IBM MQ nezaručuje, že odmítne všechna taková připojení, a je vaší odpovědností určit, zda je konfigurace produktu IBM MQ kompatibilní s FIPS.

EncryptionPolicySuiteB (MQLONG) pro MQSCO

Toto pole určuje, zda je použito šifrování vyhovující standardu Suite B a jaká úroveň síly je použita. Hodnota může být jedna nebo více z následujících hodnot:

- MQ_SUITE_B_NONE

  Šifrování vyhovující standardu Suite B se nepoužívá.
- MQ_SUITE_B_128_BIT

  Používá se 128bitové zabezpečení Suite B.
- MQ_SUITE_B_192_BIT

  Používá se 192bitové zabezpečení síly Suite B.

Poznámka: Použití MQ_SUITE_B_NONE s jakoukoli jinou hodnotou v tomto poli je neplatné.

CertificateValPolicy (MQLONG) pro MQSCO

Toto pole uvádí, jaký typ zásady ověření platnosti certifikátu se používá.

Pole lze nastavit na jednu z následujících hodnot:

MQ_CERT_VAL_POLICY_ANY
Použijte všechny zásady ověřování certifikátů podporované knihovnou zabezpečených soketů. Přijměte řetěz certifikátů, pokud některá ze zásad považuje řetěz certifikátů za platný.

MQ_CERT_VAL_POLICY_RFC5280
Použijte pouze zásadu ověření certifikátu kompatibilní s produktem RFC5280 . Toto nastavení poskytuje přísnější ověření než nastavení ANY, ale odmítá některé starší digitální certifikáty.

Počáteční hodnota tohoto pole je MQ_CERT_VAL_POLICY_ANY.

CertificateLabel (MQCHAR64) pro MQSCO

V tomto poli jsou uvedeny podrobnosti o používaném popisku certifikátu.

IBM MQ inicializuje výchozí hodnotu pro pole *CertificateLabel* jako mezery.

Tato hodnota je za běhu interpretována jako výchozí hodnota a je zpětně kompatibilní.

Například při zadání verze MQSCO nižší než 5.0nebo při použití výchozí hodnoty mezer pro pole *CertificateLabel* se použije již existující výchozí hodnota `ibmwebspheremq`*`user_id`*.

V 9.3.0 Multi V 9.3.0 *KeyRepoPasswordPtr (MQPTR) pro MQSCO*

Jedná se o adresu hesla úložiště klíčů TLS v bajtech.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou Null. Toto pole je ignorováno, pokud je hodnota *`Version`* menší než MQSCO_VERSION_6.

Toto pole je relevantní pouze pro IBM MQ MQI clients spuštění na systémech IBM i, AIX, Linux, and Windows .

Přístupovou frázi úložiště klíčů lze zadat jako prostý textový řetězec nebo přístupovou frázi, která byla zašifrována pomocí obslužného programu **`runmqicred`** .

Zadáte-li šifrovanou přístupovou frázi, zadejte počáteční klíč, který byl použit k zašifrování přístupové fráze ve struktuře MQCSP poskytované stejnou klientskou aplikací.

Přístupová fráze úložiště klíčů uvedená pomocí tohoto pole přepíše jakoukoli přístupovou frázi úložiště klíčů uvedenou pomocí proměnné prostředí *MQKEYRPWD* nebo pomocí vlastnosti *SSLKeyRepositoryPassword* v sekci SSL konfiguračního souboru klienta.

K určení přístupové fráze úložiště klíčů můžete použít buď *`KeyRepoPasswordOffset`* , nebo *`KeyRepoPasswordPtr`* , ale ne obojí.

Související úlohy
Zadání počátečního klíče pro klienta MQI IBM MQ v systému AIX, Linux a Windows
Ochrana hesel v konfiguračních souborech komponenty IBM MQ
Související odkazy
runmqicred (chránit hesla klienta IBM MQ)

“InitialKeyPtr (MQPTR) pro MQCSP.” na stránce 343
Adresa pro počáteční klíč pro systém ochrany heslem.

V 9.3.0 Multi V 9.3.0 *KeyRepoPasswordOffset (MQLONG) pro MQSCO*

Toto je posun v bajtech přístupové fráze úložiště klíčů TLS od začátku struktury MQSCO. Posun může být kladný nebo záporný.

K určení přístupové fráze úložiště klíčů můžete použít buď *KeyRepoPasswordOffset* , nebo *KeyRepoPasswordPtr* , ale ne obojí. Další informace viz popis pole *KeyRepoPasswordPtr* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQSCO_VERSION_6.

V 9.3.0 Multi V 9.3.0 *KeyRepoPasswordLength (MQLONG) pro MQSCO*

Jedná se o délku přístupové fráze úložiště klíčů TLS.

V systému IBM ije maximální délka přístupové fráze úložiště klíčů 128 znaků. Pokud je přístupová fráze úložiště klíčů větší než maximální povolená délka, připojení se nezdaří s chybou MQRC_KEY_REPOSITORY_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *Version* menší než MQSCO_VERSION_6.

MQSD-deskriptor odběru

Struktura MQSD se používá k určení podrobností o provedeného odběru. Struktura je vstupní/výstupní parametr volání MQSUB.Další informace viz Poznámky k použití MQSUB.

Dostupnost

Struktura MQSD je k dispozici na následujících platformách:

- AIX AIX
- IBM i IBM i
- Linux Linux
- Windows Windows
- z/OS z/OS

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Verze

Aktuální verze MQSD je MQSD_VERSION_1.

Znaková sada a kódování

Data v modulu MQSD musí být ve znakové sadě zadané atributem správce front **CodedCharSetId** a v kódování lokálního správce front daném proměnnou MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Spravované odběry

Pokud aplikace nemá specifickou potřebu používat konkrétní frontu jako místo určení pro publikování, která odpovídají jejímu odběru, může použít funkci spravovaného odběru. Pokud aplikace zvolí použití spravovaného odběru, správce front informuje odběratele o místě určení, kam jsou odesílány publikované zprávy, a to poskytnutím popisovače objektu jako výstupu z volání MQSUB. Další informace viz Hobj (MQHOBJ)-vstupní/výstupní.

Při odebrání odběru se správce front také zavazuje k vyčištění zpráv, které nebyly načteny ze spravovaného místa určení, v následujících situacích:

- Je-li odběr odebrán pomocí příkazu MQCLOSE s parametrem MQCO_REMOVE_SUB-a spravovaný objekt Hobj je zavřen.
- Implicitní znamená, že dojde-li ke ztrátě připojení k aplikaci s použitím netrvalého odběru (MQSO_NON_TRVALÝ)
- Do vypršení platnosti, když je odběr odebrán, protože vypršela jeho platnost a spravovaný Hobj je uzavřen.

Je třeba použít spravované odběry s netrvalými odběry, aby mohlo dojít k tomuto vyčištění a aby zprávy pro uzavřené netrvalé odběry nezabívaly místo ve správci front. Trvalé odběry mohou také používat spravované cíle.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	ID_STRUC_MQSD_STRUC_ID	'SD¬¬'
Verze (číslo verze struktury)	MQSD_VERSION_1	1
Volby (volby)	MQSO_NON_TRVALÝ	0
ObjectName (název objektu)	Není	Prázdný řetězec nebo mezery
AlternateUserID (alternativní ID uživatele)	Není	Prázdný řetězec nebo mezery
AlternateSecurityID (alternativní ID zabezpečení)	MQSID_NONE	Hodnoty null
SubExpiry (vypršení platnosti odběru)	MQEI_UNLIMITED	-1
ObjectString (řetězec objektu)	Není	Názvy a hodnoty definované pro MQCHARV
SubName (název odběru)	Není	Názvy a hodnoty definované pro MQCHARV
SubUserData (uživatelská data odběru)	Není	Názvy a hodnoty definované pro MQCHARV
SubCorrelSubCorrel (ID korelace odběru)	MQCI_NONE	Hodnoty null
PubPriority (priorita publikování)	MQPRI_PRIORITY_AS_Q_DEF	-3
PubAccountingPubAccounting (token evidence publikování)	MQACT_NONE	Hodnoty null
PubAppIdentityData (data identity publikační aplikace)	Není	Prázdný řetězec nebo mezery

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
SelectionString (řetězec poskytující kritéria výběru)	Není	Názvy a hodnoty definované pro MQCHARV
SubLevel (úroveň odběru)	Není	1
ResObjectŘetězec (dlouhý název objektu)	Není	Názvy a hodnoty definované pro MQCHARV

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQSD_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQSD MySD = {MQSD_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQSD

```
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 */
};
```

Deklarace jazyka COBOL pro MQSD

```
** 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.
```

Deklarace PL/I pro MQSD

```
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 */
```

Deklarace High Level Assembler pro MQSD

```
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)
```

StrucId (MQCHAR4) pro MQSD

Jedná se o identifikátor struktury struktury deskriptoru odběru. Vždy se jedná o vstupní pole. Jeho hodnota je MQSD_STRUC_ID.

Hodnota musí být:

ID_STRUC_MQSD_STRUC_ID
Identifikátor pro strukturu deskriptoru odběru.

Pro programovací jazyk C je definována také konstanta MQSD_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQSD_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQSD

Toto je číslo verze struktury; hodnota musí být:

MQSD_VERSION_1
Version-1 Struktura deskriptoru odběru.

Následující konstanta určuje číslo verze aktuální verze:

MQSD_CURRENT_VERSION
Aktuální verze struktury deskriptoru odběru.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSD_VERSION_1.

Volby (MQLONG) pro MQSD

Tato volba poskytuje volby pro řízení akce volání MQSUB.

Musíte uvést alespoň jednu z následujících voleb:

- MQSO_ALTER
- MQSO_RESUME
- MQSO_CREATE

Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

V tomto tématu jsou uvedeny neplatné kombinace; všechny ostatní kombinace jsou platné.

Volby přístupu nebo vytvoření: Volby přístupu a vytvoření řídí, zda je vytvořen odběr, nebo zda je vrácen či změněn existující odběr. Musíte uvést alespoň jednu z těchto voleb.

Tabulka 526. Platné kombinace voleb přístupu a vytvoření

Kombinace voleb	**Notes**
MQSO_CREATE	Vytvoří odběr, pokud neexistuje. Tato kombinace selže, pokud odběr existuje.
MQSO_RESUME	Obnoví existující odběr. Tato kombinace selže, pokud neexistuje žádný odběr.
MQSO_CREATE + MQSO_RESUME	Vytvoří odběr, pokud neexistuje, a obnoví odpovídající odběr, pokud existuje. Tato kombinace je užitečná v případě, že je použita v aplikaci, která je spuštěna vícekrát.

Tabulka 526. Platné kombinace voleb přístupu a vytvoření (pokračování)	
Kombinace voleb	**Notes**
MQSO_ALTER (viz poznámka)	Obnoví existující odběr a změní pole tak, aby odpovídala poli zadanému v MQSD. Tato kombinace selže, pokud neexistuje žádný odběr.
MQSO_CREATE + MQSO_ALTER (viz poznámka)	Vytvoří odběr, pokud neexistuje, a obnoví odpovídající odběr, pokud existuje, a změní libovolná pole tak, aby odpovídala poli zadanému v MQSD. Tato kombinace je užitečná, když se používá v aplikaci, která chce zajistit, aby byl její odběr v určitém stavu, než bude pokračovat.

Poznámka:

Volby určující parametr MQSO_ALTER mohou také určovat parametr MQSO_RESUME, ale tato kombinace nemá žádný další vliv na určení parametru MQSO_ALTER samostatně. Příkaz MQSO_ALTER znamená MQSO_RESUME, protože volání MQSUB pro změnu odběru znamená, že odběr bude také obnoven. Opak není pravdou, nicméně: obnovení předplatného neznamená, že má být změněno.

MQSO_CREATE

Vytvořte nový odběr pro zadané téma. Pokud existuje odběr používající stejný *SubName* , volání se nezdaří s MQRC_SUB_ALREADY_EXISTS. Tomuto selhání se lze vyhnout kombinací volby MQSO_CREATE s volbou MQSO_RESUME. Parametr *SubName* není vždy nezbytný. Další podrobnosti viz popis tohoto pole.

Kombinace příkazu MQSO_CREATE a MQSO_RESUME vrátí manipulátor pro již existující odběr pro určený objekt *SubName* , pokud je nalezen. Pokud neexistuje žádný odběr, bude vytvořen nový odběr se všemi poli zadanými v modulu MQSD.

Příkaz MQSO_CREATE lze také kombinovat s příkazem MQSO_ALTER s podobným efektem.

MQSO_RESUME

Vraťte manipulátor k již existujícímu odběru, který odpovídá předplatnému určenému parametrem *SubName*. V odpovídajících atributech odběrů nejsou provedeny žádné změny a jsou vráceny ve výstupu ve struktuře MQSD. Používají se pouze následující pole MQSD: StrucId, Version, Options, AlternateUserId a AlternateSecurityId a SubName.

Volání selže s kódem příčiny MQRC_NO_SUBSCRIPTION, pokud neexistuje odběr odpovídající úplnému názvu odběru. Tomuto selhání se lze vyhnout kombinací volby MQSO_CREATE s volbou MQSO_RESUME.

ID uživatele odběru je ID uživatele, který vytvořil odběr, nebo pokud byl později změněn jiným ID uživatele, jedná se o ID uživatele poslední úspěšné změny. Je-li použito ID AlternateUsera pro tohoto uživatele je povoleno použití alternativních ID uživatele, bude alternativní ID uživatele zaznamenáno jako ID uživatele, který vytvořil odběr, namísto ID uživatele, pod kterým byl odběr vytvořen.

Pokud existuje odpovídající odběr, který byl vytvořen bez volby MQSO_ANY_USERID, a ID uživatele odběru se liší od ID aplikace požadující popisovač pro odběr, volání se nezdaří s kódem příčiny MQRC_IDENTITY_MISMATCH.

Pokud existuje odpovídající odběr a je aktuálně používán, volání se nezdaří s hodnotou MQRC_SUBSCRIPTION_IN_USE.

Pokud odběr uvedený v poli SubName není platným odběrem pro obnovení nebo změnu z aplikace, volání se nezdaří s MQRC_INVALID_SUBSCRIPTION.

MQSO_RESUME je odvozen z MQSO_ALTER, takže jej nemusíte kombinovat s touto volbou. Kombinace těchto dvou voleb však nezpůsobí chybu.

MQSO_ALTER

Vraťte manipulátor na již existující odběr s úplným názvem odběru, který odpovídá názvu zadanému v souboru *SubName*. Všechny atributy odběru, které se liší od atributů určených v modulu MQSD, jsou v odběru změněny, pokud není změna pro tento atribut zakázána. Podrobnosti jsou uvedeny v popisu každého atributu a jsou shrnuty v následující tabulce. Pokud se pokusíte změnit atribut, který nelze změnit, nebo změnit odběr, který nastavil volbu MQSO_IMMUTABLE, volání selže s kódem příčiny uvedeným v následující tabulce.

Volání selže s kódem příčiny MQRC_NO_SUBSCRIPTION, pokud neexistuje odběr odpovídající úplnému názvu odběru. Tomuto selhání se můžete vyhnout kombinací volby MQSO_CREATE s volbou MQSO_ALTER.

Kombinace příkazu MQSO_CREATE a příkazu MQSO_ALTER vrátí manipulátor pro již existující odběr pro určený objekt *SubName* , pokud je nalezen. Pokud neexistuje žádný odběr, bude vytvořen nový manipulátor se všemi poli zadanými v modulu MQSD.

ID uživatele odběru je ID uživatele, který vytvořil odběr, nebo pokud je později změněno jiným ID uživatele, jedná se o ID uživatele poslední úspěšné změny. Pokud je použito ID AlternateUsera pro tohoto uživatele je povoleno použití alternativních ID uživatelů, bude alternativní ID uživatele zaznamenáno jako ID uživatele, který vytvořil odběr, namísto ID uživatele, pod kterým byl odběr vytvořen.

Pokud existuje odpovídající odběr, který byl vytvořen bez volby MQSO_ANY_USERID a ID uživatele odběru se liší od ID aplikace požadující popisovač pro odběr, volání se nezdaří s kódem příčiny MQRC_IDENTITY_MISMATCH.

Pokud existuje odpovídající odběr a je aktuálně používán, volání se nezdaří s hodnotou MQRC_SUBSCRIPTION_IN_USE.

Pokud odběr uvedený v poli SubName není platným odběrem pro obnovení nebo změnu z aplikace, volání se nezdaří s MQRC_INVALID_SUBSCRIPTION.

Následující tabulka zobrazuje schopnost příkazu MQSO_ALTER změnit hodnoty atributů v MQSD a MQSUB.

Tabulka 527. Atributy v MQSD a MQSUB, které lze změnit

Deskriptor datového typu nebo volání funkce	Název pole	Lze tento atribut změnit pomocí příkazu MQSO_ALTER	Kód příčiny
MQSD	Možnosti trvanlivosti	Ne	MQRC_DURABILITY_NOT_ALTERABLE
MQSD	Volby cíle	Ano	Není
MQSD	Volby registrace	Ano (viz poznámka “1” na stránce 577)	MQRC_GROUPING_NOT_ALTERABLE, pokud se pokusíte změnit MQSO_GROUP_SUB
MQSD	Volby publikování	Ano (viz poznámka “2” na stránce 577)	Není
MQSD	Volby zástupných znaků	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD	Další volby	Ne (viz poznámka “3” na stránce 577)	Není
MQSD	ObjectName	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD	AlternateUserid	Ne (viz poznámka “4” na stránce 577)	Není
MQSD	AlternateSecurityId	Ne (viz poznámka “4” na stránce 577)	Není
MQSD	SubExpiry	Ano	Není
MQSD	ObjectString	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD	SubName	Ne (viz poznámka “5” na stránce 577)	Není
MQSD	SubUserData	Ano	Není
MQSD	SubCorrelId	Ano (viz poznámka “6” na stránce 577)	MQRC_GROUPING_NOT_ALTERABLE v seskupeném odběru

Tabulka 527. Atributy v MQSD a MQSUB, které lze změnit (pokračování)			
Deskriptor datového typu nebo volání funkce	**Název pole**	**Lze tento atribut změnit pomocí příkazu MQSO_ALTER**	**Kód příčiny**
MQSD	PubPriority	Ano	Není
MQSD	Token PubAccounting	Ano	Není
MQSD	PubApplIdentityData	Ano	Není
MQSD	SubLevel	Ne	MQRC_SUBLEVEL_NOT_ALTERABLE
MQSUB	HOBJ	Ano (viz poznámka “6” na stránce 577)	MQRC_GROUPING_NOT_ALTERABLE v seskupeném odběru

Notes:

1. MQSO_GROUP_SUB nelze změnit.
2. MQSO_NEW_PUBLICATIONS_ONLY nelze změnit, protože není součástí odběru
3. Tyto volby nejsou součástí odběru
4. Tento atribut není součástí odběru
5. Tento atribut je identitou měněného odběru
6. Lze změnit s výjimkou případů, kdy je část seskupeného podprogramu (MQSO_GROUP_SUB)

Volby trvanlivosti: Následující volby určují, jak je odběr trvalý. Můžete uvést pouze jednu z těchto voleb. Pokud měníte existující odběr pomocí volby MQSO_ALTER, nemůžete změnit trvanlivost odběru. Při návratu z volání MQSUB pomocí MQSO_RESUME je nastavena příslušná volba trvanlivosti.

MQSO_TRVALÉ
Požádejte o to, aby odběr tohoto tématu zůstal, dokud nebude explicitně odebrán pomocí příkazu MQCLOSE s volbou MQCO_REMOVE_SUB. Není-li tento odběr explicitně odebrán, zůstane zachován i po zavření tohoto připojení aplikací ke správci front.

Pokud je trvalý odběr požadován pro téma, které je definováno jako nepovolující trvalé odběry, volání selže s hodnotou MQRC_DURABILITY_NOT_ALLOWED.

MQSO_NON_TRVALÝ
Požadavek na odebrání odběru tohoto tématu při zavření připojení aplikací ke správci front, pokud již není explicitně odebrán. MQSO_NON_TRVALÉ je opakem volby MQSO_TRVALÉ a je definována jako pomocná programová dokumentace. Jedná se o předvolbu, není-li zadán ani jeden z nich.

Volby místa určení: Následující volba určuje místo určení, kam jsou odesílána publikování pro téma, k jehož odběru jste se přihlásili. Při změně existujícího odběru pomocí volby MQSO_ALTER lze změnit místo určení použité pro publikování pro daný odběr. Při návratu z volání MQSUB pomocí MQSO_RESUME je tato volba nastavena v případě potřeby.

MQSO_MANAGED

Požadujte, aby místo určení, do kterého jsou odesílána publikování, bylo spravováno správcem front.

Popisovač objektu vrácený v produktu *Hobj* představuje spravovanou frontu správce front a je určen pro použití s následnými voláními MQGET, MQCB, MQINQ nebo MQCLOSE.

Manipulátor objektu vrácený z předchozího volání MQSUB nelze v parametru **Hobj** zadat, není-li zadána hodnota MQSO_MANAGED.

Výběrové vysílání MQSO_NO_MULTICAST

Požádejte o to, aby místo určení, kam jsou publikace odesílány, nebylo adresou skupiny výběrového vysílání. Tato volba je platná pouze v kombinaci s volbou MQSO_MANAGED. Je-li v parametru **Hobj** zadán manipulátor pro frontu, nelze pro tento odběr použít výběrové vysílání a volba není platná.

Je-li téma definováno tak, aby povolovalo pouze výběrové odběry s použitím nastavení `MCAST(ONLY)` , dojde k selhání volání s kódem příčiny MQRC_MULTICAST_REQUIRED.

Volba rozsahu: Následující volba řídí rozsah odběru, který se provádí. Pokud měníte existující odběr pomocí volby MQSO_ALTER, nelze tuto volbu oboru odběru změnit. Při návratu z volání MQSUB pomocí příkazu MQSO-RESUME je nastavena příslušná volba rozsahu.

MQSO_SCOPE_QMGR
Tento odběr se provádí pouze v lokálním správci front. Žádný proxy odběr není distribuován jiným správcům front v síti. Tomuto odběrateli jsou odesílána pouze publikování publikovaná v tomto správci front. Toto potlačí jakékoli chování nastavené pomocí atributu tématu SUBSCOPE.

Poznámka: Není-li nastaven, je rozsah odběru určen atributem tématu SUBSCOPE.

Volby registrace: Podrobnosti registrace provedené ve správci front pro tento odběr řídí následující volby. Pokud měníte existující odběr pomocí volby MQSO_ALTER, lze tyto volby registrace změnit. Při návratu z volání MQSUB pomocí MQSO_RESUME jsou nastaveny příslušné volby registrace.

MQSO_GROUP_SUB

Tento odběr má být seskupen s jinými odběry stejné úrovně SubLevel s použitím stejné fronty a se stejným ID korelace, aby všechna publikování do témat, která by způsobila poskytnutí více než jedné zprávy publikování skupině odběrů, v důsledku použití překrývající se sady řetězců témat, byla do fronty doručena pouze jedna zpráva. Pokud tato volba není použita, pak každý jedinečný odběr (identifikovaný jako SubName), který se shoduje s kopií publikování, což může znamenat, že do fronty sdílené několika odběry může být umístěna více než jedna kopie publikování.

Pouze nejvýznamnější předplatné ve skupině je poskytováno s kopií publikace. Nejvýznamnější odběr je založen na úplném názvu tématu až do okamžiku, kdy je nalezen zástupný znak. Pokud je ve skupině použita směs schémat zástupných znaků, je důležitá pouze pozice zástupného znaku. Doporučuje se nekombinovat různá schémata zástupných znaků ve skupině odběrů, které sdílejí stejnou frontu.

Při vytváření nového seskupeného odběru musí mít stále jedinečný SubName, ale pokud se shoduje s úplným názvem tématu existujícího odběru ve skupině, volání selže s MQRC_DUPLICATE_GROUP_SUB.

Pokud nejvýznamnější odběr ve skupině také uvádí MQSO_NOT_OWN_PUBS a jedná se o publikaci ze stejné aplikace, pak se do fronty nedoručí žádné publikování.

Při změně odběru provedeného pomocí této volby nelze změnit pole, která implikují seskupení, Hobj ve volání MQSUB (reprezentující název fronty a správce front) a ID SubCorrel. Pokus o jejich změnu způsobí selhání volání MQRC_GROUPING_NOT_ALTERABLE.

Tuto volbu je třeba kombinovat s hodnotou MQSO_SET_CORREL_ID s hodnotou SubCorrelId, která není nastavena na hodnotu MQCI_NONE, a nelze ji kombinovat s hodnotou MQSO_MANAGED.

MQSO_ANY_USERID

Je-li zadána hodnota MQSO_ANY_USERID, není identita odběratele omezena na jediné ID uživatele. To umožňuje každému uživateli změnit nebo obnovit odběr, pokud má odpovídající oprávnění. Odběr může mít v daném okamžiku pouze jeden uživatel. Pokus o obnovení použití odběru, který je aktuálně používán jinou aplikací, způsobí selhání volání MQRC_SUBSCRIPTION_IN_USE.

Chcete-li přidat tuto volbu k existujícímu odběru, musí volání MQSUB (pomocí příkazu MQSO_ALTER) pocházet ze stejného ID uživatele jako samotný původní odběr.

Pokud volání MQSUB odkazuje na existující odběr se sadou MQSO_ANY_USERID a ID uživatele se liší od původního odběru, volání bude úspěšné pouze v případě, že nové ID uživatele má oprávnění k odběru tématu. Po úspěšném dokončení jsou budoucí publikování pro tohoto odběratele vložena do fronty odběratelů s novým ID uživatele nastaveným ve zprávě publikování.

Nezadávejte současně MQSO_ANY_USERID a MQSO_FIXED_USERID. Není-li zadána žádná z těchto hodnot, bude použita výchozí hodnota MQSO_FIXED_USERID.

MQSO_FIXED_USERID

Je-li zadána hodnota MQSO_FIXED_USERID, může být odběr změněn nebo obnoven pouze posledním ID uživatele, aby se změnil odběr. Pokud odběr nebyl změněn, jedná se o ID uživatele, který jej vytvořil.

Pokud příkaz MQSUB odkazuje na existující odběr se sadou MQSO_ANY_USERID a změní odběr pomocí příkazu MQSO_ALTER pro použití volby MQSO_FIXED_USERID, bude ID uživatele odběru nyní opraveno na toto nové ID uživatele. Volání je úspěšné pouze v případě, že nové ID uživatele má oprávnění přihlásit se k odběru tématu.

Pokud se jiné ID uživatele, než které bylo zaznamenáno jako vlastník odběru, pokusí obnovit nebo změnit odběr MQSO_FIXED_USERID, volání se nezdaří s MQRC_IDENTITY_MISMATCH. ID vlastnícího uživatele odběru lze zobrazit pomocí příkazu DISPLAY SBSTATUS.

Nezadávejte současně MQSO_ANY_USERID a MQSO_FIXED_USERID. Není-li zadána žádná z těchto hodnot, bude použita výchozí hodnota MQSO_FIXED_USERID.

Volby publikování: Způsob odesílání publikování tomuto odběrateli určují následující volby. Pokud měníte existující odběr pomocí volby MQSO_ALTER, můžete tyto volby publikování změnit.

MQSO_NOT_OWN_PUBS
Zprostředkovateli sdělí, že aplikace nechce zobrazit žádné vlastní publikace. Publikace jsou považovány za pocházející ze stejné aplikace, pokud jsou manipulátory připojení stejné. Při návratu z volání MQSUB pomocí MQSO_RESUME je tato volba nastavena v případě potřeby.

MQSO_NEW_PUBLICATIONS_ONLY
Při vytvoření tohoto odběru nejsou odesílána žádná zachovaná publikování, pouze nová publikování. Tato volba se používá pouze v případě, že je zadána volba MQSO_CREATE. Žádné následné změny v odběru nemění tok publikování, a proto všechna publikování uchovaná v tématu již byla odběrateli odeslána jako nová publikování.

Je-li tato volba zadána bez příkazu MQSO_CREATE, volání selže s chybou MQRC_OPTIONS_ERROR. Při návratu z volání MQSUB pomocí MQSO_RESUME není tato volba nastavena ani v případě, že byl odběr vytvořen pomocí této volby.

Není-li tato volba použita, jsou dříve uchované zprávy odesílány do zadané cílové fronty. Pokud se tato akce nezdaří kvůli chybě, buď MQRC_RETAINED_MSG_Q_ERROR, nebo MQRC_RETAINED_NOT_DELIVERED, vytvoření odběru se nezdaří.

MQSO_PUBLICATIONS_ON_REQUEST
Nastavení této volby označuje, že odběratel bude požadovat informace specificky v případě potřeby. Správce front neodesílá nevyžádané zprávy odběrateli. Zachované publikování (nebo případně více publikování, je-li v tématu uveden zástupný znak) je odesláno odběrateli při každém volání MQSUBRQ s použitím manipulátoru Hsub z předchozího volání MQSUB. V důsledku volání MQSUB pomocí této volby nejsou odesílána žádná publikování. Při návratu z volání MQSUB pomocí MQSO_RESUME je tato volba nastavena v případě potřeby.

Tato volba není platná v kombinaci s hodnotou SubLevel větší než 1.

Volby dopředného čtení: Následující volby řídí, zda jsou přechodné zprávy odesílány do aplikace před aplikací, která je požaduje.

MQSO_READ_AHEAD_AS_Q_DEF
Pokud volání MQSUB používá spravovaný popisovač, výchozí atribut dopředného čtení modelové fronty přidružené k odebíranému tématu určuje, zda jsou zprávy odesílány do aplikace před tím, než si je aplikace vyžádá.

Toto je výchozí hodnota.

MQSO_NO_READ_AHEAD
Pokud volání MQSUB používá spravovaný manipulátor, zprávy se do aplikace neodesílají dříve, než si je aplikace vyžádá.

MQSO_READ_AHEAD
Pokud volání MQSUB používá spravovaný popisovač, mohou být zprávy odeslány aplikaci dříve, než si je aplikace vyžádá.

Poznámka:

Pro volby dopředného čtení platí následující poznámky:

1. Lze zadat pouze jednu z těchto voleb. Jsou-li zadány volby MQOOO_READ_AHEAD i MQOO_NO_READ_AHEAD, vrátí se kód příčiny MQRC_OPTIONS_ERROR. Tyto volby lze použít pouze v případě, že je zadána volba MQSO_MANAGED.
2. Nelze je použít pro MQSUB, když je předána fronta, která byla dříve otevřena. Dopředné čtení nemusí být při požadavku povoleno. Volby MQGET použité při prvním volání MQGET mohou zabránit povolení dopředného čtení. Dopředné čtení je také zakázáno, když se klient připojuje ke správci front, kde dopředné čtení není podporováno. Pokud aplikace není spuštěna jako klient IBM MQ , jsou tyto volby ignorovány.

Volby se zástupnými znaky: Následující volby určují způsob interpretace zástupných znaků v řetězci zadaném v poli ObjectString modulu MQSD. Můžete uvést pouze jednu z těchto voleb. Pokud měníte existující odběr pomocí volby MQSO_ALTER, nelze tyto volby zástupných znaků změnit. Při návratu z volání MQSUB pomocí MQSO_RESUME je nastavena příslušná volba zástupného znaku.

MQSO_WILDCARD_CHAR
Zástupné znaky pracují pouze se znaky v řetězci tématu.

Chování definované pomocí MQSO_WILDCARD_CHAR je zobrazeno v následující tabulce.

Tabulka 528. Způsob interpretace zástupných znaků

Speciální znak	**Chování**
Lomítko (/)	Žádný význam, jen další znak
Hvězdička (*)	Zástupný znak, žádný nebo více znaků
Otazník (?)	Zástupný znak, 1 znak
Procento (%)	Řídicí znak, který umožňuje použití znaků (*), (?) nebo (%) v řetězci a který není interpretován jako speciální znak, například (% *), (%?) nebo (%%).

Například publikování na následující téma:

```
/level0/level1/level2/level3/level4
```

odpovídá odběratelům pomocí následujících témat:

```
*
/*
/ level0/level1/level2/level3/*
/ level0/level1/*/level3/level4
/ level0/level1/le?el2/level3/level4
```

Poznámka: Toto použití zástupných znaků určuje přesně význam uvedený v části IBM MQ V6 a WebSphere MB V6 při použití zpráv ve formátu MQRFH1 pro publikování/odběr. Doporučuje se, aby se toto nepoužívalo pro nově napsané aplikace a používalo se pouze pro aplikace, které byly dříve spuštěny pro tuto verzi a nebyly změněny tak, aby používaly výchozí chování zástupných znaků, jak je popsáno v tématu MQSO_WILDCARD_TOPIC.

MQSO_WILDCARD_TOPIC

Zástupné znaky pracují pouze s prvky tématu v řetězci tématu. Toto je výchozí chování, pokud není vybráno.

Chování vyžadované příkazem MQSO_WILDCARD_TOPIC je uvedeno v následující tabulce:

Tabulka 529. Způsob interpretace zástupných znaků

Speciální znak	**Chování**
(/)	Oddělovač úrovně tématu
Znak čísla (#)	Zástupný znak: více úrovní tématu
Znaménko plus (+)	Zástupný znak: úroveň jednoho tématu

Notes:

Znaky (+) a (#) nejsou považovány za zástupné znaky, pokud jsou v rámci úrovně tématu smíchány s jinými znaky (včetně sebe). V následujícím řetězci jsou znaky (#) a (+) považovány za běžné znaky.

```
level0/level1/#+/level3/level#
```

Například publikování na následující téma:

```
/level0/level1/level2/level3/level4
```

odpovídá odběratelům pomocí následujících témat:

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1/+/level3/level4
```

Další volby: Následující volby řídí způsob, jakým je volání rozhraní API vydáváno, a nikoli odběr. Při návratu z volání MQSUB pomocí MQSO_RESUME jsou tyto volby nezměněny. Další informace viz část “AlternateUserID (MQCHAR12) pro MQSD” na stránce 583.

MQSO_ALTERNATE_USER_AUTHORITY

Pole AlternateUserId obsahuje identifikátor uživatele, který se má použít k ověření tohoto volání MQSUB. Volání může být úspěšné pouze v případě, že je toto ID AlternateUserautorizováno k otevření objektu se zadanými volbami přístupu, bez ohledu na to, zda je k tomu autorizován identifikátor uživatele, pod kterým je aplikace spuštěna.

MQSO_SET_CORREL_ID

Odběr má používat identifikátor korelace zadaný v poli *SubCorrelId* . Není-li tato volba určena, správce front v době odběru automaticky vytvoří identifikátor korelace a vrátí jej aplikaci v poli *SubCorrelId* . Další informace viz “SubCorrelId (MQBYTE24) pro MQSD” na stránce 585 .

Tuto volbu nelze kombinovat s volbou MQSO_MANAGED.

MQSO_SET_IDENTITY_CONTEXT

Odběr bude používat údaje o tokenu evidence a identitě aplikace zadané v polích *PubAccountingToken* a *PubApplIdentityData* .

Je-li zadána tato volba, provede se stejná kontrola autorizace, jako kdyby k cílové frontě bylo přistupováno pomocí volání MQOPEN s volbou MQOO_SET_IDENTITY_CONTEXT, s výjimkou případu, kdy je použita také volba MQSO_MANAGED, v němž neexistuje žádná kontrola autorizace v cílové frontě.

Není-li tato volba určena, jsou k publikacím odesílaným tomuto odběrateli přidruženy výchozí informace o kontextu, a to následujícím způsobem:

Tabulka 530. Výchozí informace o kontextu pro publikování odeslaná tomuto odběrateli	
Pole v deskriptoru MQMD	**Použitá hodnota**
UserIdentifier	ID uživatele přidružené k odběru v době, kdy byl odběr proveden.
AccountingToken	Určuje se z prostředí, je-li to možné. Pokud ne, nastavte na hodnotu MQACT_NONE.
ApplIdentityData	Nastavit na mezery

Tato volba je platná pouze pro příkazy MQSO_CREATE a MQSO_ALTER. Pokud se používá s MQSO_RESUME, pole *PubAccountingToken* a *PubApplIdentityData* se ignorují, takže tato volba nemá žádný účinek.

Pokud dojde ke změně odběru bez použití této volby, kde byly dříve zadány informace o kontextu identity, budou pro změněný odběr vygenerovány výchozí informace o kontextu.

Pokud je odběr, který umožňuje použití různých ID uživatelů s volbou MQSO_ANY_USERID, obnoven jiným ID uživatele, vygeneruje se výchozí kontext identity pro nové ID uživatele, které nyní vlastní odběr, a všechna následná publikování budou doručena s novým kontextem identity.

MQSO_FAIL_IF_QUIESCING

Volání MQSUB se nezdaří, pokud je správce front ve stavu uvedení do klidového stavu. V systému z/OSpro aplikaci CICS nebo IMS tato volba také vynutí selhání volání MQSUB, pokud je připojení ve stavu uvedení do klidového stavu.

ObjectName (MQCHAR48) pro MQSD

Jedná se o název objektu tématu, jak je definován v lokálním správci front.

Název může obsahovat následující znaky:

- Velká písmena abecedy (A až Z)
- Malá písmena abecedy (a až z)
- Číselné číslice (0 až 9)
- Tečka (.), dopředné lomítko (/), podtržítko (_), procento (%)

Název nesmí obsahovat úvodní ani vložené mezery, ale může obsahovat koncové mezery. Použijte znak null, abyste označili konec důležitých dat v názvu; hodnota null a všechny následující znaky jsou považovány za mezery. V označených prostředích platí následující omezení:

- V systémech, které používají EBCDIC Katakana, nelze použít malá písmena.
- V systému z/OS:
  - Vyvarujte se názvů, které začínají nebo končí podtržítkem; nemohou být zpracovány operacemi a ovládacími panely.
  - Znak procenta má speciální význam pro RACF. Pokud se RACF používá jako externí správce zabezpečení, názvy nesmí obsahovat procenta. Pokud ano, nejsou tyto názvy zahrnuty do žádných kontrol zabezpečení, když se používají generické profily RACF .
- V systému IBM imusí být názvy obsahující malá písmena, dopředné lomítko nebo procenta uzavřeny v uvozovkách, jsou-li zadány v příkazech. Tyto uvozovky nesmí být uvedeny pro názvy, které se vyskytují jako pole ve strukturách nebo jako parametry ve voláních.

ObjectName se používá k vytvoření úplného názvu tématu.

Úplný název tématu lze sestavit ze dvou různých polí: *ObjectName* a *ObjectString*. Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat.

Pokud objekt identifikovaný polem *ObjectName* nelze nalézt, volání selže s kódem příčiny MQRC_UNKNOWN_OBJECT_NAME, i když je v souboru *ObjectString*uveden řetězec.

Při návratu z volání MQSUB pomocí volby MQSO_RESUME se toto pole nezměnilo.

Délka tohoto pole je dána hodnotou MQ_TOPIC_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, nelze název objektu tématu, který je přihlášen k odběru, změnit. Toto pole a pole *ObjectString* lze vynechat. Pokud jsou poskytnuty, musí být vyřešeny na stejný úplný název tématu. Pokud ne, volání selže s MQRC_TOPIC_NOT_ALTERABLE.

AlternateUserID (MQCHAR12) pro MQSD

Zadáte-li MQSO_ALTERNATE_USER_AUTHORITY, bude toto pole obsahovat alternativní identifikátor uživatele, který se používá ke kontrole autorizace pro odběr a pro výstup do cílové fronty (určené v parametru **Hobj** volání MQSUB) namísto identifikátoru uživatele, pod kterým je aplikace aktuálně spuštěna.

V případě úspěchu je identifikátor uživatele uvedený v tomto poli zaznamenán jako identifikátor uživatele vlastnící odběr namísto identifikátoru uživatele, pod kterým je aplikace aktuálně spuštěna.

Je-li zadána volba MQSO_ALTERNATE_USER_AUTHORITY a toto pole je prázdné až do prvního znaku s hodnotou Null nebo do konce pole, může být odběr úspěšný pouze v případě, že pro přihlášení k odběru tohoto tématu s určenými volbami nebo cílovou frontou pro výstup není potřebná žádná autorizace uživatele.

Není-li zadána volba MQSO_ALTERNATE_USER_AUTHORITY, bude toto pole ignorováno.

V označených prostředích existují následující rozdíly:

- V systému z/OSse ke kontrole autorizace pro odběr používá pouze prvních 8 znaků ID AlternateUser. Avšak aktuální identifikátor uživatele musí být autorizován k uvedení tohoto konkrétního alternativního identifikátoru uživatele; pro tuto kontrolu se použije všech 12 znaků alternativního identifikátoru uživatele. Identifikátor uživatele musí obsahovat pouze znaky povolené externím správcem zabezpečení.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nezměněno.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 12 prázdných znaků v jiných programovacích jazycích.

AlternateSecurityID (MQBYTE40) pro MQSD

Jedná se o identifikátor zabezpečení, který je předán s ID AlternateUserautorizační službě, aby bylo možné provést odpovídající kontroly autorizace.

AlternateSecurityId se používá pouze v případě, že je uvedeno MQSO_ALTERNATE_USER_AUTHORITY a pole AlternateUserId není zcela prázdné až do prvního znaku null nebo konce pole.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nezměněno.

Další informace viz popis souboru “AlternateSecurityId (MQBYTE40) pro MQOD” na stránce 493 v datovém typu MQOD.

SubExpiry (MQLONG) pro MQSD

Jedná se o dobu vyjádřenou v desetinách sekundy, po jejímž uplynutí vyprší platnost odběru. Po uplynutí tohoto intervalu nebudou tomuto odběru odpovídat žádná další publikování. Jakmile vyprší platnost odběru, publikování již nebudou odesílána do fronty. Publikace, které již existují, však nejsou nijak ovlivněny. *SubExpiry* nemá žádný vliv na vypršení platnosti publikování.

Je rozpoznána následující speciální hodnota:

MQEI_UNLIMITED
Předplatné má neomezenou dobu vypršení platnosti.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, lze vypršení platnosti odběru změnit.

Při návratu z volání MQSUB s použitím volby MQSO_RESUME je toto pole nastaveno na původní vypršení platnosti odběru, nikoli na zbývající dobu vypršení platnosti.

ObjectString (MQCHARV) pro MQSD

Jedná se o dlouhý název objektu, který se má použít.

ObjectString se používá k vytvoření úplného názvu tématu.

Úplný název tématu lze sestavit ze dvou různých polí: *ObjectName* a *ObjectString*. Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat.

Maximální délka *ObjectString* je 10240.

Není-li parametr *ObjectString* zadán správně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_OBJECT_STRING_ERROR.

Toto je vstupní pole. Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQCHARV.

Pokud jsou v souboru *ObjectString* zástupné znaky, lze interpretaci těchto zástupných znaků řídit pomocí voleb zástupných znaků uvedených v poli Volby na disku MQSD.

Při návratu z volání MQSUB pomocí volby MQSO_RESUME se toto pole nezměnilo. Je-li poskytnuta vyrovnávací paměť, vrátí se v poli *ResObjectString* úplný název použitého tématu.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, nelze dlouhý název objektu tématu, který je přihlášen k odběru, změnit. Toto pole a pole *ObjectName* lze vynechat. Pokud jsou poskytnuty, musí se přeložit na stejný úplný název tématu, jinak volání selže s MQRC_TOPIC_NOT_ALTERABLE.

SubName (MQCHARV) pro MQSD

Tato volba určuje název odběru. Toto pole je povinné pouze v případě, že parametr *Options* určuje volbu MQSO_TRVALÝ, ale pokud je k dispozici, bude jej používat i správce front pro volbu MQSO_NON_TRVALÝ.

Je-li zadána volba *SubName* , musí být v rámci správce front jedinečná, protože se jedná o metodu používanou k identifikaci odběru.

Maximální délka *SubName* je 10240.

Toto pole slouží dvěma účelům. V případě odběru MQSO_TRVALÝ můžete toto pole použít k identifikaci odběru, abyste jej mohli obnovit po jeho vytvoření, pokud jste buď zavřeli manipulátor pro odběr (pomocí volby MQCO_KEEP_SUB), nebo jste byli odpojeni od správce front. To se provádí pomocí volání MQSUB s volbou MQSO_RESUME. Zobrazí se také v administrativním zobrazení odběrů v poli SUBID v části DISPLAY SBSTATUS.

Je-li parametr *SubName* zadán nesprávně, je v závislosti na popisu použití struktury MQCHARV vynechán v případě, že je vyžadována (tj. *SubName*). *VSLength* je nula), nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_SUB_NAME_ERROR.

Toto je vstupní pole. Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQCHARV.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, název odběru nelze změnit, protože se jedná o identifikační pole použité k vyhledání odkazovaného odběru. Při výstupu volání MQSUB s volbou MQSO_RESUME se nezmění.

SubUser(MQCHARV) pro MQSD

Tato volba určuje uživatelská data odběru. Data poskytnutá v rámci odběru v tomto poli budou zahrnuta jako vlastnost datové zprávy MQSubUserpro každé publikování odeslané v rámci tohoto odběru.

Maximální délka *SubUserData* je 10240.

Pokud je parametr *SubUserData* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV nebo pokud překračuje maximální délku, volání se nezdaří s kódem příčiny MQRC_SUB_USER_DATA_ERROR.

Toto je vstupní pole. Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQCHARV.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, lze uživatelská data odběru změnit.

Toto pole délky proměnné je vráceno na výstupu volání MQSUB pomocí volby MQSO_RESUME, pokud je poskytnuta vyrovnávací paměť a v souboru *VSBufLen*je kladná délka vyrovnávací paměti. Není-li pro volání poskytnuta žádná vyrovnávací paměť, bude do pole *VSLength* v tabulce MQCHARV vrácena pouze délka data uživatele odběru. Pokud je poskytnutá vyrovnávací paměť menší než prostor požadovaný pro vrácení pole, vrátí se v poskytnuté vyrovnávací paměti pouze *VSBufLen* bajtů.

SubCorrelId (MQBYTE24) pro MQSD

Toto pole obsahuje identifikátor korelace společný pro všechna publikování odpovídající tomuto odběru.

Upozornění: Identifikátor korelace lze předávat pouze mezi správci front v klastru publikování/odběru, nikoli v hierarchii.

Všechna publikování odeslaná za účelem shody s tímto odběrem obsahují tento identifikátor korelace v deskriptoru zprávy. Pokud více odběrů získá publikování ze stejné fronty, použití příkazu MQGET podle identifikátoru korelace umožní získat pouze publikování pro konkrétní odběr. Tento identifikátor korelace může být generován správcem front nebo uživatelem.

Pokud není zadána volba MQSO_SET_CORREL_ID, je identifikátor korelace generován správcem front a toto pole je výstupní pole obsahující identifikátor korelace, který bude nastaven v každé zprávě publikované pro tento odběr. Generovaný identifikátor korelace se skládá ze 4bajtového identifikátoru produktu (AMQX nebo CSQM v ASCII nebo EBCDIC) následovaného specifickou implementací jedinečného řetězce.

Je-li zadána volba MQSO_SET_CORREL_ID, je identifikátor korelace generován uživatelem a toto pole je vstupní pole obsahující identifikátor korelace, který má být nastaven v každém publikování pro tento odběr. V tomto případě, pokud pole obsahuje MQCI_NONE, je identifikátorem korelace, který je nastaven v každé zprávě publikované pro tento odběr, identifikátor korelace vytvořený původním vložením zprávy.

Je-li zadána volba MQSO_GROUP_SUB a zadaný identifikátor korelace je stejný jako existující seskupený odběr používající stejnou frontu a překrývající se řetězec tématu, bude s kopií publikování poskytnut pouze nejvýznamnější odběr ve skupině.

Délka tohoto pole je dána hodnotou MQ_CORREL_ID_LENGTH. Počáteční hodnota tohoto pole je MQCI_NONE.

Pokud měníte existující odběr pomocí volby MQSO_ALTER a toto pole je vstupní pole, lze změnit identifikátor korelace odběru, pokud se nejedná o seskupený odběr, tj. pokud byl vytvořen pomocí volby MQSO_GROUP_SUB. V takovém případě nelze změnit identifikátor korelace odběru.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nastaveno na aktuální identifikátor korelace pro odběr.

PubPriority (MQLONG) pro MQSD

Jedná se o hodnotu, která bude v poli *Priority* deskriptoru zpráv (MQMD) všech publikačních zpráv odpovídajících tomuto odběru. Další informace o poli *Priority* v deskriptoru MQMD viz “Priorita (MQLONG) pro MQMD” na stránce 453.

Hodnota musí být větší nebo rovna nule; nula je nejnižší priorita. Lze také použít následující speciální hodnoty:

MQPRI_PRIORITY_AS_Q_DEF

Pokud je do pole *Hobj* ve volání MQSUB zadána fronta odběru a nejedná se o spravovaný manipulátor, bude priorita zprávy převzata z atributu **DefPriority** této fronty. Pokud se jedná o frontu klastru nebo pokud je v cestě k rozlišení názvu fronty více než jedna definice, bude priorita určena při vložení zprávy publikování do fronty, jak je popsáno v tématu "Priorita (MQLONG) pro MQMD" na stránce 453.

Pokud volání MQSUB používá spravovaný popisovač, priorita zprávy je převzata z atributu **DefPriority** modelové fronty přidružené k odebíranému tématu.

MQPRI_PRIORITY_AS_PUBLISHED

Prioritou zprávy je priorita původní publikace. Toto je počáteční hodnota pole.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, lze změnit hodnotu *Priority* všech budoucích zpráv publikování.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nastaveno na aktuální prioritu používanou pro odběr.

PubAccountingToken (MQBYTE32) pro MQSD

Jedná se o hodnotu, která bude v poli *AccountingToken* deskriptoru zpráv (MQMD) všech publikačních zpráv odpovídajících tomuto odběru. *AccountingToken* je součástí kontextu identity zprávy. Další informace o kontextu zprávy viz Kontext zprávy. Další informace o poli *AccountingToken* v deskriptoru MQMD viz "AccountingToken (MQBYTE32) pro MQMD" na stránce 461

Pro pole *PubAccountingToken* můžete použít následující speciální hodnotu:

MQACT_NONE

Není uveden žádný token evidence.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQACT_NONE_ARRAY, která má stejnou hodnotu jako MQACT_NONE, ale je to pole znaků namísto řetězce.

Pokud není zadána volba MQSO_SET_IDENTITY_CONTEXT, je token evidence generován správcem front jako výchozí informace o kontextu a toto pole je výstupní pole obsahující položku *AccountingToken* , která bude nastavena v každé zprávě publikované pro tento odběr.

Je-li zadána volba MQSO_SET_IDENTITY_CONTEXT, je token evidence generován uživatelem a toto pole je vstupní pole obsahující položku *AccountingToken* , která má být nastavena v každém publikování pro tento odběr.

Délka tohoto pole je dána hodnotou MQ_ACCOUNTING_TOKEN_LENGTH. Počáteční hodnota tohoto pole je MQACT_NONE.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, můžete změnit hodnotu *AccountingToken* v budoucích zprávách publikování.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nastaveno na aktuální hodnotu *AccountingToken* používanou pro odběr.

PubApplIdentityData (MQCHAR32) pro MQSD

Jedná se o hodnotu, která je v poli *ApplIdentityData* deskriptoru zpráv (MQMD) všech zpráv publikování odpovídajících tomuto odběru. *ApplIdentityData* je součástí kontextu identity zprávy. Další informace o kontextu zprávy viz Kontext zprávy. Další informace o poli *ApplIdentityData* v deskriptoru MQMD viz "ApplIdentity-data (MQCHAR32) pro MQMD" na stránce 462

Pokud není zadána volba MQSO_SET_IDENTITY_CONTEXT, je hodnota *ApplIdentityData* , která je nastavena v každé zprávě publikované pro tento odběr, prázdná, jako výchozí informace o kontextu.

Je-li zadána volba MQSO_SET_IDENTITY_CONTEXT, je objekt *PubApplIdentityData* generován uživatelem a toto pole je vstupním polem obsahujícím soubor *ApplIdentityData* , který má být nastaven v každém publikování pro tento odběr.

Délka tohoto pole je dána hodnotou MQ_APPL_IDENTITY_DATA_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 32 prázdných znaků v jiných programovacích jazycích.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, lze změnit hodnotu *ApplIdentityData* všech budoucích zpráv publikování.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nastaveno na aktuální hodnotu *ApplIdentityData* používanou pro odběr.

SelectionString (MQCHARV) pro MQSD

Jedná se o řetězec používaný k zadání kritérií výběru používaných při přihlašování k odběru zpráv z tématu.

Toto pole délky proměnné bude vráceno na výstupu volání MQSUB s použitím volby MQSO_RESUME, pokud je poskytnuta vyrovnávací paměť, a také je zde kladná délka vyrovnávací paměti v VSBufSize. Není-li pro volání poskytnuta žádná vyrovnávací paměť, bude v poli VSLength hodnoty MQCHARV vrácena pouze délka řetězce výběru. Je-li poskytnutá vyrovnávací paměť menší než prostor požadovaný pro vrácení pole, jsou v poskytnuté vyrovnávací paměti vráceny pouze bajty VSBufSize .

Pokud je parametr *SelectionString* zadán nesprávně, podle popisu způsobu použití struktury “MQCHARV-Řetězec délky proměnné” na stránce 296 , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_SELECTION_STRING_ERROR.

Použití SelectionString je popsáno v části Selektory.

SubLevel (MQLONG) pro MQSD

Toto je úroveň přidružená k odběru. Publikování jsou pro tento odběr doručena pouze v případě, že jsou v sadě odběrů s nejvyšší hodnotou SubLevel menší nebo rovnou hodnotě PubLevel použité v době publikování. Pokud však bylo publikování zachováno, není již k dispozici pro odběratele na vyšších úrovních, protože je znovu publikována na PubLevel 1.

Hodnota musí být v rozsahu od 0 do 9. Nula je nejnižší úroveň.

Počáteční hodnota tohoto pole je 1.

Další informace naleznete v tématu Zachycení publikací.

Pokud měníte existující odběr pomocí volby MQSO_ALTER, nelze SubLevel změnit.

Kombinace SubLevel s hodnotou větší než 1 s volbou MQSO_PUBLICATIONS_ON_REQUEST není povolena.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nastaveno na aktuální úroveň používanou pro odběr.

ResObjectŘetězec (MQCHARV) pro MQSD

Jedná se o dlouhý název objektu poté, co správce front interpretuje název uvedený v souboru *ObjectName*.

Pokud je v souboru *ObjectString* uveden dlouhý název objektu a v souboru *ObjectName*není uveden žádný název, hodnota vrácená v tomto poli je stejná jako v souboru *ObjectString*.

Pokud je toto pole vynecháno (tj. ResObjectString.VSBufSize je nula), pak se *ResObjectString* nevrátí, ale délka je vrácena v ResObjectString.VSLength. Je-li délka kratší než úplný řetězec ResObject, je oříznut a vrací tolik znaků, kolik je nejvíce vpravo, kolik se vejde do zadané délky.

Pokud je parametr *ResObjectString* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_RES_OBJECT_STRING_ERROR.

MQSMPO-Nastavení voleb vlastností zprávy

Struktura **MQSMPO** umožňuje aplikacím určit volby, které řídí způsob nastavení vlastností zpráv. Struktura je vstupní parametr volání **MQSETMP** .

Dostupnost

Všechny systémy IBM MQ a klienty IBM MQ .

Znaková sada a kódování

Data v souboru **MQSMPO** musí být ve znakové sadě aplikace a kódování aplikace (**MQENC_NATIVE**).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 531. Pole v objektu MQSMPO

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQSMPO_STRUC_ID	'SMPO'
Verze (číslo verze struktury)	MQSMPO_VERSION_1	1
Volby (volby)	MQSMPO_NONE	0
ValueEncoding (kódování hodnoty vlastnosti)	MQENC_NATIVE	Závisí na prostředí
ValueCCSID (znaková sada hodnoty vlastnosti)	MQCCSI_APPL	-3

Notes:

1. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyku C se jedná o proměnnou makra.MQSMPO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQSMPO MySMPO = {MQSMPO_DEFAULT};
```

Deklarace jazyka

Deklarace jazyka C pro objekt MQSMPO

```
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 */
};
```

Deklarace jazyka COBOL pro objekt MQSMPO

```
**   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.
```

Deklarace PL/I pro objekt MQSMPO

```
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 */
```

Deklarace High Level Assembler pro objekt MQSMPO

```
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)
```

StrucId (MQCHAR4) pro MQSMPO

Jedná se o identifikátor struktury struktury voleb nastavení vlastností zprávy. Vždy se jedná o vstupní pole. Jeho hodnota je MQSMPO_STRUC_ID.

Hodnota musí být:

MQSMPO_STRUC_ID

Identifikátor pro strukturu voleb nastavení vlastností zprávy.

Pro programovací jazyk C je definována také konstanta **MQSMPO_STRUC_ID_ARRAY** . Má stejnou hodnotu jako **MQSMPO_STRUC_ID**, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro objekt MQSMPO

Toto je číslo verze struktury; hodnota musí být:

MQSMPO_VERSION_1
Version-1 nastavte strukturu voleb vlastností zprávy.

Následující konstanta určuje číslo verze aktuální verze:

MQSMPO_CURRENT_VERSION
Aktuální verze struktury vlastností nastavení zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQSMPO_VERSION_1**.

Volby (MQLONG) pro MQSMPO

Volby umístění

Následující volby se vztahují k relativnímu umístění vlastnosti v porovnání s kurzorem vlastnosti:

MQSMPO_SET_FIRST
Nastaví hodnotu první vlastnosti, která odpovídá zadanému názvu, nebo pokud neexistuje, přidá novou vlastnost za všechny ostatní vlastnosti s odpovídající hierarchií.

MQSMPO_SET_PROP_UNDER_CURSOR
Nastaví hodnotu vlastnosti, na kterou ukazuje kurzor vlastnosti. Vlastnost, na kterou ukazuje kurzor vlastnosti, je ta, která byla naposledy zjišťována pomocí volby MQIMPO_INQ_FIRST nebo MQIMPO_INQ_NEXT.

Kurzor vlastnosti je resetován při opětovném použití manipulátoru zprávy ve volání MQGET nebo při zadání manipulátoru zprávy v poli *MsgHandle* struktury MQGMO nebo MQPMO ve volání MQPUT.

Pokud je tato volba použita v případě, že kurzor vlastnosti dosud nebyl vytvořen nebo pokud byl ukazatel vlastnosti na kurzor vlastnosti odstraněn, volání selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_PROPERTY_NOT_AVAILABLE.

MQSMPO_SET_PROP_BEFORE_CURSOR
Nastaví novou vlastnost před vlastnost, na kterou ukazuje kurzor vlastnosti. Vlastnost, na kterou ukazuje kurzor vlastnosti, je ta, která byla naposledy zjišťována pomocí volby MQIMPO_INQ_FIRST nebo MQIMPO_INQ_NEXT.

Kurzor vlastnosti je resetován při opětovném použití manipulátoru zprávy ve volání MQGET nebo při zadání manipulátoru zprávy v poli *MsgHandle* struktury MQGMO nebo MQPMO ve volání MQPUT.

Pokud je tato volba použita v případě, že kurzor vlastnosti dosud nebyl vytvořen nebo pokud byl ukazatel vlastnosti na kurzor vlastnosti odstraněn, volání selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_PROPERTY_NOT_AVAILABLE.

MQSMPO_SET_PROP_AFTER_CURSOR
Nastaví novou vlastnost za vlastnost, na kterou ukazuje kurzor vlastnosti. Vlastnost, na kterou ukazuje kurzor vlastnosti, je ta, která byla naposledy zjišťována pomocí volby MQIMPO_INQ_FIRST nebo MQIMPO_INQ_NEXT.

Kurzor vlastnosti je resetován při opětovném použití manipulátoru zprávy ve volání MQGET nebo při zadání manipulátoru zprávy v poli *MsgHandle* struktury MQGMO nebo MQPMO ve volání MQPUT.

Pokud je tato volba použita v případě, že kurzor vlastnosti dosud nebyl vytvořen nebo pokud byl ukazatel vlastnosti na kurzor vlastnosti odstraněn, volání selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_PROPERTY_NOT_AVAILABLE.

MQSMPO_APPEND_PROPERTY
Způsobí přidání nové vlastnosti za všechny ostatní vlastnosti s odpovídající hierarchií. Pokud existuje alespoň jedna vlastnost, která odpovídá zadanému názvu, přidá se nová vlastnost na konec za konec tohoto seznamu vlastností.

Tato volba umožňuje vytvoření seznamu vlastností se stejným názvem.

Pokud nepotřebujete žádnou z popsaných voleb, použijte následující volbu:

MQSMPO_NONE
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSMPO_SET_FIRST.

ValueEncoding (MQLONG) pro objekt MQSMPO

Kódování hodnoty vlastnosti, která má být nastavena, pokud je hodnota číselná.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQENC_NATIVE**.

ValueCCSID (MQLONG) pro MQSMPO

Znaková sada hodnoty vlastnosti, která má být nastavena, pokud je hodnota znakový řetězec.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQCCSI_APPL**.

MQSRO-Volby požadavku na odběr

Struktura MQSRO umožňuje aplikaci určit volby, které řídí způsob provedení požadavku na odběr. Struktura je vstupní/výstupní parametr volání MQSUBRQ.

Dostupnost

Struktura MQSRO je k dispozici na následujících platformách:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows
- 

z/OS

a pro systém IBM MQ MQI clients připojený k těmto systémům.

Verze

Aktuální verze MQSRO je MQSRO_VERSION_1.

Znaková sada a kódování

Data v MQSRO musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE. Pokud je však aplikace spuštěna jako klient MQI produktu MQ , musí být struktura ve znakové sadě a kódování klienta.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQSRO_STRUC_ID	'SRO¬'
Verze (číslo verze struktury)	MQSRO_VERSION_1	1
Volby (volby)	MQSRO_NONE	0
NumPubs (počet publikování)	Není	0

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQSRO_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQSRO MySRO = {MQSRO_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQSRO

```
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 */
};
```

Deklarace jazyka COBOL pro MQSRO

```
** 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.
```

Deklarace PL/I pro MQSRO

```
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 */
```

Deklarace High Level Assembler pro MQSRO

```
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)
```

StrucId (MQCHAR4) pro MQSRO

Jedná se o identifikátor struktury struktury voleb požadavku na odběr. Vždy se jedná o vstupní pole. Jeho hodnota je MQSRO_STRUC_ID.

Hodnota musí být:

MQSRO_STRUC_ID
Identifikátor struktury voleb požadavku na odběr.

Pro programovací jazyk C je definována také konstanta MQSRO_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQSRO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQSRO

Toto je číslo verze struktury; hodnota musí být:

MQSRO_VERSION_1
Version-1 Struktura voleb požadavku na odběr.

Následující konstanta určuje číslo verze aktuální verze:

MQSRO_CURRENT_VERSION
Aktuální verze struktury voleb požadavku na odběr.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSRO_VERSION_1.

Volby (MQLONG) pro MQSRO

Musí být zadána jedna z následujících voleb. Lze zadat pouze jednu volbu.

MQSRO_FAIL_IF_QUIESCING

Volání MQSUBRQ se nezdaří, pokud je správce front ve stavu uvedení do klidového stavu. V systému z/OSpro aplikaci CICS nebo IMS tato volba také vynutí selhání volání MQSUBRQ, pokud je připojení v klidovém stavu.

Výchozí volba: Pokud dříve popsaná volba není povinná, musí být použita následující volba:

MQSRO_NONE

Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

MQSRO_NONE pomáhá s dokumentací programu. Ačkoli není zamýšleno, aby byla tato volba použita s ostatními, protože její hodnota je nula, nelze toto použití zjistit.

NumPubs (MQLONG) pro MQSRO

Jedná se o výstupní pole vrácené aplikaci, které určuje počet publikování odeslaných do fronty odběrů v důsledku tohoto volání. Ačkoli byl tento počet publikací odeslán jako výsledek tohoto volání, není zde žádná záruka, že tento počet zpráv bude k dispozici pro získání aplikace, zejména pokud se jedná o dočasné zprávy.

Pokud téma obsahovalo zástupný znak, může existovat více než jedno publikování. Pokud v řetězci tématu nebyly při vytvoření odběru reprezentovaného *Hsub* přítomny žádné zástupné znaky, bude v důsledku tohoto volání odesláno nejvýše jedno publikování.

MQSTS-Struktura vykazování stavu

Struktura MQSTS je výstupní parametr z příkazu MQSTAT. Příkaz MQSTAT se používá k načtení informací o stavu. Tyto informace jsou vráceny ve struktuře MQSTS.

Znaková sada a kódování

Znaková data v MQSTS jsou ve znakové sadě lokálního správce front. Tato hodnota je dána atributem správce front *CodedCharSetId* . Číselná data v MQSTS jsou v nativním kódování počítače; toto je dáno hodnotou *Kódování*.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 532. Pole v MQSTS

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQSTS_STRUC_ID	'STAT¬'
Verze (číslo verze struktury)	MQSTS_VERSION_1	1
CompCode (kód dokončení první chyby)	MQCC_OK	0
Příčina (kód příčiny první chyby)	MQRC_NONE	0
PutSuccessPočet (počet úspěšných asynchronních volání vložení)	Není	0
PutWarningCount (počet asynchronních volání vložení, která měla varování)	Není	0
PutFailurePočet (počet nezdařených asynchronních volání vložení)	Není	0
ObjectType (typ selhávajícího objektu)	MQOT_Q	1
ObjectName (název selhávajícího objektu)	Není	Prázdný řetězec nebo mezery
ObjectQMgrNázev (název správce front, který vlastní objekt, který selhal.	Není	Prázdný řetězec nebo mezery
ResolvedObjectName (přeložený název cílové fronty)	Není	Prázdný řetězec nebo mezery
ResolvedQMgrNázev (vyřešený název správce cílových front)	Není	Prázdný řetězec nebo mezery
Poznámka: Zbývající pole jsou ignorována, pokud je verze menší než MQSTS_VERSION_2.		
ObjectString (dlouhý název selhávajícího objektu)	VÝCHOZÍ	{NULL,0,0,0,-3}
SubName (název odběru, u kterého došlo k selhání odběru)	VÝCHOZÍ	{NULL,0,0,0,-3}
OpenOptions (volby otevření přidružené k selhání)	Není	0
SubOptions(volby odběru přidružené k selhání)	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C obsahuje proměnná makra MQSTS_DEFAULT hodnoty uvedené v tabulce. Lze jej použít následujícím způsobem k poskytnutí počátečních hodnot pro pole ve struktuře:

```
MQSTS MySTS = {MQSTS_DEFAULT};
```

Deklarace jazyka

Deklarace C pro MQSTS

```
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 */
};
```

Deklarace jazyka COBOL pro MQSTS

```
      ** 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 **
```

Deklarace PL/I pro MQSTS

```
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 */
```

Deklarace High Level Assembler pro MQSTS

```
    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)
```

Související odkazy

“MQSTAT-Načtení informací o stavu” na stránce 786
Pomocí volání MQSTAT načtěte informace o stavu. Typ vrácených informací o stavu je určen hodnotou typu uvedenou ve volání.

StrucId (MQCHAR4) pro MQSTS

Jedná se o identifikátor struktury struktury hlášení stavu. Vždy se jedná o vstupní pole. Jeho hodnota je MQSTS_STRUC_ID.

Hodnota musí být:

MQSTS_STRUC_ID
Identifikátor pro strukturu vykazování stavu.

Pro programovací jazyk C je definována také konstanta MQSTS_STRUC_ID_ARRAY . Má stejnou hodnotu jako MQSTS_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQSTS

Číslo verze struktury.

Hodnota musí být buď:

MQSTS_VERSION_1
Struktura vykazování stavu verze 1.

MQSTS_VERSION_2
Struktura vykazování stavu verze 2.

Následující konstanta určuje číslo verze aktuální verze:

MQSTS_CURRENT_VERSION
Aktuální verze struktury hlášení stavu. Aktuální verze je MQSTS_VERSION_2.

Version je vždy vstupní pole. Jeho počáteční hodnota je MQSTS_VERSION_1.

CompCode (MQLONG) pro MQSTS

Kód dokončení operace, která se vykazuje.

Interpretace parametru CompCode závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Jedná se o kód dokončení, který je výsledkem předchozí asynchronní operace vložení pro objekt určený v souboru ObjectName.

MQSTAT_TYPE_RECONNECTION

Pokud se připojení znovu připojuje nebo se mu nepodařilo znovu připojit, jedná se o kód dokončení, který způsobil, že se připojení začalo znovu připojovat.

Pokud je připojení momentálně připojeno, hodnota je MQCC_OK.

MQSTAT_TYPE_RECONNECTION_ERROR

Pokud se připojení nepodařilo znovu připojit, jedná se o kód dokončení, který způsobil selhání opětovného připojení.

Pokud je připojení aktuálně připojeno nebo se znovu připojuje, hodnota je MQCC_OK.

CompCode je vždy výstupní pole. Jeho počáteční hodnota je MQCC_OK.

Příčina (MQLONG) pro MQSTS
Kód příčiny operace, pro kterou se provádí hlášení.

Interpretace parametru Reason závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Jedná se o kód příčiny, který je výsledkem předchozí asynchronní operace vložení pro objekt určený v souboru ObjectName.

MQSTAT_TYPE_RECONNECTION

Pokud se připojení znovu připojuje nebo se mu nepodařilo znovu připojit, jedná se o kód příčiny, který způsobil, že se opětovné připojení začalo znovu připojovat.

Pokud je připojení momentálně připojeno, hodnota je MQRC_NONE.

MQSTAT_TYPE_RECONNECTION_ERROR

Pokud se připojení nepodařilo znovu připojit, jedná se o kód příčiny, který způsobil selhání opětovného připojení.

Pokud je připojení aktuálně připojeno nebo se znovu připojuje, hodnota je MQRC_NONE.

Reason je výstupní pole. Jeho počáteční hodnota je MQRC_NONE.

PutSuccess(MQLONG) pro MQSTS
Počet operací asynchronního vložení, které byly úspěšné.

Hodnota parametru PutSuccessCount závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Počet asynchronních operací vložení do objektu s názvem ve struktuře MQSTS , které byly dokončeny pomocí MQCC_OK.

MQSTAT_TYPE_RECONNECTION

Nula.

MQSTAT_TYPE_RECONNECTION_ERROR

Nula.

PutSuccessCount je výstupní pole. Jeho počáteční hodnota je nula.

PutWarningPočet (MQLONG) pro MQSTS
Počet asynchronních operací vložení, které skončily s varováním.

Hodnota parametru PutWarningCount závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Počet asynchronních operací vložení do objektu s názvem ve struktuře MQSTS , které byly dokončeny pomocí MQCC_WARNING.

MQSTAT_TYPE_RECONNECTION

Nula.

MQSTAT_TYPE_RECONNECTION_ERROR

Nula.

PutWarningCount je výstupní pole. Jeho počáteční hodnota je nula.

PutFailurePočet (MQLONG) pro MQSTS
Počet asynchronních operací vložení, které se nezdařily.

Hodnota parametru PutFailureCount závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Počet asynchronních operací vložení do objektu s názvem ve struktuře MQSTS , které byly dokončeny pomocí MQCC_FAILED.

MQSTAT_TYPE_RECONNECTION

Nula.

MQSTAT_TYPE_RECONNECTION_ERROR

Nula.

PutFailureCount je výstupní pole. Jeho počáteční hodnota je nula.

ObjectType (MQLONG) pro MQSTS

Typ objektu uvedeného v souboru *ObjectName* , který je vykazován.

Možné hodnoty ObjectType jsou uvedeny v seznamu “MQOT_ * (Typy objektů a rozšířené typy objektů)” na stránce 164.

ObjectType je výstupní pole. Jeho počáteční hodnota je MQOT_Q.

ObjectName (MQCHAR48) pro MQSTS

Název objektu, který je vykazován.

Interpretace parametru ObjectName závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Jedná se o název fronty nebo tématu použitého v operaci vložení, jehož selhání je hlášeno v polích *CompCode* a *Reason* ve struktuře MQSTS .

MQSTAT_TYPE_RECONNECTION

Pokud se připojení znovu připojuje, jedná se o název správce front přidruženého k připojení.

MQSTAT_TYPE_RECONNECTION_ERROR

Pokud se připojení nepodařilo znovu připojit, jedná se o název objektu, který způsobil selhání opětovného připojení. Příčina poruchy je hlášena v polích *CompCode* a *Reason* ve struktuře MQSTS .

ObjectName je výstupní pole. Jeho počáteční hodnota je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ObjectQMgrNázev (MQCHAR48) pro MQSTS

Název správce front, pro kterého je sestavováno hlášení.

Interpretace parametru ObjectQMgrName závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Jedná se o název správce front, v němž je definován objekt *ObjectName* . Název, který je zcela prázdný až do prvního znaku null nebo do konce pole, označuje správce front, ke kterému je aplikace připojena (lokální správce front).

MQSTAT_TYPE_RECONNECTION

Multi

Pole **ObjectQMgrName** obsahuje název správce front, pro kterého je vyžadováno opětovné připojení, nebo je prázdné, pokud není zadán žádný správce front. Pokud je to možné, klient se pokusí znovu připojit ke správci front s tímto názvem.

z/OS

Prázdné.

MQSTAT_TYPE_RECONNECTION_ERROR

Pokud se připojení nepodařilo znovu připojit, jedná se o název objektu, který způsobil selhání opětovného připojení. Příčina poruchy je hlášena v polích *CompCode* a *Reason* ve struktuře MQSTS .

ObjectQMgrName je výstupní pole. Jeho hodnota je řetězec null v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ResolvedObjectNázev (MQCHAR48) pro MQSTS

Název objektu uvedeného v souboru *ObjectName* po vyřešení názvu lokálním správcem front.

Interpretace parametru ResolvedObjectName závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

ResolvedObjectName je název objektu uvedeného v souboru *ObjectName* poté, co lokální správce front přeloží název. Vrácený název je název objektu, který existuje ve správci front identifikovaném pomocí *ResolvedQMgrName*.

MQSTAT_TYPE_RECONNECTION

Prázdné.

MQSTAT_TYPE_RECONNECTION_ERROR

Prázdné.

ResolvedObjectName je výstupní pole. Jeho počáteční hodnota je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ResolvedQMgrNázev (MQCHAR48) pro MQSTS

Název cílového správce front po vyřešení názvu lokálního správce front.

Interpretace parametru ResolvedQMgrName závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

ResolvedQMgrName je název cílového správce front poté, co lokální správce front přeloží název. Vrácený název je název správce front, který vlastní objekt identifikovaný jako *ResolvedObjectName*. *ResolvedQMgrName* může být název lokálního správce front.

MQSTAT_TYPE_RECONNECTION

Prázdné.

MQSTAT_TYPE_RECONNECTION_ERROR

Prázdné.

ResolvedQMgrName je vždy výstupní pole. Jeho počáteční hodnota je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ObjectString (MQCHARV) pro MQSTS

Dlouhý název objektu, pro který došlo k selhání. K dispozici pouze ve verzi 2 produktu MQSTS nebo vyšší.

Interpretace parametru ObjectString závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Jedná se o dlouhý název objektu fronty nebo tématu použitého v operaci MQPUT , která se nezdařila.

MQSTAT_TYPE_RECONNECTION

Řetězec s nulovou délkou

MQSTAT_TYPE_RECONNECTION_ERROR

Jedná se o dlouhý název objektu, který způsobil selhání opětovného připojení.

ObjectString je výstupní pole. Jeho počáteční hodnota je řetězec s nulovou délkou.

SubName (MQCHARV) pro MQSTS

Název selhávajícího odběru. K dispozici pouze ve verzi 2 produktu MQSTS nebo vyšší.

Interpretace parametru SubName závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Řetězec s nulovou délkou.

MQSTAT_TYPE_RECONNECTION

Řetězec s nulovou délkou.

MQSTAT_TYPE_RECONNECTION_ERROR

Název odběru, který způsobil selhání opětovného připojení. Pokud není k dispozici žádný název odběru nebo pokud selhání nesouvisí s odběrem, jedná se o řetězec s nulovou délkou.

SubName je výstupní pole. Jeho počáteční hodnota je řetězec s nulovou délkou.

OpenOptions (MQLONG) pro MQSTS

OpenOptions použitý k otevření objektu, pro který se má hlášení provést. K dispozici pouze ve verzi 2 produktu MQSTS nebo vyšší.

Hodnota parametru OpenOptions závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Nula.

MQSTAT_TYPE_RECONNECTION

Nula.

MQSTAT_TYPE_RECONNECTION_ERROR

Hodnota OpenOptions použitá v době, kdy došlo k selhání. Příčina poruchy je hlášena v polích *CompCode* a *Reason* ve struktuře MQSTS .

OpenOptions je výstupní pole. Jeho počáteční hodnota je nula.

SubOptions (MQLONG) pro MQSTS

Soubor SubOptions použitý k otevření selhávajícího odběru. K dispozici pouze ve verzi 2 produktu MQSTS nebo vyšší.

Interpretace parametru SubOptions závisí na hodnotě parametru MQSTAT **Type** .

MQSTAT_TYPE_ASYNC_ERROR

Nula.

MQSTAT_TYPE_RECONNECTION

Nula.

MQSTAT_TYPE_RECONNECTION_ERROR

Hodnota SubOptions použitá v době, kdy došlo k selhání. Pokud selhání nesouvisí s přihlášením k odběru tématu, vrácená hodnota je nula.

SubOptions je výstupní pole. Jeho počáteční hodnota je nula.

MQTM-zpráva spouštěče

Struktura MQTM popisuje data ve zprávě spouštěče, která je odeslána správcem front do aplikace monitoru spouštěčů při výskytu události spouštěče pro frontu.Tato struktura je součástí rozhraní IBM MQ Trigger Monitor Interface (TMI), které je jedním z rozhraní rámce IBM MQ .

Název formátu

MQFMT_TRIGGER.

Znaková sada a kódování

Znaková data v MQTM jsou ve znakové sadě správce front, který generuje MQTM. Číselná data v MQTM jsou v kódování počítače správce front, který generuje MQTM.

Znaková sada a kódování MQTM jsou dány poli *CodedCharSetId* a *Encoding* v:

- MQMD (pokud je struktura MQTM na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQTM (všechny ostatní případy).

Použití

Aplikace monitoru spouštěčů může potřebovat předat některé nebo všechny informace ve zprávě spouštěče aplikaci, kterou spouští aplikace monitoru spouštěčů. Informace, které může spuštěná aplikace potřebovat, zahrnují *QName*, *TriggerData*a *UserData*. Aplikace monitoru spouštěčů může předat strukturu MQTM přímo spuštěné aplikaci, nebo místo toho předat strukturu MQTMC2 v závislosti na tom, co je povoleno prostředím a vhodné pro spuštěnou aplikaci. Informace o MQTMC2viz “MQTMC2 -zpráva spouštěče 2 (znakový formát)” na stránce 608.

- z/OS V systému z/OSje pro aplikaci MQAT_CICS spuštěnou pomocí transakce CKTI zpřístupněna celá struktura zprávy spouštěče MQTM pro spuštěnou transakci. Informace lze načíst pomocí příkazu EXEC CICS NAČÍST.
- IBM i V systému IBM iaplikace monitoru spouštěčů dodávané s produktem IBM MQ předává spuštěnou aplikaci strukturu MQTMC2 .

Informace o použití spouštěčů naleznete v tématu Spuštění IBM MQ aplikací pomocí spouštěčů.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 533. Pole v MQTM pro MQTM

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	ID_STRUC_MQTM_STRUC_ID	'TM¬¬'
Verze (číslo verze struktury)	MQTM_VERSION_1	1
QName (název spuštěné fronty)	Není	Prázdný řetězec nebo mezery
ProcessName (název objektu procesu)	Není	Prázdný řetězec nebo mezery
TriggerData (data spouštěče)	Není	Prázdný řetězec nebo mezery
ApplType (typ aplikace	Není	0
ApplId (identifikátor aplikace)	Není	Prázdný řetězec nebo mezery

Tabulka 533. Pole v MQTM pro MQTM (pokračování)

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
EnvData (data prostředí)	Není	Prázdný řetězec nebo mezery
UserData (uživatelská data)	Není	Prázdný řetězec nebo mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQTM_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQTM MyTM = {MQTM_DEFAULT};
```

Deklarace jazyka

Prohlášení C pro MQTM

```
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 */
};
```

Deklarace jazyka COBOL pro MQTM

```
**   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).
```

Deklarace PL/I pro MQTM

```
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 */
```

Deklarace High Level Assembler pro MQTM

```
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)
```

Deklarace jazyka Visual Basic pro MQTM

```
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
```

MQMD pro zprávu spouštěče

Tabulka 534. Nastavení pro pole v deskriptoru MQMD zprávy spouštěče generované správcem front

Pole v deskriptoru MQMD	Použitá hodnota
StrucId	ID_STRUC_MQMD_STRUC_ID
Version	MQMD_VERSION_1
Report	MQRO_NONE
MsgType	MQMT_DATAGRAM
Expiry	MQEI_UNLIMITED
Feedback	MQFB_NONE
Encoding	MQENC_NATIVE
CodedCharSetId	Atribut **CodedCharSetId** správce front
Format	MQFMT_TRIGGER
Priority	Atribut **DefPriority** inicializační fronty

Tabulka 534. Nastavení pro pole v deskriptoru MQMD zprávy spouštěče generované správcem front (pokračování)

Pole v deskriptoru MQMD	Použitá hodnota
Persistence	MQPER_NOT_PERSISTENT
MsgId	Jedinečná hodnota
CorrelId	MQCI_NONE
BackoutCount	0
ReplyToQ	Mezery
ReplyToQMgr	Název správce front
UserIdentifier	Mezery
AccountingToken	MQACT_NONE
ApplIdentityData	Mezery
PutApplType	MQAT_QMGR nebo podle potřeby pro agenta kanálu zpráv
PutApplName	Prvních 28 bajtů názvu správce front
PutDate	Datum odeslání zprávy spouštěče
PutTime	Čas odeslání zprávy spouštěče
ApplOriginData	Mezery

Aplikaci, která generuje zprávu spouštěče, se doporučuje nastavit podobné hodnoty, s výjimkou následujících:

- Pole *Priority* lze nastavit na hodnotu MQPRI_PRIORITY_AS_Q_DEF (správce front jej při vložení zprávy změní na výchozí prioritu pro inicializační frontu).
- Pole *ReplyToQMgr* lze nastavit na prázdné hodnoty (správce front jej změní na název lokálního správce front při vložení zprávy).
- Nastavte kontextová pole podle potřeby pro aplikaci.

StrucId (MQCHAR4) pro MQTM

Jedná se o identifikátor struktury struktury zprávy spouštěče. Vždy se jedná o vstupní pole. Jeho hodnota je MQTM_STRUC_ID.

Hodnota musí být:

ID_STRUC_MQTM_STRUC_ID
Identifikátor pro strukturu zprávy spouštěče.

Pro programovací jazyk C je definována také konstanta MQTM_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQTM_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQTM

Toto je číslo verze struktury. Hodnota musí být:

MQTM_VERSION_1
Číslo verze pro strukturu zprávy spouštěče.

Následující konstanta určuje číslo verze aktuální verze:

MQTM_CURRENT_VERSION
Aktuální verze struktury zprávy spouštěče.

Počáteční hodnota tohoto pole je MQTM_VERSION_1.

Název QName (MQCHAR48) pro MQTM

Jedná se o název fronty, pro kterou došlo k události spouštěče, a je používán aplikací spuštěnou aplikací monitoru spouštěčů. Správce front inicializuje toto pole s hodnotou atributu **QName** spuštěné fronty. Podrobnosti o tomto atributu naleznete v části"Atributy pro fronty" na stránce 836.

Názvy, které jsou kratší než definovaná délka pole, jsou vpravo vyplněny mezerami; nejsou předčasně ukončeny znakem null.

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

ProcessName (MQCHAR48) pro MQTM

Jedná se o název objektu procesu správce front určeného pro spuštěnou frontu a může být použit aplikací monitoru spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **ProcessName** fronty určené polem *QName* ; podrobnosti o tomto atributu naleznete v části"Atributy pro fronty" na stránce 836.

Názvy, které jsou kratší než definovaná délka pole, jsou vždy zprava vyplněny mezerami; nejsou předčasně ukončeny znakem null.

Délka tohoto pole je dána hodnotou MQ_PROCESS_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

TriggerData (MQCHAR64) pro MQTM

Jedná se o data ve volném formátu pro použití aplikací monitoru spouštěčů, která obdrží zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **TriggerData** fronty určené polem *QName* ; podrobnosti o tomto atributu naleznete v části "Atributy pro fronty" na stránce 836 . Obsah těchto dat nemá pro správce front žádný význam.

V systému z/OSse pro aplikaci systému CICS spuštěnou pomocí transakce CKTI tyto informace nepoužívají.

Délka tohoto pole je dána hodnotou MQ_TRIGGER_DATA_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 64 prázdných znaků v jiných programovacích jazycích.

ApplType (MQLONG) pro MQTM

To identifikuje povahu programu, který se má spustit, a je používán aplikací pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole hodnotou atributu **ApplType** objektu procesu identifikovaného polem *ProcessName* . Podrobnosti o tomto atributu naleznete v části "Atributy pro definice procesu" na stránce 871 . Obsah těchto dat nemá pro správce front žádný význam.

ApplType může mít jednu z následujících standardních hodnot. Lze také použít uživatelem definované typy, ale měly by být omezeny na hodnoty v rozsahu MQAT_USER_FIRST až MQAT_USER_LAST:

MQAT_AIX-operační systém
Aplikace AIX (stejná hodnota jako MQAT_UNIX).

MQAT_BATCH
aplikace pro dávkové úlohy

MQAT_BROKER
Aplikace zprostředkovatele

MQAT_CICS
CICS transakce.

MQAT_CICS_BRIDGE
CICS bridge .

MQAT_CICS_VSE
CICS/VSE transakce.

MQAT_DOS
Aplikace IBM MQ MQI client na PC DOS.

MQAT_IMS
IMS .

MQAT_IMS_BRIDGE
Aplikace mostu IMS .

MQAT_JAVA
Java .

MQAT_MVS
Aplikace MVS nebo TSO (stejná hodnota jako MQAT_ZOS).

MQAT_NOTES_AGENT
Lotus Notes Aplikace agenta.

MQAT_OS390
Aplikace OS/390 (stejná hodnota jako MQAT_ZOS).

MQAT_OS400
IBM i .

MQAT_RRS_BATCH
Dávková aplikace RRS.

MQAT_UNIX
UNIX .

MQAT_UNKNOWN (neznámý)
Aplikace neznámého typu.

MQAT_USER
Typ aplikace definovaný uživatelem.

MQAT_VOS
Stratus VOS aplikace.

MQAT_WINDOWS
16bitová aplikace Windows .

MQAT_WINDOWS_NT
32bitová aplikace Windows .

MQAT_WLM
z/OS aplikace správce pracovní zátěže.

MQAT_XCF
XCF.

MQAT_ZOS
z/OS .

MQAT_USER_FIRST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

MQAT_USER_LAST
Nejvyšší hodnota pro typ aplikace definovaný uživatelem.

Počáteční hodnota tohoto pole je 0.

ApplId (MQCHAR256) pro MQTM

Jedná se o znakový řetězec, který identifikuje aplikaci, která má být spuštěna, a je používán aplikací monitoru spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole hodnotou atributu **ApplId** objektu procesu identifikovaného polem *ProcessName* . Podrobnosti o tomto atributu naleznete v části “Atributy pro definice procesu” na stránce 871 . Obsah těchto dat nemá pro správce front žádný význam.

Význam parametru *ApplId* je určen aplikací pro monitorování spouštěčů. Monitor spouštěčů poskytovaný produktem IBM MQ vyžaduje, aby *ApplId* byl název spustitelného programu. Následující poznámky platí pro uvedená prostředí:

- V systému z/OSje *ApplId* :
  - Identifikátor transakce CICS pro aplikace spuštěné pomocí CKTI transakce monitoru spouštěčů CICS .
  - Identifikátor transakce IMS pro aplikace spuštěné pomocí monitoru spouštěčů IMS CSQQTRMN
- Na systémech Windows může mít název programu předponu s cestou k jednotce a adresáři.
- V systému IBM imůže mít název programu předponu s názvem knihovny a/znak.
- V systému AIX and Linuxmůže mít název programu předponu s cestou k adresáři.

Délka tohoto pole je dána hodnotou MQ_PROCESS_APPL_ID_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 256 prázdných znaků v jiných programovacích jazycích.

EnvData (MQCHAR128) pro MQTM

Jedná se o znakový řetězec, který obsahuje informace související s prostředím aplikace, která má být spuštěna, a je používán aplikací monitoru spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole hodnotou atributu **EnvData** objektu procesu identifikovaného polem *ProcessName* . Podrobnosti o tomto atributu naleznete v části “Atributy pro definice procesu” na stránce 871 . Obsah těchto dat nemá pro správce front žádný význam.

V systému z/OSse pro aplikaci systému CICS spuštěnou pomocí transakce CKTI nebo pro aplikaci IMS , která má být spuštěna pomocí transakce CSQQTRMN, tyto informace nepoužívají.

Délka tohoto pole je dána hodnotou MQ_PROCESS_ENV_DATA_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 128 prázdných znaků v jiných programovacích jazycích.

UserData (MQCHAR128) pro MQTM

Jedná se o znakový řetězec, který obsahuje informace o uživateli související s aplikací, která má být spuštěna, a je používán aplikací monitoru spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole hodnotou atributu **UserData** objektu procesu identifikovaného polem *ProcessName* . Podrobnosti o tomto atributu naleznete v části “Atributy pro definice procesu” na stránce 871 . Obsah těchto dat nemá pro správce front žádný význam.

V případě produktu Microsoft Windowsnesmí znakový řetězec obsahovat dvojité uvozovky, pokud má být definice procesu předána do **runmqtrm**.

Délka tohoto pole je dána hodnotou MQ_PROCESS_USER_DATA_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 128 prázdných znaků v jiných programovacích jazycích.

MQTMC2 -zpráva spouštěče 2 (znakový formát)

Když aplikace monitoru spouštěčů načte zprávu spouštěče (MQTM) z inicializační fronty, může být nutné, aby monitor spouštěčů předal některé nebo všechny informace ve zprávě spouštěče aplikaci, kterou spouští monitor spouštěčů.

Informace, které může spuštěná aplikace potřebovat, zahrnují *QName*, *TriggerData*a *UserData*. Aplikace monitoru spouštěčů může předat strukturu MQTM přímo spuštěné aplikaci, nebo místo toho předat strukturu MQTMC2 v závislosti na tom, co je povoleno prostředím a vhodné pro spuštěnou aplikaci.

Tato struktura je součástí rozhraní IBM MQ Trigger Monitor Interface (TMI), které je jedním z rozhraní rámce IBM MQ .

Znaková sada a kódování

Znaková data v produktu MQTMC2 jsou ve znakové sadě lokálního správce front. Tato sada je určena atributem správce front **CodedCharSetId** .

Použití

Struktura MQTMC2 je velmi podobná formátu struktury MQTM. Rozdíl je v tom, že neznaková pole v produktu MQTM jsou v produktu MQTMC2 změněna na znaková pole stejné délky a na konec struktury je přidán název správce front.

- z/OS V systému z/OSje pro aplikaci MQAT_IMS spuštěnou pomocí aplikace CSQQTRMN zpřístupněna struktura MQTMC2 pro spuštěnou aplikaci.
- IBM i V systému IBM iaplikace monitoru spouštěčů dodávané s produktem IBM MQ předává spuštěnou aplikaci strukturu MQTMC2 .

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 535. Pole v tabulce MQTMC2

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQTMC_STRUC_ID	'TMC¬'
Verze (číslo verze struktury)	MQTMC_VERSION_2	'¬¬¬2'
QName (název spuštěné fronty)	Není	Prázdný řetězec nebo mezery
ProcessName (název objektu procesu)	Není	Prázdný řetězec nebo mezery
TriggerData (data spouštěče)	Není	Prázdný řetězec nebo mezery
ApplType (typ aplikace)	Není	Mezery
ApplId (identifikátor aplikace)	Není	Prázdný řetězec nebo mezery
EnvData (data prostředí)	Není	Prázdný řetězec nebo mezery
UserData (uživatelská data)	Není	Prázdný řetězec nebo mezery
QMgrName (název správce front)	Není	Prázdný řetězec nebo mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQTMC2_DEFAULT obsahuje výše uvedené hodnoty. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQTMC2 MyTMC = {MQTMC2_DEFAULT};
```

Deklarace jazyka

Deklarace jazyka C pro MQTMC2

```
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 */
};
```

Deklarace jazyka COBOL pro MQTMC2

```
**   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).
```

Deklarace PL/I pro MQTMC2

```
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 */
```

Deklarace High Level Assembler pro MQTMC2

```
MQTMC2              DSECT
MQTMC2_STRUCID      DS   CL4    Structure identifier
MQTMC2_VERSION      DS   CL4    Structure version number
MQTMC2_QNAME        DS   CL48   Name of triggered queue
MQTMC2_PROCESSNAME  DS   CL48   Name of process object
MQTMC2_TRIGGERDATA  DS   CL64   Trigger data
MQTMC2_APPLTYPE     DS   CL4    Application type
MQTMC2_APPLID       DS   CL256  Application identifier
MQTMC2_ENVDATA      DS   CL128  Environment data
MQTMC2_USERDATA     DS   CL128  User data
MQTMC2_QMGRNAME     DS   CL48   Queue manager name
*
MQTMC2_LENGTH       EQU  *-MQTMC2
```

```
                    ORG  MQTMC2
MQTMC2_AREA         DS   CL(MQTMC2_LENGTH)
```

Deklarace jazyka Visual Basic pro MQTMC2

```
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
```

StrucId (MQCHAR4) pro MQTMC2

Jedná se o identifikátor struktury struktury zprávy spouštěče 2 (znakový formát). Vždy se jedná o vstupní pole. Jeho hodnota je MQTMC2_STRUC_ID.

Hodnota musí být:

MQTMC2_STRUC_ID
Identifikátor struktury zprávy spouštěče (znakový formát).

Pro programovací jazyk C je definována také konstanta MQTMC2_STRUC_ID_ARRAY . Má stejnou hodnotu jako MQTMC2_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQCHAR4) pro MQTMC2

Číslo verze struktury.

Hodnota musí být:

MQTMC_VERSION_2
Struktura zprávy spouštěče verze 2 (znakový formát).

Pro programovací jazyk C je definována také konstanta MQTMC_VERSION_2_ARRAY ; má stejnou hodnotu jako MQTMC_VERSION_2, ale je to pole znaků namísto řetězce.

Následující konstanta určuje číslo verze aktuální verze:

MQTMC_CURRENT_VERSION
Aktuální verze struktury zprávy spouštěče (znakový formát).

Název QName (MQCHAR48) pro MQTMC2

Název spuštěné fronty.

Viz pole *QName* ve struktuře MQTM.

ProcessName (MQCHAR48) pro MQTMC2

Název objektu procesu.

Viz pole *ProcessName* ve struktuře MQTM.

TriggerData (MQCHAR64) pro MQTMC2

Data spouątěče.

Viz pole *TriggerData* ve struktuře MQTM.

ApplType (MQCHAR4) pro MQTMC2

Typ aplikace.

Toto pole vždy obsahuje mezery, bez ohledu na hodnotu v poli *ApplType* ve struktuře MQTM původní zprávy spouštěče.

ApplId (MQCHAR256) pro MQTMC2

Identifikátor aplikace.

Viz pole *ApplId* ve struktuře MQTM.

EnvData (MQCHAR128) pro MQTMC2

Data prostředí.

Viz pole *EnvData* ve struktuře MQTM.

UserData (MQCHAR128) pro MQTMC2

Uživatelská data.

Viz pole *UserData* ve struktuře MQTM.

QMgrName (MQCHAR48) pro MQTMC2

Název správce front.

Jedná se o název správce front, ve kterém došlo k události spouštěče.

MQWIH-záhlaví informací o práci

Má-li být zpráva zpracována správcem pracovní zátěže z/OS (WLM), musí zpráva začínat strukturou MQWIH. Tato struktura popisuje informace, které musí být přítomny na začátku zprávy, která má být zpracována modulem WLM.

Dostupnost

Všechny systémy IBM MQ a klienti IBM MQ připojení k těmto systémům.

Název formátu

MQFMT_WORK_INFO_HEADER.

Znaková sada a kódování

Pole ve struktuře MQWIH jsou ve znakové sadě a kódování dané poli *CodedCharSetId* a *Encoding* ve struktuře záhlaví, která předchází MQWIH, nebo těmito poli ve struktuře MQMD, pokud je MQWIH na začátku dat zprávy aplikace.

Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech front.

Pouľití

Pro libovolnou podporovanou platformu IBM MQ můžete vytvořit a přenést zprávu, která obsahuje strukturu MQWIH, ale pouze správce front IBM MQ for z/OS může interaktivně spolupracovat s WLM. Proto, aby se zpráva dostala do WLM ze správce front jiného nežz/OS , musí síť správců front obsahovat alespoň jednoho správce front z/OS , jehož prostřednictvím může být zpráva směrována.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 536. Pole v MQWIH

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQWIH_STRUC_ID	'WIH¬'
Verze (číslo verze struktury)	MQWIH_VERSION_1	1
StrucLength (délka struktury MQWIH)	MQWIH_LENGTH_1	120
Kódování (číselné kódování dat, která následují za MQWIH)	Není	0
CodedCharSetId (identifikátor znakové sady dat, která následují za MQWIH)	MQCCSI_UNDEFINED	0
Formát (název formátu dat, která následují za MQWIH)	MQFMT_NONE	Mezery
Příznaky (příznaky)	MQWIH_NONE	0
ServiceName (název služby)	Není	Mezery
ServiceStep (název kroku služby)	Není	Mezery
MsgToken (token zprávy)	MQMTOK_NONE	Hodnoty null
Vyhrazeno (vyhrazeno)	Není	Mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C se jedná o proměnnou makra.MQWIH_DEFAULT obsahuje hodnoty uvedené v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQWIH MyWIH = {MQWIH_DEFAULT};
```

Deklarace jazyka

C prohlášení pro MQWIH

```
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 */
};
```

Deklarace jazyka COBOL pro MQWIH

```
**   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).
```

Deklarace PL/I pro MQWIH

```
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 */
```

Deklarace High Level Assembler pro MQWIH

```
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)
```

Vizuální základní deklarace pro MQWIH

```
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
```

StrucId (MQCHAR4) pro MQWIH

Jedná se o identifikátor struktury struktury záhlaví informací o práci. Vždy se jedná o vstupní pole. Jeho hodnota je MQWIH_STRUC_ID.

Hodnota musí být:

MQWIH_STRUC_ID
Identifikátor pro strukturu záhlaví informací o práci.

Pro programovací jazyk C je definována také konstanta MQWIH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQWIH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQWIH

Toto je číslo verze struktury. Hodnota musí být:

MQWIH_VERSION_1
Struktura záhlaví pracovních informací Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

MQWIH_CURRENT_VERSION
Aktuální verze struktury záhlaví informací o práci.

Počáteční hodnota tohoto pole je MQWIH_VERSION_1.

StrucLength (MQLONG) pro MQWIH

Jedná se o délku struktury MQWIH. Hodnota musí být:

MQWIH_LENGTH_1
Délka struktury záhlaví pracovních informací version-1 .

Následující konstanta určuje délku aktuální verze:

MQWIH_CURRENT_LENGTH
Délka aktuální verze struktury záhlaví informací o práci.

Počáteční hodnota tohoto pole je MQWIH_LENGTH_1.

Kódování (MQLONG) pro MQWIH

Tato volba určuje číselné kódování dat, která následují za strukturou MQWIH; nevztahuje se na číselná data v samotné struktuře MQWIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

CodedCharSetId (MQLONG) pro MQWIH

Tato volba určuje identifikátor znakové sady dat, která se řídí strukturou MQWIH; nevztahuje se na znaková data v samotné struktuře MQWIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Můžete použít následující speciální hodnotu:

MQCCSI_INHERIT
Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota MQCCSI_INHERIT vrácena voláním MQGET.

MQCCSI_INHERIT nelze použít, pokud hodnota pole *PutApplType* v MQMD je MQAT_BROKER.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8) pro MQWIH

Tato volba určuje název formátu dat, která se řídí strukturou MQWIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *Format* v MQMD.

Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG) pro MQWIH

Hodnota musí být:

MQWIH_NONE
Žádné příznaky.

Počáteční hodnota tohoto pole je MQWIH_NONE.

ServiceName (MQCHAR32) pro MQWIH

Jedná se o název služby, která má zpracovat zprávu.

Délka tohoto pole je dána hodnotou MQ_SERVICE_NAME_LENGTH. Počáteční hodnota tohoto pole je 32 prázdných znaků.

ServiceStep (MQCHAR8) pro MQWIH

Jedná se o název kroku *ServiceName* , ke kterému se zpráva vztahuje.

Délka tohoto pole je dána hodnotou MQ_SERVICE_STEP_LENGTH. Počáteční hodnota tohoto pole je 8 prázdných znaků.

MsgToken (MQBYTE16) pro MQWIH

Jedná se o token zprávy, který jednoznačně identifikuje zprávu.

Pro volání MQPUT a MQPUT1 je toto pole ignorováno. Délka tohoto pole je dána hodnotou MQ_MSG_TOKEN_LENGTH. Počáteční hodnota tohoto pole je MQMTOK_NONE.

Vyhrazeno (MQCHAR32) pro MQWIH

Toto je vyhrazené pole; musí být prázdné.

MQXP-blok parametrů ukončení

Struktura MQXP se používá jako vstupní/výstupní parametr pro uživatelskou proceduru křížení rozhraní API.Další informace o této uživatelské proceduře naleznete v tématu Překračující uživatelská procedura rozhraní API.

Znaková sada a kódování

Znaková data v systému MQXP jsou ve znakové sadě lokálního správce front. Tato sada je určena atributem správce front **CodedCharSetId** . Číselná data v MQXP jsou v nativním kódování počítače; toto je dáno MQENC_NATIVE.

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 537. Pole v MQXP	
Název a popis pole	**Název konstanty**
StrucId (identifikátor struktury)	MQXP_STRUC_ID
Verze (číslo verze struktury)	MQXP_VERSION_1
ExitId (identifikátor ukončení)	MQXT_API_CROSSING_EXIT
ExitReason (příčina vyvolání ukončení)	Není
ExitResponse (odezva od ukončení)	Není
ExitCommand (kód volání rozhraní API)	Není
ExitParmPočet (počet parametrů)	Není
Vyhrazeno (vyhrazeno)	Není
ExitUserOblast (oblast uživatele)	Není

Deklarace jazyka

Deklarace jazyka C pro systém MQXP

```
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 */
};
```

Deklarace jazyka COBOL pro MQXP

```
**   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).
```

Deklarace PL/I pro MQXP

```
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 */
```

Deklarace High Level Assembler pro systém MQXP

```
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)
```

StrucId (MQCHAR4) pro MQXP

Jedná se o identifikátor struktury struktury parametru ukončení. Vždy se jedná o vstupní pole. Jeho hodnota je MQXP_STRUC_ID.

Hodnota musí být:

MQXP_STRUC_ID
Identifikátor pro strukturu parametru ukončení.

Pro programovací jazyk C je definována také konstanta MQXP_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQXP_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQXP

Toto je číslo verze struktury. Hodnota musí být:

MQXP_VERSION_1
Číslo verze pro strukturu výstupního parametru-bloku.

Poznámka: Při zavedení nové verze této struktury se nezmění rozvržení stávající části. Uživatelská procedura proto musí zkontrolovat, zda je číslo verze stejné nebo větší než nejnižší verze obsahující pole, která uživatelská procedura potřebuje použít.

Toto je vstupní pole pro ukončení.

ExitId (MQLONG) pro MQXP

Tato hodnota je nastavena při vstupu do uživatelské procedury a označuje typ uživatelské procedury:

MQXT_API_CROSSING_EXIT
Uživatelská procedura přechodu rozhraní API pro CICS.

Toto je vstupní pole pro ukončení.

ExitReason (MQLONG) pro MQXP

Tato volba je nastavena při vstupu do uživatelské procedury. Pro uživatelskou proceduru rozhraní API označuje, zda je rutina volána před nebo po provedení volání rozhraní API:

MQXR_BEFORE (předchozí)
Před provedením rozhraní API.

MQXR_AFTER
Po provedení rozhraní API.

Toto je vstupní pole pro ukončení.

ExitResponse (MQLONG) pro MQXP

Hodnota je nastavena uživatelskou procedurou pro komunikaci s volajícím. Jsou definovány tyto hodnoty:

MQXCC_OK
Ukončení bylo úspěšně dokončeno.

MQXCC_SUPPRESS_FUNCTION
Potlačit funkci.

Je-li tato hodnota nastavena uživatelskou procedurou pro přechod rozhraní API s názvem *před* voláním rozhraní API, volání rozhraní API se neprovede. Parametr *CompCode* pro volání je nastaven na hodnotu MQCC_FAILED, parametr *Reason* je nastaven na hodnotu MQRC_SUPPRESSED_BY_EXIT a všechny ostatní parametry zůstávají při zanechání výstupu.

Je-li tato hodnota nastavena uživatelskou procedurou pro přechod rozhraní API s názvem *po* volání rozhraní API, správce front ji ignoruje.

Funkce MQXCC_SKIP_FUNCTION
Funkce přeskočení.

Je-li tato hodnota nastavena uživatelskou procedurou pro křížení rozhraní API s názvem *před* voláním rozhraní API, volání rozhraní API se neprovede; parametry *CompCode* a *Reason* a všechny ostatní parametry zůstanou v době, kdy je uživatelská procedura opustila.

Je-li tato hodnota nastavena uživatelskou procedurou pro přechod rozhraní API s názvem *po* volání rozhraní API, správce front ji ignoruje.

Toto je výstupní pole z uživatelské procedury.

ExitCommand (MQLONG) pro MQXP

Toto pole je nastaveno při vstupu do uživatelské procedury. Identifikuje volání rozhraní API, které způsobilo vyvolání uživatelské procedury:

MQXC_CALLBACK
Volání CALLBACK.

MQXC_MQBACK
Volání MQBACK.

MQXC_MQCB
Volání MQCB.

MQXC_MQCLOSE
Volání MQCLOSE.

MQXC_MQCMIT
Volání MQCMIT.

MQXC_MQCTL
Volání MQCTL.

MQXC_MQGET
Volání MQGET.

MQXC_MQINQ
Volání MQINQ.

MQXC_MQOPEN
Volání MQOPEN.

MQXC_MQPUT
Volání MQPUT.

MQXC_MQPUT1
Volání MQPUT1 .

MQXC_MQSET
Volání MQSET.

MQXC_MQSTAT
Volání MQSTAT.

MQXC_MQSUB
Volání MQSUB.

MQXC_MQSUBRQ
Volání MQSUBRQ.

Toto je vstupní pole pro ukončení.

ExitParmPočet (MQLONG) pro MQXP

Toto pole je nastaveno při vstupu do uživatelské procedury. Obsahuje počet parametrů, které volání MQ přijímá.

Tabulka 538. Počet parametrů pro každé volání MQ

Název volání	Počet parametrů
MQBACK	3
MQCLOSE	5
MQCMIT	3
MQGET	9
MQINQ	10
MQOPEN	6
MQPUT	8
MQPUT1	8
MQSET	10

Toto je vstupní pole pro ukončení.

Vyhrazeno (MQLONG) pro MQXP

Toto je vyhrazené pole. Jeho hodnota není pro ukončení významná.

Oblast ExitUser(MQBYTE16) pro MQXP

Jedná se o pole, které je k dispozici pro použití uživatelskou procedurou. Je inicializován na binární nulu pro délku pole před prvním vyvoláním uživatelské procedury pro úlohu a poté jsou veškeré změny provedené v tomto poli uživatelskou procedurou zachovány při vyvolání uživatelské procedury. Je definována následující hodnota:

MQXUA_NONE
Žádné informace o uživateli.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQXUA_NONE_ARRAY; má stejnou hodnotu jako MQXUA_NONE, ale je to pole znaků místo řetězce.

Délka tohoto pole je dána hodnotou MQ_EXIT_USER_AREA_LENGTH. Toto je vstupní/výstupní pole pro ukončení.

MQXQH-záhlaví přenosové fronty

Struktura MQXQH popisuje informace, které mají předponu k datům zpráv aplikace pro zprávy, když jsou v přenosových frontách.Přenosová fronta je speciální typ lokální fronty, která dočasně zadržuje zprávy určené pro vzdálené fronty (tj. určené pro fronty, které nepatří lokálnímu správci front). Přenosová fronta je označena atributem fronty **Usage** s hodnotou MQUS_TRANSMISSION.

Název formátu

MQFMT_XMIT_Q_HEADER

Znaková sada a kódování

Data v MQXQH musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané MQENC_NATIVE.

Nastavte znakovou sadu a kódování MQXQH do polí *CodedCharSetId* a *Encoding* v:

- samostatné MQMD (pokud je struktura MQXQH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQXQH (všechny ostatní případy).

Pole

Poznámka: V následující tabulce jsou pole seskupena podle použití a nikoli abecedně. Podřízená témata se řídí stejnou posloupností.

Tabulka 539. Pole v MQXQH pro MQXQH

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQXQH_STRUC_ID	'XQH¬'
Verze (číslo verze struktury)	MQXQH_VERSION_1	1
RemoteQName (název cílové fronty)	Není	Prázdný řetězec nebo mezery
RemoteQMgrNázev (název cílového správce front)	Není	Prázdný řetězec nebo mezery
MsgDesc (původní deskriptor zprávy)	Stejné názvy a hodnoty jako MQMD; viz Tabulka 500 na stránce 425	-

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyku C se jedná o proměnnou makra.MQXQH_DEFAULT obsahuje hodnoty, které jsou uvedeny v tabulce. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQXQH MyXQH = {MQXQH_DEFAULT};
```

Deklarace jazyka

Deklarace jazyka C pro MQXQH

```
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 */
};
```

Deklarace jazyka COBOL pro MQXQH

```
**   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).
```

Deklarace PL/I pro MQXQH

```
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 */
```

Deklarace High Level Assembler pro MQXQH

```
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)
```

Deklarace jazyka Visual Basic pro MQXQH

```
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
```

Pole v odděleném deskriptoru zprávy

Zpráva, která je v přenosové frontě, má *dva* deskriptory zpráv:

- Jeden deskriptor zprávy je uložen odděleně od dat zprávy. Tento deskriptor se nazývá *samostatný deskriptor zprávy*a je generován správcem front při umístění zprávy do přenosové fronty. Některá pole v odděleném deskriptoru zpráv jsou zkopírována z deskriptoru zpráv poskytnutého aplikací ve volání MQPUT nebo MQPUT1 .

  Oddělený deskriptor zprávy je ten, který je vrácen aplikaci v parametru **MsgDesc** volání MQGET při odebrání zprávy z přenosové fronty.
- Druhý deskriptor zprávy je uložen ve struktuře MQXQH jako součást dat zprávy; tento deskriptor se nazývá *vložený deskriptor zprávy*a je kopií deskriptoru zprávy, který byl poskytnut aplikací ve volání MQPUT nebo MQPUT1 (s menšími variacemi).

  Vložený deskriptor zprávy je vždy MQMD version-1 . Pokud má zpráva vložená aplikací jiné než výchozí hodnoty pro jedno nebo více polí version-2 v deskriptoru MQMD, následuje struktura MQMDE za MQXQH a je následována daty zprávy aplikace (pokud existují). MQMDE je buď:

  – Generováno správcem front (pokud aplikace k vložení zprávy používá MQMD version-2), nebo
  – Již se nachází na začátku dat zprávy aplikace (pokud aplikace používá k vložení zprávy MQMD version-1).

  Vložený deskriptor zprávy je ten, který je vrácen aplikaci v parametru **MsgDesc** volání MQGET při odebrání zprávy z konečné cílové fronty.

Pole v odděleném deskriptoru zpráv jsou nastavena správcem front, jak je zobrazeno. Pokud správce front nepodporuje rozhraní MQMD version-2 , bude použit modul MQMD version-1 bez ztráty funkce.

Tabulka 540. Hodnoty použité pro pole v odděleném deskriptoru MQMD

Pole v odděleném deskriptoru MQMD	Použitá hodnota
StrucId	ID_STRUC_MQMD_STRUC_ID
Version	MQMD_VERSION_2
Report	Zkopírováno z vloženého deskriptoru zprávy, ale s bity identifikovanými hodnotou MQRO_ACCEPT_UNSUP_IF_XMIT_MASK nastavenou na nulu. (To zabraňuje generování zprávy sestavy COA nebo COD, když je zpráva umístěna do přenosové fronty nebo odebrána z přenosové fronty.)
MsgType	Zkopírováno z vloženého deskriptoru zprávy.
Expiry	Zkopírováno z vloženého deskriptoru zprávy.

Tabulka 540. Hodnoty použité pro pole v odděleném deskriptoru MQMD (pokračování)

Pole v odděleném deskriptoru MQMD	Použitá hodnota
Feedback	Zkopírováno z vloženého deskriptoru zprávy.
Encoding	MQENC_NATIVE (viz poznámka)
CodedCharSetId	Atribut **CodedCharSetId** správce front.
Format	MQFMT_XMIT_Q_HEADER
Priority	Zkopírováno z vloženého deskriptoru zprávy.
Persistence	Zkopírováno z vloženého deskriptoru zprávy.
MsgId	Správce front vygeneruje novou hodnotu. Tento identifikátor zprávy se liší od identifikátoru *MsgId* , který mohl být vygenerován správcem front pro dříve popsaný vložený deskriptor zprávy.
CorrelId	*MsgId* z vloženého deskriptoru zprávy. Pro zprávy vkládané do SYSTEM.CLUSTER.TRANSMIT.QUEUE, *CorrelId* je vyhrazeno pro vnitřní použití.
BackoutCount	0
ReplyToQ	Zkopírováno z vloženého deskriptoru zprávy.
ReplyToQMgr	Zkopírováno z vloženého deskriptoru zprávy.
UserIdentifier	Zkopírováno z vloženého deskriptoru zprávy.
AccountingToken	Zkopírováno z vloženého deskriptoru zprávy. Pro zprávy vkládané do SYSTEM.CLUSTER.TRANSMIT.QUEUE, *AccountingToken* je vyhrazeno pro vnitřní použití.
ApplIdentityData	Zkopírováno z vloženého deskriptoru zprávy.
PutApplType	MQAT_QMGR
PutApplName	Prvních 28 bajtů názvu správce front.
PutDate	Datum, kdy byla zpráva vložena do přenosové fronty.
PutTime	Čas, kdy byla zpráva vložena do přenosové fronty.
ApplOriginData	Mezery
GroupId	MQGI_NONE
MsgSeqNumber	1
Offset	0
MsgFlags	MQMF_NONE
OriginalLength	MQOL_UNDEFINED

- V systému Windowsse hodnota MQENC_NATIVE pro Micro Focus COBOL liší od hodnoty pro C. Hodnota v poli *Encoding* v odděleném deskriptoru zprávy je vždy hodnota pro C v těchto prostředích; tato hodnota je 546 v desítkové soustavě. Také celočíselná pole ve struktuře MQXQH jsou v kódování, které odpovídá této hodnotě (nativní kódování Intel).

Pole ve vloženém deskriptoru zprávy

Pole ve vloženém deskriptoru zprávy mají stejné hodnoty jako pole v parametru **MsgDesc** volání MQPUT nebo MQPUT1 , s výjimkou následujících:

- Pole *Version* má vždy hodnotu MQMD_VERSION_1.

- Pokud má pole *Priority* hodnotu MQPRI_PRIORITY_AS_Q_DEF, nahradí se hodnotou atributu **DefPriority** fronty.
- Pokud má pole *Persistence* hodnotu MQPER_PERSISTENCE_AS_Q_DEF, nahradí se hodnotou atributu **DefPersistence** fronty.
- Pokud má pole *MsgId* hodnotu MQMI_NONE nebo byla zadána volba MQPMO_NEW_MSG_ID nebo je zpráva zprávou distribučního seznamu, je hodnota *MsgId* nahrazena novým identifikátorem zprávy generovaným správcem front.

  Když je zpráva distribučního seznamu rozdělena na menší zprávy distribučního seznamu umístěné v různých přenosových frontách, pole *MsgId* v každém z nových vložených deskriptorů zpráv je stejné jako v původní zprávě distribučního seznamu.
- Pokud byla zadána volba MQPMO_NEW_CORREL_ID, bude položka *CorrelId* nahrazena novým identifikátorem korelace generovaným správcem front.
- Pole kontextu jsou nastavena podle volby MQPMO_ * _CONTEXT určené v parametru **PutMsgOpts** ; pole kontextu jsou:
  - *AccountingToken*
  - *ApplIdentityData*
  - *ApplOriginData*
  - *PutApplName*
  - *PutApplType*
  - *PutDate*
  - *PutTime*
  - *UserIdentifier*
- Pole version-2 (pokud byla přítomna) jsou odebrána z MQMD a přesunuta do struktury MQMDE, pokud jedno nebo více polí version-2 má jinou než výchozí hodnotu.

Vkládání zpráv do vzdálených front

Pokud aplikace vloží zprávu do vzdálené fronty (buď zadáním názvu vzdálené fronty přímo, nebo pomocí lokální definice vzdálené fronty), lokální správce front:

- Vytvoří strukturu MQXQH obsahující vložený deskriptor zprávy.
- Připojí MQMDE, je-li potřeba a není-li již přítomen
- Připojí data zprávy aplikace
- Umístí zprávu do odpovídající přenosové fronty.

Vkládání zpráv přímo do přenosových front

Aplikace může také vložit zprávu přímo do přenosové fronty. V tomto případě musí aplikace před data zprávy aplikace přidat strukturu MQXQH a inicializovat pole s odpovídajícími hodnotami. Kromě toho pole *Format* v parametru **MsgDesc** volání MQPUT nebo MQPUT1 musí mít hodnotu MQFMT_XMIT_Q_HEADER.

Znaková data ve struktuře MQXQH vytvořené aplikací musí být ve znakové sadě lokálního správce front (definované atributem správce front **CodedCharSetId**) a celočíselná data musí být v nativním kódování počítače. Kromě toho musí být znaková data ve struktuře MQXQH vyplněna mezerami na definovanou délku pole; data nesmí být předčasně ukončena pomocí znaku null, protože správce front nepřevede hodnotu null a následné znaky na mezery ve struktuře MQXQH.

Správce front však nekontroluje, zda je přítomna struktura MQXQH nebo zda byly pro pole zadány platné hodnoty.

Aplikace by neměly vkládat své zprávy přímo do SYSTEM.CLUSTER.TRANSMIT.QUEUE.

Získávání zpráv z přenosových front

Aplikace, které získají zprávy z přenosové fronty, musí informace ve struktuře MQXQH zpracovat odpovídajícím způsobem. Přítomnost struktury MQXQH na začátku dat zprávy aplikace je označena hodnotou MQFMT_XMIT_Q_HEADER vrácenou v poli *Format* v parametru **MsgDesc** volání MQGET. Hodnoty vrácené v polích *CodedCharSetId* a *Encoding* v parametru **MsgDesc** označují znakovou sadu a kódování znakových a celočíselných dat ve struktuře MQXQH. Znaková sada a kódování dat zprávy aplikace jsou definovány poli *CodedCharSetId* a *Encoding* ve vloženém deskriptoru zprávy.

StrucId (MQCHAR4) pro MQXQH

Toto je identifikátor struktury struktury záhlaví přenosové fronty. Vždy se jedná o vstupní pole. Jeho hodnota je MQXQH_STRUC_ID.

Hodnota musí být:

MQXQH_STRUC_ID
Identifikátor pro strukturu záhlaví přenosové fronty.

Pro programovací jazyk C je definována také konstanta MQXQH_STRUC_ID_ARRAY. Má stejnou hodnotu jako MQXQH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG) pro MQXQH

Toto je číslo verze struktury. Hodnota musí být:

MQXQH_VERSION_1
Číslo verze pro strukturu záhlaví přenosové fronty.

Následující konstanta určuje číslo verze aktuální verze:

MQXQH_CURRENT_VERSION
Aktuální verze struktury záhlaví přenosové fronty.

Počáteční hodnota tohoto pole je MQXQH_VERSION_1.

RemoteQName (MQCHAR48) pro MQXQH

Jedná se o název fronty zpráv, která je zdánlivě konečným cílem zprávy (to se může ukázat jako nevhodné, pokud je například tato fronta definována v *RemoteQMgrName* jako lokální definice jiné vzdálené fronty).

Pokud se jedná o zprávu distribučního seznamu (tj. pole *Format* ve vloženém deskriptoru zprávy je MQFMT_DIST_HEADER), je pole *RemoteQName* prázdné.

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

RemoteQMgrNázev (MQCHAR48) pro MQXQH

Jedná se o název správce front nebo skupiny sdílení front, která vlastní frontu, jež je zdánlivě eventuálním cílem zprávy.

Pokud se jedná o zprávu rozdělovníku, *RemoteQMgrName* je prázdné.

Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích.

MsgDesc (MQMD1) pro MQXQH

Jedná se o vložený deskriptor zprávy a jedná se o blízkou kopii deskriptoru zprávy MQMD, který byl zadán jako parametr **MsgDesc** ve volání MQPUT nebo MQPUT1 při původním vložení zprávy do vzdálené fronty.

Poznámka: Jedná se o databázi MQMD version-1 .

Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQMD.

Volání funkcí

Tento oddíl poskytuje informace o všech možných voláních MQI. Popisy, syntaxe, informace o parametrech, poznámky k použití a vyvolání jazyka pro každý možný jazyk jsou uvedeny pro každé z různých volání.

Související odkazy

z/OS Příklady výstupu CEDF z volání MQI

Popisy volání

Tento oddíl popisuje volání MQI.

Pro tato volání je k dispozici nápověda online na platformách UNIX ve formě stránek *man* .

Poznámka: Volání přidružená k převodu dat MQXCNVC a MQ_DATA_CONV_EXIT jsou v adresáři "Uživatelská procedura převodu dat" na stránce 905.

Konvence používané v popisech volání

Pro každé volání tato kolekce témat poskytuje popis parametrů a použití volání ve formátu, který je nezávislý na programovacím jazyku. Následuje typická vyvolání volání a typická deklarace jeho parametrů v každém z podporovaných programovacích jazyků.

Důležité: Při kódování volání rozhraní API IBM MQ musíte zajistit, aby byly poskytnuty všechny příslušné parametry (jak je popsáno v následujících sekcích). Pokud tak neučiníme, může dojít k nepředvídatelným výsledkům.

Popis každého volání obsahuje následující sekce:

Název volání
Název volání následovaný stručným popisem účelu volání.

Parametry
Pro každý parametr je název následován jeho datovým typem v závorkách () a jeden z následujících:

Vstup
Informace do parametru zadáváte při volání.

výstup
Správce front vrátí informace v parametru po dokončení nebo selhání volání.

Vstup a výstup
Do parametru zadáte informace při provádění volání a správce front změní informace při dokončení nebo selhání volání.

Příklad:

Compcode (MQLONG)-výstup

V některých případech je datový typ struktura. Ve všech případech jsou další informace o datovém typu nebo struktuře v souboru "Základní datové typy" na stránce 235.

Poslední dva parametry v každém volání jsou kód dokončení a kód příčiny. Kód dokončení označuje, zda bylo volání úspěšně, částečně nebo vůbec dokončeno. Další informace o částečném úspěchu nebo selhání volání jsou uvedeny v kódu příčiny. Další informace o každém dokončení a kódu příčiny viz "Návratové kódy" na stránce 874.

Poznámky k použití
Další informace o volání, popisující jeho použití a případná omezení jeho použití.

Vyvolání jazyka asembleru
Typické vyvolání volání a deklarace jeho parametrů v jazyku asembleru.

Vyvolání jazyka C
Typické vyvolání volání a deklarace jeho parametrů v jazyce C.

Vyvolání COBOL
Typické vyvolání volání a deklarace jeho parametrů v COBOLu.

Vyvolání PL/I
Typické vyvolání volání a deklarace jeho parametrů v PL/I.

Všechny parametry jsou předávány odkazem.

Vyvolání jazyka Visual Basic
Typické vyvolání volání a deklarace jeho parametrů v jazyku Visual Basic.

Další konvence zápisu jsou:

Konstanty
Názvy konstant jsou uvedeny velkými písmeny; například MQOO_OUTPUT. Sada konstant se stejnou předponou je zobrazena takto: MQIA_ *. Hodnotu konstanty viz "Konstanty" na stránce 61 .

Pole
V některých voláních jsou parametry pole znakových řetězců, které nemají pevnou velikost.
V popisech těchto parametrů představuje malá písmena n číselnou konstantu. Při kódování deklarace pro tento parametr nahraďte hodnotu n požadovanou číselnou hodnotou.

Použití volání v jazyce C

Parametry, které jsou *pouze vstupní* a typu MQHCONN, MQHOBJ, MQHMSG nebo MQLONG, jsou předávány hodnotou. Pro všechny ostatní parametry je *adresa* parametru předána hodnotou.

Nemusíte zadávat všechny parametry, které jsou předávány adresou při každém vyvolání funkce. Kde nepotřebujete konkrétní parametr, uveďte ukazatel na hodnotu null jako parametr při vyvolání funkce místo adresy dat parametru. Parametry, pro které je to možné, jsou identifikovány v popisech volání.

Jako hodnota volání není vrácen žádný parametr; v terminologii C to znamená, že všechna volání vrátí hodnotu `void`.

Deklarování parametru vyrovnávací paměti

Volání **MQGET**, **MQPUT**a **MQPUT1** mají každý jeden parametr, který má nedefinovaný datový typ: parametr *Vyrovnávací paměť* . Pomocí tohoto parametru můžete odeslat a přijmout data zprávy aplikace.

Parametry tohoto řazení jsou zobrazeny v příkladech jazyka C jako pole MQBYTE. Tímto způsobem můžete deklarovat parametry, ale obvykle je vhodnější je deklarovat jako konkrétní strukturu, která popisuje rozvržení dat ve zprávě. Prototyp funkce deklaruje parametr jako ukazatel na prázdno, takže můžete zadat adresu libovolného druhu dat jako parametr při vyvolání volání.

Ukazatel-na-void je ukazatel na data nedefinovaného formátu. Je definován jako:

```
typedef void *PMQVOID;
```

MQBACK-Vrátit zpět změny

Volání MQBACK označuje správci front, že všechny zprávy, které se vyskytly od posledního synchronizačního bodu, mají být vráceny zpět.

Zprávy vložené jako součást pracovní jednotky jsou odstraněny; zprávy načtené jako součást pracovní jednotky jsou obnoveny do fronty.

- V systému z/OSje toto volání používáno pouze dávkovými programy (včetně IMS dávkových DL/I programů).

Syntaxe

MQBACK (*Hconn*, *Kód dokončení*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_OUTCOME_PENDING
(2124, X'84C') Výsledek operace vrácení je nevyřízený.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility je používána.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není v prostředí platné.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OUTCOME_MIXED (smíšený)
(2123, X'84B') Výsledek operace potvrzení nebo vrácení je smíšený.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny .

Poznámky k použití

1. Toto volání lze použít pouze v případě, že správce front sám koordinuje jednotku práce. Může se jednat o:
   - Lokální pracovní jednotka, kde změny ovlivňují pouze prostředky produktu MQ .
   - Globální jednotka práce, v níž mohou změny ovlivnit prostředky patřící jiným správcům prostředků a prostředky produktu MQ .

   Další podrobnosti o lokálních a globálních jednotkách práce viz “MQBEGIN-Začátek transakce” na stránce 634.
2. V prostředích, kde správce front nekoordinuje pracovní jednotku, použijte místo MQBACK příslušné zpětné volání. Prostředí může také podporovat implicitní vrácení zpět způsobené nestandardním ukončením aplikace.
   - V systému z/OSpoužijte následující volání:

- Dávkové programy (včetně IMS dávkových programů DL/I) mohou používat volání MQBACK, pokud pracovní jednotka ovlivňuje pouze prostředky MQ . Pokud však pracovní jednotka ovlivňuje prostředky produktu MQ i prostředky patřící jiným správcům prostředků (například Db2), použijte volání SRRBACK poskytované službou RRS (z/OS Obnovitelný Resource Service). Volání SRRBACK vrací změny prostředků náležejících správcům prostředků, kteří mají povolenou koordinaci RRS.
- Aplikace systému CICS musí použít příkaz `EXEC CICS SYNCPOINT ROLLBACK` k vrácení pracovní jednotky zpět. Nepoužívejte volání MQBACK pro aplikace CICS .
- Aplikace IMS (jiné než dávkové programy DL/I) musí používat IMS volání, jako např. `ROLB` , aby se vrátila jednotka práce. Nepoužívejte volání MQBACK pro aplikace IMS (jiné než dávkové programy DL/I).

- V systému IBM ipoužijte toto volání pro lokální pracovní jednotky koordinované správcem front. To znamená, že na úrovni úlohy nesmí existovat definice vázaného zpracování, to znamená, že pro úlohu nesmí být vydán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** .

3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, odebrání těchto změn závisí na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v části “MQDISC-Odpojit správce front” na stránce 686 .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, správce front uchovává informace týkající se skupiny zpráv a logické zprávy pro poslední úspěšná volání MQPUT a MQGET. Tyto informace jsou přidruženy k popisovači fronty a zahrnují například:
   - Hodnoty polí *GroupId*, *MsgSeqNumber*, *Offset*a *MsgFlags* v deskriptoru MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo dočasná.

   Správce front uchovává *tři* sady informací o skupinách a segmentech, jednu sadu pro každou z následujících položek:

   - Poslední úspěšné volání MQPUT (může být součástí pracovní jednotky).
   - Poslední úspěšné volání MQGET, které odebrala zprávu z fronty (může být součástí pracovní jednotky).
   - Poslední úspěšné volání MQGET, které procházelo zprávu ve frontě (nemůže být součástí pracovní jednotky).

5. Informace přidružené k volání MQGET jsou obnoveny na hodnotu, kterou měly před prvním úspěšným voláním MQGET pro daný manipulátor fronty v aktuální transakci.

   Fronty, které byly aktualizovány aplikací po spuštění jednotky práce, ale mimo rozsah jednotky práce, nemají obnoveny informace o skupinách a segmentech, pokud je jednotka práce odvolána.

   Obnova informací o skupině a segmentu na předchozí hodnotu, když je jednotka práce odvolána, umožňuje aplikaci rozložit velkou skupinu zpráv nebo velkou logickou zprávu sestávající z mnoha segmentů do několika jednotek práce a restartovat ji ve správném bodě ve skupině zpráv nebo logické zprávě, pokud jedna z jednotek práce selže.

   Použití několika pracovních jednotek může být výhodné, pokud má lokální správce front pouze omezené úložiště front. Aplikace však musí udržovat dostatečné informace, aby mohla znovu spustit vkládání nebo získávání zpráv ve správném bodě, dojde-li k selhání systému.

   Podrobnosti o tom, jak provést restart ve správném bodě po selhání systému, viz volba MQPMO_LOGICAL_ORDER popsaná v části “MQPMO-Volby vložení zprávy” na stránce 502a volba MQGMO_LOGICAL_ORDER popsaná v části “MQGMO-Volby získání zprávy” na stránce 371.

   Zbývající poznámky k použití se použijí pouze v případě, že správce front koordinuje jednotky práce.
6. Jednotka práce má stejný rozsah jako manipulátor připojení. Všechna volání produktu MQ , která ovlivňují konkrétní transakci, musí být provedena pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného manipulátoru připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu popisovačů připojení viz parametr **Hconn** popsaný v části “MQCONN-Připojení správce front” na stránce 663 .

7. Toto volání ovlivní pouze zprávy, které byly vloženy nebo načteny jako součást aktuální pracovní jednotky.
8. Přerušitelná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale nikdy nevydá volání potvrzení nebo odvolání, může zaplnit fronty zprávami, které nejsou k dispozici pro jiné aplikace. Chcete-li se vyhnout této možnosti, administrátor musí nastavit atribut správce front **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila tomu, že by běžné aplikace zaplňovaly fronty, ale dostatečně vysoká, aby umožňovala správné fungování očekávaných aplikací systému zpráv.

Vyvolání jazyka C

```
MQBACK (Hconn, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

Vyvolání COBOL

```
CALL 'MQBACK' USING HCONN, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQBACK (Hconn, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
dcl Hconn     fixed bin(31);  /* Connection handle */
dcl CompCode  fixed bin(31);  /* Completion code */
dcl Reason    fixed bin(31);  /* Reason code qualifying CompCode */
```

Vyvolání High Level Assembler

```
CALL MQBACK,(HCONN,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Vyvolání jazyka Visual Basic

```
MQBACK Hconn, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQBEGIN-Začátek transakce

Volání MQBEGIN zahájí pracovní jednotku, která je koordinována správcem front a která může zahrnovat externí správce prostředků.

Syntaxe

MQBEGIN (*Hconn*, *BeginOptions*, *Compcode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

Hconn musí být nesdílený manipulátor připojení. Je-li uveden sdílený manipulátor připojení, volání selže s kódem příčiny MQRC_HCONN_ERROR. Další informace o sdílených a nesdílených popisovačích naleznete v popisu voleb MQCNO_HANDLE_SHARE_ * v souboru “MQCNO-Volby připojení” na stránce 318 .

BeginOptions
Typ: MQBO-vstup/výstup

Jedná se o volby, které řídí akci MQBEGIN, jak je popsáno v tématu “MQBO-volby zahájení” na stránce 281.

Nejsou-li vyžadovány žádné volby, mohou programy napsané v jazyku C nebo sestavovacím modulu S/390 zadat adresu parametru s hodnotou Null namísto adresy struktury MQBO.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_NO_EXTERNAL_ÚČASTNÍKŮ
(2121, X'849 ') Nejsou registrováni žádní zúčastnění správci prostředků.

MQRC_PARTICIPANT_NOT_AVAILABLE
(2122, X'84A') Participující správce prostředků není k dispozici.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_BO_ERROR
(2134, X'856 ') Struktura začátku voleb není platná.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není v prostředí platné.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UOW_IN_PROGRESS
(2128, X'850 ') Jednotka práce již byla spuštěna.

Další informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Pomocí volání MQBEGIN spusťte pracovní jednotku, která je koordinována správcem front a která může zahrnovat změny prostředků vlastněných jinými správci prostředků. Správce front podporuje tři typy pracovních jednotek:
   - **Lokální jednotka práce koordinovaná správcem front**: Jednotka práce, ve které je správce front jediným účastníkem správce prostředků, a správce front tedy vystupuje jako koordinátor jednotky práce.
     - Chcete-li spustit tento typ pracovní jednotky, uveďte volbu MQPMO_SYNCPOINT nebo MQGMO_SYNCPOINT pro první volání MQPUT, MQPUT1nebo MQGET v pracovní jednotce.
     - Chcete-li tento typ práce potvrdit nebo vrátit zpět, použijte volání MQCMIT nebo MQBACK.
   - **Pracovní jednotka koordinovaná správcem front**: Pracovní jednotka, ve které správce front vystupuje jako koordinátor jednotky práce, a to jak pro prostředky MQ *, tak pro prostředky* patřící jiným správcům prostředků. Tito správci prostředků spolupracují se správcem front, aby zajistili, že všechny změny prostředků v pracovní jednotce budou potvrzeny nebo vráceny zpět společně.
     - Chcete-li spustit tento typ pracovní jednotky, použijte volání MQBEGIN.
     - Chcete-li tento typ práce potvrdit nebo vrátit zpět, použijte volání MQCMIT a MQBACK.

- **Externí koordinovaná globální jednotka práce**: Jednotka práce, ve které je správce front účastníkem, ale správce front nejedná jako koordinátor jednotky práce. Místo toho existuje externí koordinátor pracovních jednotek, se kterým správce front spolupracuje.
  - Chcete-li spustit tento typ pracovní jednotky, použijte příslušné volání poskytnuté externím koordinátorem pracovní jednotky.

    Je-li při pokusu o spuštění pracovní jednotky použito volání MQBEGIN, volání se nezdaří s kódem příčiny MQRC_ENVIRONMENT_ERROR.
  - Chcete-li tento typ pracovní jednotky potvrdit nebo vrátit zpět, použijte volání potvrzení a vrácení poskytnutá externím koordinátorem pracovní jednotky.

    Pokud k potvrzení nebo vrácení transakce použijete volání MQCMIT nebo MQBACK, volání selže s kódem příčiny MQRC_ENVIRONMENT_ERROR.

2. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, odebrání těchto změn závisí na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v části "MQDISC-Odpojit správce front" na stránce 686 .
3. Aplikace se může v daném okamžiku účastnit pouze jedné jednotky práce. Volání MQBEGIN selže s kódem příčiny MQRC_UOW_IN_PROGRESS, pokud již pro aplikaci existuje jednotka práce, bez ohledu na to, o jaký typ práce se jedná.
4. Volání MQBEGIN je v prostředí klienta MQI produktu MQ neplatné. Pokus o použití volání selže s kódem příčiny MQRC_ENVIRONMENT_ERROR.
5. Pokud správce front vystupuje jako koordinátor jednotek práce pro globální jednotky práce, jsou správci prostředků, kteří se mohou podílet na jednotce práce, definováni v konfiguračním souboru správce front.
6. V systému IBM ijsou podporovány tři typy jednotek práce:
   - **Lokální pracovní jednotku koordinovanou správcem front** lze použít pouze v případě, že na úrovni úlohy neexistuje definice vázaného zpracování, tj. že pro úlohu nebyl zadán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** .
   - **Globální jednotka práce koordinovaná správcem front** není podporována.
   - Volbu **Externě koordinovaná globální jednotka práce** lze použít pouze v případě, že na úrovni úlohy existuje definice vázaného zpracování, to znamená, že pro úlohu musí být zadán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** . Pokud k tomu došlo, operace IBM i COMMIT a ROLLBACK se vztahují na prostředky produktu MQ i na prostředky patřící ostatním zúčastněným správcům prostředků.

Vyvolání jazyka C

```
MQBEGIN (Hconn, &BeginOptions, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN  Hconn;         /* Connection handle */
MQBO     BeginOptions;  /* Options that control the action of MQBEGIN */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

Vyvolání COBOL

```
CALL 'MQBEGIN' USING HCONN, BEGINOPTIONS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQBEGIN (Hconn, BeginOptions, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání jazyka Visual Basic

```
MQBEGIN Hconn, BeginOptions, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Převést vyrovnávací paměť na manipulátor zprávy

Volání funkce MQBUFMH převádí vyrovnávací paměť na manipulátor zprávy a je inverzní funkcí volání MQMHBUF.

Toto volání vezme deskriptor zprávy a vlastnosti MQRFH2 do vyrovnávací paměti a zpřístupní je prostřednictvím manipulátoru zprávy. Vlastnosti MQRFH2 v datech zprávy jsou volitelně odebrány. Pole *Encoding*, *CodedCharSetId*a *Format* deskriptoru zprávy jsou v případě potřeby aktualizována tak, aby správně popisovala obsah vyrovnávací paměti po odebrání vlastností.

Syntaxe

MQBUFMH (*Hconn*, *Hmsg*, *BufMsgHOpts*, *MsgDesc*, *BufferLength*, *Buffer*, *DataLength*, *Compcode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota **Hconn** musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být ustanoveno platné připojení pro podproces převádějící vyrovnávací paměť na popisovač zprávy. Není-li navázáno platné připojení, volání se nezdaří s hodnotou MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMQSG-vstup

Jedná se o popisovač zprávy, pro který je vyžadována vyrovnávací paměť. Hodnota byla vrácena předchozím voláním MQCRTMH.

BufMsgHOpts
Typ: MQBMHO-vstup

Struktura MQBMHO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou manipulátory zpráv vytvářeny z vyrovnávacích pamětí.

Podrobnosti viz "MQBMHO-Volby zpracování vyrovnávací paměti pro zprávu" na stránce 276.

MsgDesc
Typ: MQMD-vstup/výstup

Struktura *MsgDesc* obsahuje vlastnosti deskriptoru zpráv a popisuje obsah oblasti vyrovnávací paměti.

Při výstupu z volání jsou vlastnosti volitelně odebrány z oblasti vyrovnávací paměti a v tomto případě je deskriptor zprávy aktualizován tak, aby správně popisoval oblast vyrovnávací paměti.

Data v této struktuře musí být ve znakové sadě a kódování aplikace.

BufferLength
Typ: MQLONG-vstup

BufferLength je délka oblasti vyrovnávací paměti v bajtech.

Hodnota *BufferLength* s nulovým počtem bajtů je platná a označuje, že oblast vyrovnávací paměti neobsahuje žádná data.

Vyrovnávací paměť
Typ: MQBYTExBufferdélka-vstupní/výstupní

Jedná se o volby, které řídí akci MQBEGIN, jak je popsáno v tématu "MQBEGIN-Začátek transakce" na stránce 634.

Buffer definuje oblast obsahující vyrovnávací paměť zpráv. Pro většinu dat byste měli vyrovnávací paměť zarovnat na 4bajtovou hranici.

Pokud **Buffer** obsahuje znaková nebo číselná data, nastavte pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** na hodnoty odpovídající datům; to v případě potřeby umožní převod dat.

Pokud jsou ve vyrovnávací paměti zpráv nalezeny vlastnosti, jsou volitelně odebrány; později budou k dispozici v popisovači zprávy při návratu z volání.

V programovacím jazyku C je parametr deklarován jako ukazatel na prázdno, což znamená, že jako parametr lze zadat adresu libovolného typu dat.

Je-li parametr **BufferLength** nulový, **Buffer** se na něj neodkazuje; v tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo sestavovacím programem System/390 hodnotu null.

DataLength
Typ: MQLONG-výstup

Délka vyrovnávací paměti, která může mít odebrané vlastnosti, v bajtech.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BMHO_ERROR
(2489, X'09B9') Struktura voleb vyrovnávací paměti pro obsluhu zpráv je neplatná.

MQRC_BUFFER_ERROR
(2004, X'07D4') Parametr vyrovnávací paměti není platný.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

MQRC_HMSG_ERROR
(2460, X'099C') popisovač zprávy není platný.

MQRC_MD_ERROR
(2026, X'07EA') Deskriptor zpráv není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Obsluha zprávy je již používána.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_RFH_ERROR
(2334, X'091E') Struktura MQRFH2 není platná.

MQRC_RFH_FORMAT_ERROR
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nelze analyzovat.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

Volání MQBUFMH nelze zachytit prostřednictvím uživatelských procedur rozhraní API-vyrovnávací paměť je převedena na manipulátor zprávy v prostoru aplikace; volání nedosáhne správce front.

Vyvolání jazyka C

```
MQBUFMH (Hconn, Hmsg, &BufMsgHOpts, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQBUFMH' USING HCONN, HMSG, BUFMSGHOPTS, MSGDESC, BUFFERLENGTH,
                          BUFFER, DATALENGTH, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQBUFMH (Hconn, Hmsg, BufMsgHOpts, MsgDesc, BufferLength, Buffer,
DataLength, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQBUFMH,(HCONN,HMSG,BUFMSGHOPTS,MSGDESC,BUFFERLENGTH,BUFFER,
               DATALENGTH,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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-Spravovat zpětné volání

Volání MQCB registruje zpětné volání pro určený manipulátor objektu a řídí aktivaci a změny zpětného volání.

Zpětné volání je část kódu (určená buď jako název funkce, která může být dynamicky propojena, nebo jako ukazatel funkce), který je volán produktem IBM MQ , když dojde k určitým událostem.

Chcete-li použít MQCB a MQCTL na klientovi, musíte být připojeni k serveru, kde vyjednaný parametr **SHARECNV** kanálu souhlasil s nenulovou hodnotou.

Typy zpětného volání, které lze definovat, jsou:

Spotřebitel zpráv.
Funkce zpětného volání spotřebitele zpráv je volána, když je zpráva splňující zadaná kritéria výběru k dispozici na popisovači objektu.

Pro každý popisovač objektu lze registrovat pouze jednu funkci zpětného volání. Má-li být přečtena jedna fronta s více výběrovými kritérii, musí být fronta otevřena vícekrát a na každém popisovači musí být registrována funkce spotřebitele.

obslužná rutina událostí
Obslužná rutina událostí je volána pro podmínky, které ovlivňují celé prostředí zpětného volání.

Funkce je volána při výskytu podmínky události, například při zastavení nebo uvedení do klidového stavu správce front nebo připojení.

Funkce není volána pro podmínky, které jsou specifické pro jednoho spotřebitele zpráv, například MQRC_GET_INHIBITED; volá se však, pokud funkce zpětného volání neskončí normálně.

Syntaxe

MQCB (*Hconn*, *Operace*, *CallbackDesc*, *Hobj*, *MsgDesc*, *GetMsgOpts*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V systému z/OS pro aplikace CICS můžete zadat následující speciální hodnotu pro *MQHC_DEF_HCONN* , chcete-li použít manipulátor připojení přidružený k této prováděcí jednotce.

Operace
Typ: MQLONG-vstup

Operace, která se zpracovává na zpětném volání definovaném pro uvedený popisovač objektu. Musíte zadat jednu z následujících voleb. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

MQOP_REGISTER
Definujte funkci zpětného volání pro zadaný popisovač objektu. Tato operace definuje funkci, která se má volat, a kritéria výběru, která se mají použít.

Je-li pro manipulátor objektu již definována funkce zpětného volání, je definice nahrazena. Je-li zjištěna chyba při nahrazování zpětného volání, je zrušena registrace funkce.

Pokud je zpětné volání registrováno ve stejné funkci zpětného volání, ve které bylo dříve odregistrováno, je toto považováno za operaci nahrazení; žádná počáteční nebo konečná volání nejsou vyvolána.

Můžete použít MQOP_REGISTER s MQOP_SUSPEND nebo MQOP_RESUME.

MQOP_DEREGISTER
Zastavte spotřebu zpráv pro manipulátor objektu a odeberte manipulátor z těch, které jsou vhodné pro zpětné volání.

Pokud je přidružený popisovač zavřený, je automaticky zrušena registrace zpětného volání.

Je-li příkaz MQOP_DEREGISTER volán ze spotřebitele a pro zpětné volání je definováno volání zastavení, bude vyvolán při návratu ze spotřebitele.

Pokud je tato operace vydána pro *Hobj* bez registrovaného spotřebitele, volání se vrátí s hodnotou MQRC_CALLBACK_NOT_REGISTERED.

MQOP_SUSPEND
Pozastaví spotřebu zpráv pro popisovač objektu.

Pokud je tato operace použita na obslužnou rutinu událostí, obslužná rutina událostí nezíská události během pozastavení a všechny události, které zmeškaly v pozastaveném stavu, nejsou při obnovení operace poskytnuty.

Zatímco je funkce spotřebitele pozastavena, pokračuje v získání zpětných volání typu řízení.

MQOP_RESUME
Obnovte spotřebu zpráv pro popisovač objektu.

Pokud je tato operace použita na obslužnou rutinu událostí, obslužná rutina událostí nezíská události během pozastavení a všechny události, které zmeškaly v pozastaveném stavu, nejsou při obnovení operace poskytnuty.

CallbackDesc
Typ: MQCBD-vstup

Jedná se o strukturu, která identifikuje funkci zpětného volání, která je registrována aplikací, a volby použité při její registraci.

Podrobnosti o struktuře viz MQCBD .

Deskriptor zpětného volání je vyžadován pouze pro volbu MQOP_REGISTER; pokud deskriptor není požadován, předaná adresa parametru může být null.

HOBJ
Typ: MQHOBJ-vstup

Tento popisovač představuje přístup, který byl ustanoven k objektu, ze kterého má být zpráva využita. Jedná se o popisovač, který byl vrácen z předchozího volání MQOPEN nebo MQSUB (v parametru **Hobj**).

Parametr *Hobj* není vyžadován při definování rutiny obslužné rutiny událostí (MQCBT_EVENT_HANDLER) a měl by být určen jako MQHO_NONE.

Pokud byla z volání MQOPEN vrácena hodnota *Hobj* , musí být fronta otevřena s jednou nebo více z následujících voleb:

- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE
- MQOO_INPUT_AS_Q_DEF
- MQOO_BROWSE

MsgDesc
Typ: vstup MQMD

Tato struktura popisuje atributy požadované zprávy a atributy načtené zprávy.

Parametr **MsgDesc** definuje atributy zpráv požadovaných spotřebitelem a verzi MQMD, která má být předána spotřebiteli zpráv.

Volby *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber*a *Offset* v deskriptoru MQMD se používají pro výběr zpráv v závislosti na volbách zadaných v parametru **GetMsgOpts** .

Volby *Encoding* a *CodedCharSetId* se používají pro převod zpráv, pokud zadáte volbu MQGMO_CONVERT.

Podrobnosti viz MQMD .

MsgDesc se používá pro MQOP_REGISTER a pokud požadujete jiné hodnoty než výchozí pro libovolná pole. *MsgDesc* se nepoužívá pro obslužnou rutinu událostí.

Pokud deskriptor není požadován, předaná adresa parametru může být null.

Všimněte si, že pokud je pro stejnou frontu registrováno více spotřebitelů s překrývajícími se selektory, není vybraný spotřebitel pro každou zprávu definován.

GetMsgOpts
Typ: MQGMO-vstup

Parametr **GetMsgOpts** řídí způsob, jakým spotřebitel zpráv získává zprávy. Všechny volby tohoto parametru mají význam, jak je popsáno v tématu “MQGMO-Volby získání zprávy” na stránce 371, při použití ve volání MQGET, s výjimkou následujících:

MQGMO_SET_SIGNAL
Tato volba není povolena.

MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXT, MQGMO_MARK_ *
Pořadí zpráv doručených spotřebiteli procházení je určeno kombinacemi těchto voleb. Významné kombinace jsou:

MQGMO_BROWSE_FIRST
První zpráva ve frontě je opakovaně doručena spotřebiteli. To je užitečné, když spotřebitel destruktivně spotřebuje zprávu ve zpětném volání. Tuto volbu používejte opatrně.

MQGMO_BROWSE_NEXT
Odběratel obdrží každou zprávu ve frontě od aktuální pozice kurzoru až po dosažení konce fronty.

MQGMO_BROWSE_FIRST + MQGMO_BROWSE_NEXT
Kurzor se resetuje na začátek fronty. Odběratel pak obdrží každou zprávu, dokud kurzor nedosáhne konce fronty.

MQGMO_BROWSE_FIRST + MQGMO_MARK_ *
Počínaje začátkem fronty je spotřebiteli předána první neoznačená zpráva ve frontě, která je poté označena pro tohoto spotřebitele. Tato kombinace zajišťuje, že spotřebitel může přijímat nové zprávy přidané za aktuální bod kurzoru.

MQGMO_BROWSE_NEXT + MQGMO_MARK_ *
Počínaje pozicí kurzoru je spotřebiteli předána další neoznačená zpráva ve frontě, která je poté označena pro tohoto spotřebitele. Tuto kombinaci používejte opatrně, protože zprávy lze přidat do fronty za aktuální pozici kurzoru.

MQGMO_BROWSE_FIRST + MQGMO_BROWSE_NEXT + MQGMO_MARK_ *
Tato kombinace není povolena. Je-li použito, volání vrátí hodnotu MQRC_OPTIONS_ERROR.

MQGMO_NO_WAIT, MQGMO_WAIT a WaitInterval
Tyto volby řídí způsob vyvolání spotřebitele.

MQGMO_NO_WAIT
Spotřebitel není nikdy volán s MQRC_NO_MSG_AVAILABLE. Spotřebitel je volán pouze pro zprávy a události.

MQGMO_WAIT s nulovou hodnotou WaitInterval
Kód MQRC_NO_MSG_AVAILABLE je předán spotřebiteli, když nejsou k dispozici žádné zprávy, a buď byl spotřebitel spuštěn, nebo byl odběratel doručen alespoň jednu zprávu od posledního kódu příčiny "žádné zprávy".

To zabrání odběrateli v systému výzev v vytížené smyčce, když je uveden nulový interval čekání.

MQGMO_WAIT a kladný WaitInterval
Odběratel je volán po uvedeném intervalu čekání s kódem příčiny MQRC_NO_MSG_AVAILABLE. Toto volání se provádí bez ohledu na to, zda byly spotřebiteli doručeny nějaké zprávy. To umožňuje uživateli provést zpracování prezenčního signálu nebo dávkového zpracování.

MQGMO_WAIT a WaitInterval parametru MQWI_UNLIMITED
Tato volba určuje nekonečné čekání před vrácením MQRC_NO_MSG_AVAILABLE. Spotřebitel není nikdy volán s MQRC_NO_MSG_AVAILABLE.

Parametr *GetMsgOpts* se používá pouze pro MQOP_REGISTER a v případě, že pro některá pole požadujete jiné hodnoty než výchozí. *GetMsgOpts* se nepoužívá pro obslužnou rutinu událostí.

Není-li parametr *GetMsgOpts* povinný, předaná adresa parametru může být null. Použití tohoto parametru je stejné jako určení MQGMO_DEFAULT spolu s MQGMO_FAIL_IF_QUIESCING.

Je-li ve struktuře MQGMO poskytnut popisovač vlastností zprávy, je kopie poskytnuta ve struktuře MQGMO, která je předána do zpětného volání spotřebitele. Při návratu z volání MQCB může aplikace odstranit popisovač vlastností zprávy.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kódy příčiny v následujícím seznamu jsou ty, které může správce front vrátit pro parametr **Reason** .

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855 ') Nelze načíst moduly služeb převodu dat.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CALLBACK_LINK_ERROR
(2487, X'9B7') Nesprávné pole typu zpětného volání.

MQRC_CALLBACK_NOT_REGISTERED
(2448, X' 990 ') Nelze zrušit registraci, pozastavit nebo obnovit, protože neexistuje žádné registrované zpětné volání.

MQRC_CALLBACK_ROUTINE_ERROR
(2486, X'9B6') Musí být uveden buď *CallbackFunction* , nebo *CallbackName* , ale ne obojí.

MQRC_CALLBACK_TYPE_ERROR
(2483, X'9B3') Nesprávné pole typu zpětného volání.

MQRC_CBD_OPTIONS_ERROR
(2484, X'9B4') Chybné pole voleb MQCBD.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Není autorizováno pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Uklidení připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_CORREL_ID_ERROR
(2207, X'89F') Chyba identifikátoru korelace.

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není v prostředí platné.

MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

MQRC_GET_INHIBITED
(2016, X'7E0') Získá blokováno pro frontu.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální konflikt pracovních jednotek.

MQRC_GMO_ERROR
(2186, X'88A') Struktura voleb Get-message není platná.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Rukojeť používaná pro globální jednotku práce.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOBJ_ERROR
(2019, X'7E3') popisovač objektu není platný.

MQRC_NEKONZISTENTNÍ_PROCHÁZENÍ
(2259, X'8D3') Nekonzistentní specifikace procházení.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

MQRC_INVALID_MSG_UNDER_CURSOR
(2246, X'8C6') Zpráva pod kurzorem není platná pro načtení.

MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s lokální jednotkou práce.

MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Volby shody nejsou platné.

MQRC_MAX_MSG_LENGTH_ERROR
(2485, X'9B4') Chybné *MaxMsgLength* pole.

MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zpráv není platný.

MQRC_MODULE_ENTRY_NOT_FOUND
(2497, X'9C1') Zadaný vstupní bod funkce nebyl v modulu nalezen.

MQRC_MODULE_INVALID
(2496, X'9C0') Modul byl nalezen, ale je nesprávného typu; není 32bitový, 64bitový nebo má platnou knihovnu dynamického odkazu.

MQRC_MODULE_NOT_FOUND
(2495, X'9BF') Modul nebyl nalezen ve vyhledávací cestě nebo není autorizován k načtení.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Pořadové číslo zprávy není platné.

MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Použití tokenu zprávy není platné.

MQRC_NO_MSG_AVAILABLE
(2033, X'7F1') Není k dispozici žádná zpráva.

MQRC_NO_MSG_UNDER_CURSOR
(2034, X'7F2') Kurzor procházení není umístěn na zprávě.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') Fronta není otevřena pro procházení.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') Fronta není otevřena pro vstup.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu se od doby otevření změnila.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OPERATION_ERROR
(2206, X'89E') Chybný kód operace ve volání rozhraní API.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

MQRC_Q_DELETED
(2052, X'804 ') Fronta byla odstraněna.

MQRC_Q_INDEX_TYPE_ERROR
(2394, X'95A') Fronta má chybný typ indexu.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uveden do klidového stavu.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_SIGNAL_NEVYŘÍZENÉ
(2069, X'815 ') Pro tuto rukojeť je signál vynikající.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

MQRC_SYNCPOINT_LIMIT_DOSAŽENO
(2024, X'7E8') V rámci aktuální pracovní jednotky nelze zpracovat žádné další zprávy.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora synchronizačního bodu není k dispozici.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Registrace v globální pracovní jednotce se nezdařila.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotky práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Interval čekání v MQGMO není platný.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Dodána nesprávná verze MQGMO.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Dodána nesprávná verze MQMD.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. MQCB se používá k definování akce, která má být vyvolána pro každou zprávu, odpovídající zadaným kritériím, která je k dispozici ve frontě. Když je akce zpracována, buď je zpráva odebrána z fronty a předána definovanému spotřebiteli zpráv, nebo je poskytnut token zprávy, který se používá k načtení zprávy.
2. Funkci MQCB lze použít k definování rutin zpětného volání před spuštěním spotřeby s funkcí MQCTL nebo ji lze použít z rutiny zpětného volání.
3. Chcete-li používat funkci MQCB mimo rutinu zpětného volání, musíte nejprve pozastavit spotřebu zpráv pomocí funkce MQCTL a poté obnovit spotřebu.
4. Modul MQCB není v rámci adaptéru IMS podporován.

Posloupnost zpětného volání spotřebitele zpráv

Spotřebitele můžete nakonfigurovat tak, aby vyvolal zpětné volání v klíčových bodech během životního cyklu spotřebitele. Příklad:

- kdy je spotřebitel poprvé registrován,
- když je připojení spuštěno,
- když je připojení zastaveno a
- když je odběratel odregistrován, buď explicitně, nebo implicitně pomocí MQCLOSE.

Tabulka 541. Definice příkazových slov MQCTL	
Sloveso	**Význam**
MQCTL (začátek)	Volání MQCTL pomocí operace MQOP_START
MQCTL (STOP)	Volání MQCTL pomocí operace MQOP_STOP
MQCTL (WAIT)	Volání MQCTL pomocí operace MQOP_START_WAIT

To umožňuje spotřebiteli udržovat stav přidružený ke spotřebiteli. Když aplikace požaduje zpětné volání, pravidla pro vyvolání spotřebitele jsou následující:

Registrovat
Jedná se vždy o první typ vyvolání zpětného volání.

Je vždy voláno ve stejném podprocesu jako volání MQCB (REGISTER).

SPUSTIT
Je vždy volána synchronně se slovesem MQCTL (START).

- Všechna zpětná volání START jsou dokončena před vrácením příkazu MQCTL (START).

Je ve stejném podprocesu jako doručení zprávy, pokud je požadováno THREAD_AFFINITY.

Volání se spuštěním není zaručeno, pokud například předchozí zpětné volání během MQCTL (START) vydá příkaz MQCTL (STOP).

ZASTAVIT
Po tomto volání nebudou doručeny žádné další zprávy nebo události, dokud nebude připojení restartováno.

Příkaz STOP je zaručen v případě, že aplikace byla dříve volána pro příkaz START, zprávu nebo událost.

DEREGISTER (REGISTROVAT)
Jedná se vždy o poslední typ vyvolání zpětného volání.

Ujistěte se, že vaše aplikace provádí inicializaci založenou na podprocesech a vyčištění ve zpětných voláních START a STOP. Pomocí zpětných volání REGISTER a DEREGISTER můžete provést inicializaci a vyčištění bez použití podprocesů.

Nedělejte žádné jiné předpoklady o životnosti a dostupnosti závitu než to, co je uvedeno. Nespoléhejte se například na to, že podproces zůstane aktivní i po posledním volání příkazu DEREGISTER. Podobně, pokud jste se rozhodli nepoužívat THREAD_AFFINITY, nepředpokládejte, že vlákno existuje při každém spuštění připojení.

Pokud má vaše aplikace konkrétní požadavky na charakteristiku podprocesu, může vždy vytvořit podproces odpovídajícím způsobem a poté použít MQCTL (WAIT). To má za následek 'darování' podprocesu produktu IBM MQ pro asynchronní doručování zpráv.

Využití připojení spotřebitele zpráv

Spotřebitele můžete nakonfigurovat tak, aby vyvolal zpětné volání v klíčových bodech během životního cyklu spotřebitele. Příklad:

- kdy je spotřebitel poprvé registrován,
- když je připojení spuštěno,
- když je připojení zastaveno a
- když je odběratel odregistrován, buď explicitně, nebo implicitně pomocí MQCLOSE.

Tabulka 542. Definice příkazových slov MQCTL	
Sloveso	**Význam**
MQCTL (začátek)	Volání MQCTL pomocí operace MQOP_START
MQCTL (STOP)	Volání MQCTL pomocí operace MQOP_STOP

Tabulka 542. Definice příkazových slov MQCTL (pokračování)	
Sloveso	**Význam**
MQCTL (WAIT)	Volání MQCTL pomocí operace MQOP_START_WAIT

To umožňuje spotřebiteli udržovat stav přidružený ke spotřebiteli. Když aplikace požaduje zpětné volání, pravidla pro vyvolání spotřebitele jsou následující:

Registrovat
Jedná se vždy o první typ vyvolání zpětného volání.

Je vždy voláno ve stejném podprocesu jako volání MQCB (REGISTER).

SPUSTIT
Je vždy volána synchronně se slovesem MQCTL (START).

- Všechna zpětná volání START jsou dokončena před vrácením příkazu MQCTL (START).

Je ve stejném podprocesu jako doručení zprávy, pokud je požadováno THREAD_AFFINITY.

Volání se spuštěním není zaručeno, pokud například předchozí zpětné volání během MQCTL (START) vydá příkaz MQCTL (STOP).

ZASTAVIT
Po tomto volání nebudou doručeny žádné další zprávy nebo události, dokud nebude připojení restartováno.

Příkaz STOP je zaručen v případě, že aplikace byla dříve volána pro příkaz START, zprávu nebo událost.

DEREGISTER (REGISTROVAT)
Jedná se vždy o poslední typ vyvolání zpětného volání.

Ujistěte se, že vaše aplikace provádí inicializaci založenou na podprocesech a vyčištění ve zpětných voláních START a STOP. Pomocí zpětných volání REGISTER a DEREGISTER můžete provést inicializaci a vyčištění bez použití podprocesů.

Nedělejte žádné jiné předpoklady o životnosti a dostupnosti závitu než to, co je uvedeno. Nespoléhejte se například na to, že podproces zůstane aktivní i po posledním volání příkazu DEREGISTER. Podobně, pokud jste se rozhodli nepoužívat THREAD_AFFINITY, nepředpokládejte, že vlákno existuje při každém spuštění připojení.

Pokud má vaše aplikace konkrétní požadavky na charakteristiku podprocesu, může vždy vytvořit podproces odpovídajícím způsobem a poté použít MQCTL (WAIT). To má za následek 'darování' podprocesu produktu IBM MQ pro asynchronní doručování zpráv.

Vyvolání jazyka C

```
MQCB (Hconn, Operation, CallbackDesc, Hobj, MsgDesc,
GetMsgOpts, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQCB' USING HCONN, OPERATION, CBDESC, HOBJ, MSGDESC,
           GETMSGOPTS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQCB(Hconn, Operation, CallbackDesc, Hobj, MsgDesc, GetMsgOpts,
          CompCode, Reason)
```

Deklarujte parametry následujícím způsobem:

```
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-Funkce zpětného volání

Volání funkce MQCB_FUNCTION je funkcí zpětného volání pro obsluhu událostí a asynchronní spotřebu zpráv.

Definice volání MQCB_FUNCTION je určena výhradně pro popis parametrů předávaných funkci zpětného volání. Správce front neposkytl žádný vstupní bod s názvem MQCB_FUNCTION.

Specifikace skutečné funkce, která má být volána, je vstupem pro volání MQCB a je předávána prostřednictvím struktury MQCBD .

Syntaxe

MQCB_FUNCTION (*Hconn*, *MsgDesc*, *GetMsgOpts*, *Buffer*, *Context*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX. V systému z/OS pro aplikace CICS lze vynechat volání MQCONN a pro připojení Hconn zadat následující hodnotu:

MQHC_DEF_CONN

Výchozí manipulátor připojení.

MsgDesc
Typ: vstup MQMD

Tato struktura popisuje atributy načtené zprávy.

Podrobnosti viz “MQMD-Deskriptor zpráv” na stránce 424.

Předaná verze MQMD je stejná jako verze předaná ve volání MQCB, které definovaly funkci spotřebitele.

Adresa MQMD je předána jako znaky null, pokud byl použit MQGMO verze 4 k vyžádání vrácení popisovače zprávy místo MQMD.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny událostí.

GetMsgOpty (volby)
Typ: MQGMO-vstup

Volby používané k řízení akcí spotřebitele zpráv. Tento parametr také obsahuje další informace o vrácené zprávě.

Podrobnosti viz MQGMO .

Předaná verze MQGMO je nejnovější podporovaná verze.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny událostí.

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-vstup

Toto je oblast obsahující data zprávy.

Pokud pro toto volání není k dispozici žádná zpráva nebo pokud zpráva neobsahuje žádná data zprávy, je adresa *Buffer* předána jako hodnoty null.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny událostí.

Kontext
Typ: MQCBC-vstup/výstup

Tato struktura poskytuje informace o kontextu pro funkce zpětného volání. Podrobnosti viz “MQCBC-Kontext zpětného volání” na stránce 283.

Poznámky k použití

1. Uvědomte si, že pokud vaše rutiny zpětného volání používají služby, které by mohly zpozdit nebo zablokovat podproces, například MQGET s čekáním, mohou zpozdit odbavení ostatních zpětných volání.
2. Pro každé vyvolání rutiny zpětného volání není automaticky vytvořena samostatná pracovní jednotka, takže rutiny mohou buď vydat volání potvrzení, nebo odložit potvrzení, dokud nebude zpracována logická dávka práce. Když je dávka práce potvrzena, potvrdí zprávy pro všechny funkce zpětného volání, které byly vyvolány od posledního synchronizačního bodu.
3. Programy vyvolané parametry načtení CICS LINK nebo CICS START pomocí služeb CICS prostřednictvím pojmenovaných objektů známých jako kontejnery kanálů. Názvy kontejnerů jsou stejné jako názvy parametrů. Další informace naleznete v dokumentaci k produktu CICS .
4. Rutiny zpětného volání mohou vyvolat volání MQDISC, nikoli však pro vlastní připojení. Pokud například rutina zpětného volání vytvořila připojení, může připojení také odpojit.

5. Rutina zpětného volání by obecně neměla spoléhat na to, že bude pokaždé vyvolána ze stejného podprocesu. V případě potřeby použijte při spuštění připojení parametr MQCTLO_THREAD_AFFINITY.
6. Když rutina zpětného volání obdrží nenulový kód příčiny, musí provést odpovídající akci.
7. MQCB_FUNCTION není v rámci adaptéru IMS podporován.

MQCLOSE-Zavřít objekt

Volání MQCLOSE znovu ukončí přístup k objektu a jedná se o inverzní volání MQOPEN a MQSUB.

Syntaxe

MQCLOSE (*Hconn*, *Hobj*, *Volby*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V aplikacích z/OS for CICS můžete vynechat volání MQCONN a zadat následující hodnotu pro *Hconn*:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

HOBJ
Typ: MQHOBJ-vstup/výstup

Tento popisovač představuje objekt, který se zavírá. Objekt může být libovolného typu. Hodnota *Hobj* byla vrácena předchozím voláním MQOPEN.

Po úspěšném dokončení volání správce front nastaví tento parametr na hodnotu, která není platným popisovačem pro dané prostředí. Tato hodnota je:

MQHO_UNUSABLE_HOBJ
Nepoužitelný popisovač objektu.

V systému z/OSje hodnota *Hobj* nastavena na hodnotu, která není definována.

Volby
Typ: MQLONG-vstup

Tento parametr řídí způsob zavření objektu.

Pouze trvalé dynamické fronty a odběry lze zavřít více než jedním způsobem, protože musí být buď zachovány, nebo odstraněny; jedná se o fronty s atributem **DefinitionType** , který má hodnotu MQQDT_PERMANENT_DYNAMIC (viz atribut **DefinitionType** popsaný v tématu “Atributy pro fronty” na stránce 836). Volby zavření jsou shrnuty v tomto tématu.

Trvalé odběry lze buď zachovat, nebo odebrat. Tyto odběry jsou vytvořeny pomocí volání MQSUB s volbou MQSO_TRVALÉ.

Při zavírání manipulátoru do spravovaného místa určení (tj. parametru **Hobj** vráceného při volání MQSUB, které použilo volbu MQSO_MANAGED) správce front vyčistí všechna publikování, která nebyla načtena při odebrání přidruženého odběru. Odběr je odebrán pomocí volby MQCO_REMOVE_SUB pro parametr **Hsub** vrácený při volání MQSUB. Poznámka: Parametr MQCO_REMOVE_SUB je výchozím chováním příkazu MQCLOSE pro netrvalý odběr.

Při zavírání manipulátoru do nespravovaného místa určení odpovídáte za vyčištění fronty, kam jsou odesílána publikování. Nejprve zavřete odběr pomocí příkazu MQCO_REMOVE_SUB a poté zprávy z fronty zpracujte, dokud nezůstanou žádné.

Jednu volbu musíte zadat pouze z následujícího:

Volby dynamické fronty: Tyto volby řídí způsob zavření trvalých dynamických front.

MQCO_DELETE
Fronta se odstraní, pokud je splněna některá z následujících podmínek:

- Jedná se o trvalou dynamickou frontu vytvořenou předchozím voláním MQOPEN a ve frontě nejsou žádné zprávy a neexistují žádné nepotvrzené nevyřízené požadavky get nebo put pro frontu (buď pro aktuální úlohu, nebo pro jinou úlohu).
- Jedná se o dočasnou dynamickou frontu vytvořenou voláním MQOPEN, které vrátilo hodnotu *Hobj*. V tomto případě jsou všechny zprávy ve frontě vymazány.

Ve všech ostatních případech, včetně případu, kdy byl příkaz *Hobj* vrácen při volání MQSUB, volání selže s kódem příčiny MQRC_OPTION_NOT_VALID_FOR_TYPE a objekt nebude odstraněn.

V systému z/OSplatí, že pokud je fronta dynamickou frontou, která byla logicky odstraněna, a jedná se o její poslední popisovač, je fronta fyzicky odstraněna. Další podrobnosti viz "Poznámky k použití" na stránce 657 .

MQCO_DELETE_PURGE
Fronta je odstraněna a všechny zprávy na ní jsou vymazány, pokud platí některá z následujících podmínek:

- Jedná se o trvalou dynamickou frontu vytvořenou předchozím voláním MQOPEN a pro tuto frontu neexistují žádné nepotvrzené požadavky get nebo put (buď pro aktuální úlohu, nebo pro jinou úlohu).
- Jedná se o dočasnou dynamickou frontu vytvořenou voláním MQOPEN, které vrátilo hodnotu *Hobj*.

Ve všech ostatních případech, včetně případu, kdy byl příkaz *Hobj* vrácen při volání MQSUB, volání selže s kódem příčiny MQRC_OPTION_NOT_VALID_FOR_TYPE a objekt nebude odstraněn.

Tabulka 543. Volby zavření pro různé typy objektů

Typ objektu nebo fronty	**MQCO_NONE**	**MQCO_DELETE**	**MQCO_DELETE_PURGE**
Jiný objekt než fronta	Zachováno	Neplatné	Neplatné
Předdefinovaná fronta	Zachováno	Neplatné	Neplatné
permanentní dynamická fronta	Zachováno	Odstraněno, pokud jsou prázdné a žádné nevyřízené aktualizace	Zprávy odstraněny; fronta odstraněna, pokud nejsou žádné nevyřízené aktualizace
Dočasná dynamická fronta (volání vydané tvůrcem fronty)	Odstraněno	Odstraněno	Odstraněno
Dočasná dynamická fronta (volání nebylo vydáno tvůrcem fronty)	Zachováno	Neplatné	Neplatné
Distribuční seznam	Zachováno	Neplatné	Neplatné
Cíl spravovaného odběru	Zachováno	Neplatné	Neplatné
Distribuční seznam (odběr byl odebrán)	Zprávy odstraněny; fronta odstraněna	Neplatné	Neplatné

Volby uzavření odběru: Tyto volby určují, zda mají být při zavření popisovače odebrány trvalé odběry a zda mají být vyčištěna publikování čekající na čtení aplikací. Tyto volby jsou platné pouze pro použití s popisovačem objektu vráceným v parametru **Hsub** volání MQSUB.

MQCO_KEEP_SUB
Popisovač pro odběr je uzavřen, ale provedený odběr je zachován. Publikování budou nadále odesílána do místa určení určeného v odběru. Tato volba je platná pouze v případě, že byl odběr proveden s volbou MQSO_TRVALÝ.

MQCO_KEEP_SUB je výchozí, pokud je odběr trvalý

MQCO_REMOVE_SUB
Odběr je odebrán a popisovač odběru je uzavřen.

Parametr **Hobj** volání MQSUB není zneplatněn uzavřením parametru **Hsub** a může být nadále používán pro MQGET nebo MQCB pro příjem zbývajících publikování. Je-li parametr **Hobj** volání MQSUB také zavřen, budou v případě, že se jednalo o spravované místo určení, odebrána všechna nenačtená publikování.

MQCO_REMOVE_SUB je výchozí, pokud je odběr netrvalý.

Úspěšné dokončení příkazu MQCO_REMOVE_SUB neznamená, že akce byla dokončena. Chcete-li zkontrolovat, zda bylo toto volání dokončeno, prohlédněte si krok DELETE SUB v části Kontrola dokončení asynchronních příkazů pro distribuované sítě.

Tyto volby uzavření odběru jsou shrnuty v následujících tabulkách.

Tabulka 544. Volby pro zavření popisovače trvalého odběru, ale zachování odběru

Úloha	**Volba uzavření odběru**
Zachovat publikování na popisovači MQOPENed	MQCO_KEEP_SUB
Odebrat publikování na popisovači MQOPENed	Akce není povolena
Zachovat publikování na popisovači MQSO_MANAGED	MQCO_KEEP_SUB
Odebrat publikování na popisovači MQSO_MANAGED	Akce není povolena

Chcete-li zrušit odběr, buď uzavřením popisovače trvalého odběru a jeho zrušením, nebo uzavřením popisovače netrvalého odběru, použijte následující volby uzavření odběru:

Tabulka 545. Volby pro zrušení odběru

Úloha	**Volba uzavření odběru**
Zachovat publikování na popisovači MQOPENed	MQCO_REMOVE_SUB
Odebrat publikování na popisovači MQOPENed	Akce není povolena
Zachovat publikování na popisovači MQSO_MANAGED	MQCO_REMOVE_SUB

Volby dopředného čtení: Následující volby řídí, co se stane s dočasnými zprávami, které byly odeslány klientovi před tím, než je aplikace vyžádala a dosud nebyly aplikací spotřebovány. Tyto zprávy jsou uloženy ve vyrovnávací paměti dopředného čtení klienta, která čeká na požadavek aplikace, a mohou být buď vyřazeny, nebo spotřebovány z fronty před dokončením operace MQCLOSE.

MQCO_IMMEDIATE
Objekt je okamžitě zavřen a všechny zprávy, které byly klientovi odeslány před tím, než je aplikace požadovala, jsou vyřazeny a nejsou k dispozici pro využití žádnou aplikací. Toto je výchozí hodnota.

MQCO_QUIESCE
Je vydán požadavek na zavření objektu, ale pokud jsou zprávy, které byly odeslány klientovi před tím, než je aplikace požadovala, stále uloženy ve vyrovnávací paměti dopředného čtení klienta, volání MQCLOSE se vrátí s varováním MQRC_READ_AHEAD_MSGS a popisovač objektu zůstane platný.

Aplikace pak může pokračovat v používání popisovače objektu k načtení zpráv, dokud nebudou k dispozici žádné další, a poté objekt znovu nezavře. Před aplikací, která je požaduje, nejsou klientovi odeslány žádné další zprávy, dopředné čtení je nyní vypnuto.

Aplikacím se doporučuje používat MQCO_QUIESCE místo toho, aby se pokusily dosáhnout bodu, ve kterém nejsou žádné další zprávy ve vyrovnávací paměti dopředného čtení klienta, protože mezi posledním voláním MQGET a následujícím voláním MQCLOSE by mohla být přijata zpráva, která by byla zahozena, kdyby bylo použito MQCO_IMMEDIATE.

Pokud je příkaz MQCLOSE s MQCO_QUIESCE vydán z funkce asynchronního zpětného volání, použije se stejné chování při čtení dopředných zpráv. Pokud je vráceno varování MQRC_READ_AHEAD_MSGS, je funkce zpětného volání volána alespoň jednou. Po předání poslední zbývající zprávy dopředné čtení funkci zpětného volání je pole MQCBC ConsumerFlags nastaveno na hodnotu MQCBCF_READA_BUFFER_EMPTY.

Výchozí volba: Pokud nevyžadujete žádnou z dříve popsaných voleb, můžete použít následující volbu:

MQCO_NONE
Není vyžadováno žádné volitelné zpracování zavření.

Toto musí být uvedeno pro:

- Jiné objekty než fronty
- Předdefinované fronty
- Dočasné dynamické fronty (ale pouze v těch případech, kdy *Hobj* není manipulátor vrácený voláním MQOPEN, které vytvořilo frontu).
- Distribuční seznamy

Ve všech výše uvedených případech je objekt zachován a není odstraněn.

Je-li tato volba zadána pro dočasnou dynamickou frontu:

- Fronta je odstraněna, pokud byla vytvořena voláním MQOPEN, které vrátilo hodnotu *Hobj* ; všechny zprávy, které jsou ve frontě, jsou vymazány.
- Ve všech ostatních případech se fronta (a všechny zprávy na ní) uchovávají.

Je-li tato volba uvedena pro trvalou dynamickou frontu, fronta se zachová a neodstraní.

V systému z/OSplatí, že pokud je fronta dynamickou frontou, která byla logicky odstraněna, a jedná se o její poslední popisovač, je fronta fyzicky odstraněna. Další podrobnosti viz "Poznámky k použití" na stránce 657 .

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Uvedené kódy příčiny jsou ty, které může správce front vrátit pro parametr **Reason** .

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není dokončena.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není dokončena.

MQRC_READ_AHEAD_MSGS
(nnnn, X'xxx ') Klient přečetl dopředné zprávy, které aplikace dosud nevyužila.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Zařízení pro spojku není k dispozici.

MQRC_CF_STRUC_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility je používána.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Není autorizováno pro připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Db2 subsystém není k dispozici.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOBJ_ERROR
(2019, X'7E3') popisovač objektu není platný.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OPTION_NOT_VALID_FOR_TYPE
(2045, X'7FD') Ve volání MQOPEN nebo MQCLOSE: volba není platná pro typ objektu.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_Q_NOT_EMPTY
(2055, X'807 ') Fronta obsahuje jednu nebo více zpráv nebo nepotvrzených požadavků na vložení nebo získání.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

Chyba MQRC_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Když aplikace zadá volání MQDISC nebo skončí buď normálně, nebo abnormálně, všechny objekty, které byly otevřeny aplikací a jsou stále otevřené, se automaticky zavřou pomocí volby MQCO_NONE.
2. Následující body platí v případě, že zavřený objekt je *fronta*:
   - Jsou-li operace ve frontě prováděny jako součást pracovní jednotky, lze frontu zavřít před nebo po výskytu synchronizačního bodu bez ovlivnění výsledku synchronizačního bodu. Je-li fronta spuštěna, může provedení odvolání před zavřením fronty způsobit vydání zprávy spouštěče. Další informace o zprávách spouštěče viz Vlastnosti zpráv spouštěče.
   - Pokud byla fronta otevřena s volbou MQOO_BROWSE, je kurzor procházení zničen. Pokud se fronta znovu otevře s volbou MQOO_BROWSE, vytvoří se nový kurzor procházení (viz MQOO_BROWSE).
   - Pokud je pro tento manipulátor v době volání MQCLOSE aktuálně uzamknuta zpráva, zámek se uvolní (viz MQGMO_LOCK).
   - Pokud v systému z/OSexistuje požadavek MQGET s nevyřízenou volbou MQGMO_SET_SIGNAL pro zavřenou manipulátor fronty, požadavek se zruší (viz MQGMO_SET_SIGNAL). Požadavky na signál pro stejnou frontu, ale podané pro různé popisovače (*Hobj*) nejsou ovlivněny (pokud není odstraněna dynamická fronta, v takovém případě jsou také zrušeny).
3. Následující body platí v případě, že zavřený objekt je *dynamická fronta* (trvalá nebo dočasná):
   - Pro dynamickou frontu můžete určit volby MQCO_DELETE a MQCO_DELETE_PURGE bez ohledu na volby zadané v odpovídajícím volání MQOPEN.
   - Když je dynamická fronta odstraněna, všechna volání MQGET s volbou MQGMO_WAIT, která jsou vůči frontě nevyřízená, jsou zrušena a vrátí se kód příčiny MQRC_Q_DELETED. Viz MQGMO_WAIT.

     Ačkoli aplikace nemohou přistupovat k odstraněné frontě, fronta není odebrána ze systému a přidružené prostředky nejsou uvolněny, dokud nebudou všechny popisovače, které odkazují na frontu, uzavřeny a všechny jednotky práce, které mají vliv na frontu, buď potvrzeny, nebo vráceny zpět.

     V systému z/OSfronta, která byla logicky odstraněna, ale ještě nebyla odebrána ze systému, brání vytvoření nové fronty se stejným názvem jako odstraněná fronta; volání MQOPEN v tomto případě selže s kódem příčiny MQRC_NAME_IN_USE. Taková fronta může být také stále zobrazena pomocí příkazů MQSC, i když k ní aplikace nemají přístup.

- Pokud při odstranění trvalé dynamické fronty není manipulátor *Hobj* určený ve volání MQCLOSE vrácen voláním MQOPEN, které vytvořilo frontu, je provedena kontrola, zda je identifikátor uživatele použitý k ověření volání MQOPEN autorizován k odstranění fronty. Pokud byla ve volání MQOPEN zadána volba MQOOO_ALTERNATE_USER_AUTHORITY, bude jako identifikátor uživatele použita hodnota *AlternateUserId*.

  Tato kontrola se neprovádí, pokud:

  – Uvedený popisovač je ten, který vrátil volání MQOPEN, které vytvořilo frontu.

  – Odstraňovaná fronta je dočasná dynamická fronta.
- Pokud je při zavření dočasné dynamické fronty manipulátor *Hobj* určený ve volání MQCLOSE vrácen voláním MQOPEN, které vytvořilo frontu, bude fronta odstraněna. K tomu dochází bez ohledu na volby zavření zadané ve volání MQCLOSE. Pokud jsou ve frontě zprávy, jsou vyřazeny; žádné zprávy sestavy se nevygenerují.

  Pokud existují nepotvrzené jednotky práce, které ovlivňují frontu, fronta a její zprávy jsou stále odstraněny, ale jednotky práce neselhávají. Avšak, jak bylo popsáno dříve, prostředky přidružené k jednotkám práce se neuvolní, dokud nebudou všechny jednotky práce buď potvrzeny, nebo vráceny zpět.

4. Následující body platí, pokud je uzavírána *distribuční seznam*:
   - Jedinou platnou volbou zavření pro distribuční seznam je MQCO_NONE; volání se nezdaří s kódem příčiny MQRC_OPTIONS_ERROR nebo MQRC_OPTION_NOT_VALID_FOR_TYPE, pokud jsou zadány jiné volby.
   - Když je distribuční seznam uzavřen, individuální kódy dokončení a kódy příčiny nejsou vráceny pro fronty v seznamu; pro diagnostické účely jsou k dispozici pouze parametry **CompCode** a **Reason** volání.

     Pokud dojde k selhání při zavírání jedné z front, správce front pokračuje ve zpracování a pokusí se zavřít zbývající fronty v distribučním seznamu. Parametry **CompCode** a **Reason** volání jsou nastaveny tak, aby vracely informace popisující selhání. Kód dokončení může být MQCC_FAILED, i když většina front byla úspěšně uzavřena. Fronta, která zjistila chybu, není identifikována.

     Pokud dojde k selhání ve více než jedné frontě, není definováno, které selhání je ohlášeno v parametrech **CompCode** a **Reason** .

Vyvolání jazyka C

```
MQCLOSE (Hconn, &Hobj, Options, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
     CALL 'MQCLOSE' USING HCONN, HOBJ, OPTIONS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQCLOSE (Hconn, Hobj, Options, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQCLOSE,(HCONN,HOBJ,OPTIONS,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Vyvolání jazyka Visual Basic

```
MQCLOSE Hconn, Hobj, Options, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Potvrdit změny

Volání MQCMIT označuje správci front, že aplikace dosáhla synchronizačního bodu a že všechny zprávy, které byly od posledního synchronizačního bodu vygenerovány a vloženy, mají být trvalé.

Zprávy vkládané jako součást pracovní jednotky jsou zpřístupněny jiným aplikacím; zprávy načtené jako součást pracovní jednotky jsou odstraněny.

- z/OS V systému z/OSje volání používáno pouze dávkovými programy (včetně IMS dávkových DL/I programů).

Syntaxe

MQCMIT (*Hconn*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Uvedené kódy příčiny jsou ty, které může správce front vrátit pro parametr **Reason** .

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_BACKED_OUT
(2003, X'7D3') Jednotka práce byla vrácena zpět.

MQRC_OUTCOME_PENDING
(2124, X'84C') Výsledek operace potvrzení je nevyřízený.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CALL_PŘERUŠENO
(2549, X'9F5') Zpracování příkazu MQPUT nebo MQCMIT bylo přerušeno a zpracování opětovného připojení nemůže znovu vytvořit určitý výsledek.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility je používána.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není v prostředí platné.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OUTCOME_MIXED (smíšený)
(2123, X'84B') Výsledek operace potvrzení nebo vrácení je smíšený.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RECONNECT_FAILED
(2548, X'9F4') Po opětovném připojení došlo k chybě při obnovování popisovačů pro znovu připojitelné připojení.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Toto volání použijte pouze v případě, že správce front sám koordinuje jednotku práce. Může se jednat o:
   - Lokální pracovní jednotka, kde změny ovlivňují pouze prostředky IBM MQ .
   - Globální jednotka práce, kde změny mohou ovlivnit prostředky patřící jiným správcům prostředků a prostředky IBM MQ .

   Další podrobnosti o lokálních a globálních jednotkách práce viz “MQBEGIN-Začátek transakce” na stránce 634.
2. V prostředích, kde správce front nekoordinuje pracovní jednotku, musí být místo MQCMIT použito příslušné volání potvrzení. Prostředí může také podporovat implicitní potvrzení způsobené normálním ukončením aplikace.
   - V systému z/OSpoužijte následující volání:
     - Dávkové programy (včetně IMS dávkových programů DL/I) mohou používat volání MQCMIT, pokud pracovní jednotka ovlivňuje pouze prostředky IBM MQ . Pokud však pracovní jednotka ovlivňuje jak prostředky IBM MQ , tak prostředky patřící jiným správcům prostředků (například Db2), použijte volání SRRCMIT poskytované službou z/OS RRS (Obnovitelný Resource Service). Volání SRRCMIT potvrdí změny prostředků náležejících správcům prostředků, kteří mají povolenou koordinaci RRS.
     - Aplikace CICS musí použít příkaz `EXEC CICS SYNCPOINT` k explicitnímu potvrzení transakce. Případně ukončení transakce vyústí v implicitní potvrzení transakce. Volání MQCMIT nelze použít pro aplikace CICS .
     - Aplikace IMS (jiné než dávkové programy DL/I) musí používat IMS volání, jako např. `GU` a `CHKP` , k potvrzení transakce. Volání MQCMIT nelze použít pro aplikace IMS (jiné než dávkové programy DL/I).
   - V systému IBM ipoužijte toto volání pro lokální pracovní jednotky koordinované správcem front. To znamená, že na úrovni úlohy nesmí existovat definice vázaného zpracování, to znamená, že pro úlohu nesmí být vydán příkaz STRCMTCTL s parametrem **`CMTSCOPE(*JOB)`** .
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, odebrání těchto změn závisí na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti viz Poznámky k použití MQDISC .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, správce front uchovává informace týkající se skupiny zpráv a logické zprávy pro poslední úspěšná volání MQPUT a MQGET. Tyto informace jsou přidruženy k popisovači fronty a zahrnují například:

- Hodnoty polí *GroupId*, *MsgSeqNumber*, *Offset*a *MsgFlags* v deskriptoru MQMD.
- Zda je zpráva součástí jednotky práce.
- Pro volání MQPUT: zda je zpráva trvalá nebo dočasná.

Po potvrzení transakce správce front zachová informace o skupině a segmentu a aplikace může pokračovat v vkládání nebo získávání zpráv do aktuální skupiny zpráv nebo logické zprávy.

Uchování informací o skupině a segmentu při potvrzení transakce umožňuje aplikaci rozložit velkou skupinu zpráv nebo velkou logickou zprávu sestávající z mnoha segmentů do několika pracovních jednotek. Použití několika pracovních jednotek je výhodné v případě, že lokální správce front má pouze omezené úložiště front. Aplikace však musí udržovat dostatek informací, aby mohla v případě selhání systému znovu spustit vkládání nebo získávání zpráv ve správném bodě. Podrobnosti o tom, jak provést restart ve správném bodě po selhání systému, viz MQPMO_LOGICAL_ORDER a MQGMO_LOGICAL_ORDER.

Zbývající poznámky k použití platí pouze v případě, že správce front koordinuje pracovní jednotky:

5. Jednotka práce má stejný rozsah jako manipulátor připojení; všechna volání IBM MQ , která ovlivňují konkrétní jednotku práce, musí být provedena pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného manipulátoru připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu popisovačů připojení viz parametr **Hconn** popsaný v MQCONN.
6. Toto volání ovlivní pouze zprávy, které byly vloženy nebo načteny jako součást aktuální pracovní jednotky.
7. Přerušitelná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale nikdy nevydá potvrzení nebo zpětné volání, může zaplnit fronty zprávami, které nejsou k dispozici pro jiné aplikace. Chcete-li se tomu vyhnout, administrátor musí nastavit atribut správce front **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránil tomu, že by běžné aplikace zaplňovaly fronty, ale dostatečně vysoká, aby umožňovala správné fungování očekávaných aplikací systému zpráv.
8. ALW Je-li v systémech AIX, Linux, and Windows parametr **Reason** MQRC_CONNECTION_BROKEN (s *CompCode* MQCC_FAILED) nebo MQRC_UNEXPECTED_ERROR, je možné, že jednotka práce byla úspěšně potvrzena.

Vyvolání jazyka C

```
MQCMIT (Hconn, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

Vyvolání COBOL

```
CALL 'MQCMIT' USING HCONN, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQCMIT (Hconn, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
dcl Hconn     fixed bin(31);  /* Connection handle */
dcl CompCode  fixed bin(31);  /* Completion code */
dcl Reason    fixed bin(31);  /* Reason code qualifying CompCode */
```

Vyvolání High Level Assembler

```
CALL MQCMIT,(HCONN,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Vyvolání jazyka Visual Basic

```
MQCMIT Hconn, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQCONN-Připojení správce front

Volání MQCONN připojí aplikační program ke správci front.

Poskytuje manipulátor připojení správce front, který aplikace používá při následných voláních fronty zpráv.

- V systému z/OSnemusí aplikace CICS toto volání vydávat. Tyto aplikace jsou automaticky připojeny ke správci front, ke kterému je připojen systém CICS . Volání MQCONN a MQDISC jsou však i nadále přijímána z aplikací CICS .
- V systému IBM imusí aplikace používat volání MQCONN nebo MQCONNX pro připojení ke správci front a volání MQDISC pro odpojení od správce front.

Připojení klienta nelze vytvořit pouze na instalaci serveru a lokální připojení nelze vytvořit pouze na instalaci klienta.

Syntaxe

MQCONN (*QMgrName*, *Hconn*, *CompCode*, *Příčina*)

Parametry

QMgrName

Typ: MQCHAR48 -vstup

Jedná se o název správce front, ke kterému se chce aplikace připojit. Název může obsahovat následující znaky:

- Velká písmena abecedy (A až Z)
- Malá písmena abecedy (a až z)
- Číselné číslice (0 až 9)
- Tečka (.), dopředné lomítko (/), podtržítko (_), procento (%)

Název nesmí obsahovat úvodní ani vložené mezery, ale může obsahovat koncové mezery. Znak null lze použít k označení konce důležitých dat v názvu; hodnota null a všechny následující znaky jsou považovány za mezery. V označených prostředích platí následující omezení:

- V systémech, které používají EBCDIC Katakana, nelze použít malá písmena.
- V systému z/OSnemohou operace a ovládací panely zpracovat názvy, které začínají nebo končí podtržítkem. Z tohoto důvodu se vyvarujte takových jmen.
- V systému IBM iuzavřete názvy obsahující malá písmena, dopředné lomítko nebo procenta do uvozovek, pokud jsou zadány v příkazech. Neuvádějte tyto uvozovky v parametru **QMgrName** .

Pokud se název skládá zcela z mezer, použije se název *výchozího* správce front. Všimněte si však použití prázdných názvů správců front popsaných v sekci aplikací IBM MQ MQI client .

Název zadaný pro *QMgrName* musí být názvem *připojitelného* správce front, nebo pokud jsou použity skupiny správců front, názvem skupiny správců front.

V systému z/OSjsou správci front, ke kterým se lze připojit, určeni prostředím:

- Pro systém CICSmůžete použít pouze správce front, ke kterému je systém CICS připojen. Parametr **QMgrName** musí být stále uveden, ale jeho hodnota je ignorována; prázdné znaky jsou vhodnou volbou.
- V případě systému IMSlze připojit pouze správce front, kteří jsou uvedeni v tabulce definic subsystému (CSQQDEFV), *a* v tabulce SSM v souboru IMS(viz poznámka k použití 6).
- V případě dávkového zpracování z/OS a TSO lze připojit pouze správce front, kteří jsou umístěni ve stejném systému jako aplikace (viz poznámka k použití 6).

Skupiny sdílení front:V systémech, kde existuje několik správců front a jsou konfigurováni tak, aby tvořili skupinu sdílení front, lze název skupiny sdílení front zadat pro produkt *QMgrName* namísto názvu správce front. To umožňuje aplikaci připojit se k *libovolnému* správci front, který je k dispozici ve skupině sdílení front a který je ve stejném obrazu z/OS jako aplikace. Systém lze také nakonfigurovat tak, aby se pomocí prázdného souboru *QMgrName* připojil ke skupině sdílení front místo k výchozímu správci front.

Pokud parametr *QMgrName* určuje název skupiny sdílení front, ale v systému existuje také správce front s tímto názvem, vytvoří se připojení k posledně jmenované skupině přednostně k prvnímu. Připojení k jednomu ze správců front ve skupině sdílení front, o které jste se pokusili, je pouze v případě, že se připojení nezdaří.

Pokud je připojení úspěšné, můžete použít popisovač vrácený voláním MQCONN nebo MQCONNX pro přístup *ke všem* prostředkům (sdíleným i nesdíleným), které patří ke správci front, ke kterému bylo připojení vytvořeno. Přístup k těmto prostředkům podléhá typickým ovládacím prvkům autorizace.

Pokud aplikace zadá dvě volání MQCONN nebo MQCONNX za účelem vytvoření souběžných připojení a jedno nebo obě volání určí název skupiny sdílení front, druhé volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_ALREADY_CONNECTED při připojení ke stejnému správci front jako první volání.

Skupiny sdílení front jsou podporovány pouze v systému z/OS. Připojení ke skupině sdílení front je podporováno pouze v dávkovém prostředí, v dávkovém prostředí RRS, v prostředí CICSa v prostředí TSO. Pro systém CICSmůžete použít pouze skupinu sdílení front, ke které je systém CICS připojen. Stále musíte zadat parametr **QMgrName** , ale jeho hodnota se ignoruje; prázdné znaky jsou vhodnou volbou.

Upozornění: Produkt IMS se nemůže připojit ke skupině sdílení front.

IBM MQ MQI client aplikace: Pro aplikace IBM MQ MQI client se provádí pokus o připojení pro každou definici kanálu připojení klienta s určeným názvem správce front, dokud nebude úspěšná. Správce front však musí mít stejný název jako zadaný název. Je-li zadán prázdný název, bude každý kanál připojení klienta s prázdným názvem správce front zkoušen, dokud nebude úspěšný; v tomto případě nebude provedena žádná kontrola skutečného názvu správce front.

Aplikace klienta IBM MQ nejsou v produktu z/OSpodporovány, ale produkt z/OS může fungovat jako server IBM MQ , ke kterému se mohou připojit aplikace klienta IBM MQ .

IBM MQ MQI client skupiny správců front: Pokud zadaný název začíná hvězdičkou (*), může mít správce front, ke kterému je vytvořeno připojení, jiný název než název určený aplikací. Zadaný název (bez hvězdičky) definuje *skupinu* správců front, kteří jsou vhodní pro připojení. Implementace vybere jednu ze skupiny tak, že postupně vyzkouší každou z nich, dokud není nalezena jedna, ke které lze vytvořit připojení. Pořadí, ve kterém se provádí pokus o připojení, je ovlivněno váhou kanálu klienta a hodnotami afinity připojení kandidátských kanálů. Pokud není žádný ze správců front ve skupině k dispozici pro připojení, volání se nezdaří. Každý správce front je vyzkoušen pouze jednou. Je-li pro název zadána pouze hvězdička, bude použita výchozí skupina správců front definovaná implementací.

Skupiny správců front jsou podporovány pouze pro aplikace spuštěné v prostředí MQ-client; volání se nezdaří, pokud neklientská aplikace určuje název správce front začínající hvězdičkou. Skupina je definována zadáním několika definic kanálů připojení klienta se stejným názvem správce front (zadaný název bez hvězdičky) pro komunikaci s jednotlivými správci front ve skupině. Výchozí skupina je definována zadáním jedné nebo více definic kanálů připojení klienta, každá s prázdným názvem správce front (zadání prázdného názvu má proto stejný účinek jako zadání jediné hvězdičky pro název klientské aplikace).

Po připojení k jednomu správci front ve skupině může aplikace typickým způsobem zadat mezery v polích názvu správce front v deskriptorech zpráv a objektů tak, aby představovaly název správce front, ke kterému se aplikace připojila (*lokální správce front*). Pokud aplikace potřebuje znát tento název, použijte volání MQINQ k dotazu na atribut správce front **QMgrName** .

Předpona hvězdičky k názvu připojení znamená, že aplikace nezávisí na připojení ke konkrétnímu správci front ve skupině. Vhodné aplikace jsou:

- Aplikace, které vkládají zprávy, ale nezískají zprávy.
- Aplikace, které vkládají zprávy požadavků a pak získají zprávy odpovědí z *dočasné dynamické* fronty.

Nevhodné aplikace jsou ty, které potřebují získat zprávy z konkrétní fronty v konkrétním správci front; tyto aplikace nesmí před název uvádět hvězdičku.

Pokud uvedete hvězdičku, maximální délka zbytku názvu je 47 znaků.

Délka tohoto parametru je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

Hconn (připojení)
Typ: MQHCONN-výstup

Tento manipulátor představuje připojení ke správci front. Zadejte jej pro všechna následná volání front zpráv vydaná aplikací. Přestane být platný, když je vydáno volání MQDISC nebo když je ukončena jednotka zpracování, která definuje rozsah popisovače.

Produkt IBM MQ nyní dodává knihovně mqm balíky klienta i balíky serveru. To znamená, že pokud je provedeno volání MQI nalezené v knihovně mqm, zkontroluje se typ připojení, zda se jedná o připojení klienta nebo serveru, a poté se provede správné základní volání. Uživatelská procedura, která je předána *Hconn* , proto může být nyní propojena s knihovnou mqm, ale použita v instalaci klienta.

Rozsah popisovače: Rozsah vráceného popisovače závisí na volání použitém pro připojení ke správci front (MQCONN nebo MQCONNX). Je-li použito volání MQCONNX, závisí obor manipulátoru také na volbě MQCNO_HANDLE_SHARE_ * zadané v poli *Options* struktury MQCNO.

- Pokud je volání MQCONN nebo je zadána volba MQCNO_HANDLE_SHARE_NONE, vrácený popisovač je *nesdílený* popisovač.

Rozsah nesdíleného popisovače je nejmenší jednotka paralelního zpracování podporovaná platformou, na které je aplikace spuštěna (podrobnosti viz Tabulka 546 na stránce 666); popisovač není platný mimo jednotku paralelního zpracování, ze které bylo volání vydáno.

- Zadáte-li volbu MQCNO_HANDLE_SHARE_BLOCK nebo MQCNO_HANDLE_SHARE_NO_BLOCK, vrácený popisovač je *sdílený* popisovač.

  Rozsah sdíleného manipulátoru je proces, který vlastní podproces, ze kterého bylo volání vydáno; manipulátor lze použít z libovolného podprocesu, který patří do tohoto procesu. Ne všechny platformy podporují podprocesy.
- Pokud volání MQCONN nebo MQCONNX selže s kódem dokončení rovným MQCC_FAILED, hodnota Hconn není definována.

Tabulka 546. Rozsah nesdílených manipulátorů na různých platformách

Platforma	**Rozsah nesdíleného popisovače**
z/OS	• CICS: úloha CICS • IMS: úloha až do dalšího synchronizačního bodu (kromě dílčích úloh úlohy). • z/OS dávka a TSO: úloha (kromě dílčích úloh úlohy)
IBM i	Úloha
AIX and Linux	Podproces
32bitové aplikace Windows	Podproces
64bitové aplikace Windows	Podproces

V systému z/OS pro aplikace CICS je vrácená hodnota:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_ALREADY_CONNECTED
(2002, X'7D2') Aplikace je již připojena.

MQRC_CLUSTER_EXIT_LOAD_ERROR
(2267, X'8DB') Nelze načíst uživatelskou proceduru pracovní zátěže klastru.

MQRC_SSL_ALREADY_INICIALIZOVÁNO
(2391, X' 957 ') SSL je již inicializováno.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_CONN_LOAD_ERROR
(2129, X'851 ') Nelze načíst modul připojení adaptéru.

MQRC_ADAPTER_DEFS_ERROR
(2131, X'853 ') Modul definice subsystému adaptéru je neplatný.

MQRC_ADAPTER_DEFS_LOAD_ERROR
(2132, X'854 ') Nelze načíst modul definice subsystému adaptéru.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ADAPTER_STORAGE_SHORTAGE
(2127, X'84F') Nedostatek paměti pro adaptér.

MQRC_ANOTHER_Q_MGR_CONNECTED
(2103, X'837 ') Již je připojen jiný správce front.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_INIT_ERROR
(2375, X' 947 ') Inicializace uživatelské procedury rozhraní API se nezdařila.

MQRC_API_EXIT_TERM_ERROR
(2376, X' 948 ') Ukončení uživatelské procedury rozhraní API se nezdařilo.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONN_ID_IN_USE
(2160, X'870 ') Identifikátor připojení se již používá.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_ERROR
(2273, X'8E1') Chyba při zpracování volání MQCONN.

MQRC_CONNECTION_NOT_AVAILABLE
(2568, X'A08') Vyskytuje se u volání MQCONN nebo MQCONNX, když správce front nemůže poskytnout připojení požadovaného typu připojení v aktuální instalaci. Připojení klienta nelze vytvořit pouze na instalaci serveru. Lokální připojení nelze vytvořit pouze v instalaci klienta.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Uklidení připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_CRYPTO_HARDWARE_ERROR
(2382, X'94E') Chyba konfigurace šifrovacího hardwaru.

MQRC_DUPLICATE_RECOV_COORD
(2163, X'873 ') Koordinátor zotavení existuje.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není v prostředí platné.

Dále ve volání MQCONNX předání řídicího bloku “MQCSP-parametry zabezpečení” na stránce 337 z aplikace CICS nebo IMS .

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOST_NENÍ k dispozici
(2538, X'9EA') Bylo vyvoláno volání MQCONN z klienta pro připojení ke správci front, ale pokus o přidělení konverzace vzdálenému systému se nezdařil.

MQRC_INSTALLATION_MISMATCH
(2583, X'A17') Neshoda mezi instalací správce front a vybranou knihovnou.

MQRC_KEY_REPOSITORY_ERROR
(2381, X'94D') Úložiště klíčů není platné.

MQRC_MAX_CONNS_LIMIT_REACHED
(2025, X'7E9') Byl dosažen maximální počet připojení.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_OPEN_FAILED
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uveden do klidového stavu.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

Chyba MQRC_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

MQRC_SSL_INITIALIZATION_ERROR
(2393, X' 959 ') Chyba inicializace SSL.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Správce front, ke kterému je vytvořeno připojení pomocí volání MQCONN, se nazývá *lokální správce front*.
2. Fronty vlastněné lokálním správcem front se zobrazují aplikaci jako lokální fronty. Je možné vkládat zprávy a získávat zprávy z těchto front.

   Sdílené fronty vlastněné skupinou sdílení front, do které patří lokální správce front, se v aplikaci zobrazují jako lokální fronty. Je možné vkládat zprávy a získávat zprávy z těchto front.

   Fronty vlastněné vzdálenými správci front se zobrazují jako vzdálené fronty. Do těchto front lze vkládat zprávy, nikoli však zprávy z těchto front.
3. Pokud dojde k selhání správce front v době, kdy je spuštěna aplikace, musí aplikace znovu zadat volání MQCONN, aby získala nový manipulátor připojení pro použití v následných voláních IBM MQ . Aplikace může pravidelně provádět volání MQCONN, dokud nebude volání úspěšné.

Pokud si aplikace není jistá, zda je připojena ke správci front, může bezpečně vydat volání MQCONN pro získání manipulátoru připojení. Pokud je aplikace již připojena, vrácený popisovač je stejný jako popisovač vrácený předchozím voláním MQCONN, ale s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_ALREADY_CONNECTED.

4. Po dokončení aplikace používající volání IBM MQ musí aplikace k odpojení od správce front použít volání MQDISC.
5. Pokud volání MQCONN selže s kódem dokončení rovným MQCC_FAILED, hodnota Hconn není definována.
6. V systému z/OS:
   - Dávkové aplikace, aplikace TSO a aplikace IMS musí zadat volání MQCONN, aby používaly jiná volání IBM MQ . Tyto aplikace se mohou souběžně připojovat k více než jednomu správci front.

     Pokud se správce front nezdaří, musí aplikace po restartování správce front znovu spustit volání, aby získala nový manipulátor připojení.

     Ačkoli aplikace IMS mohou opakovaně volat MQCONN, i když jsou již připojeny, nedoporučuje se to pro online programy pro zpracování zpráv (MPP).
   - Aplikace CICS nemusí vydávat volání MQCONN pro použití ostatních volání IBM MQ , ale mohou tak učinit, pokud chtějí. Volání MQCONN i volání MQDISC jsou přijaty. Není však možné se souběžně připojit k více než jednomu správci front.

     Dojde-li k selhání správce front, budou tyto aplikace při restartování správce front automaticky znovu připojeny, a proto není nutné spouštět volání MQCONN.
7. Chcete-li v systému z/OSdefinovat dostupné správce front, postupujte takto:
   - V případě dávkových aplikací mohou systémoví programátoři použít makro CSQBDEF k vytvoření modulu (CSQBDEFV), který definuje výchozí název správce front nebo název skupiny sdílení front.
   - V případě aplikací systému IMS mohou systémoví programátoři použít makro CSQQDEFX k vytvoření modulu (CSQQDEFV), který definuje názvy dostupných správců front a určuje výchozího správce front.

     Kromě toho musí být každý správce front definován pro řídicí oblast IMS a pro každou závislou oblast přistupující k tomuto správci front. Chcete-li to provést, musíte vytvořit člena subsystému v adresáři IMS.Knihovna PROCLIB a identifikujte člena subsystému pro použitelné oblasti IMS . Pokud se aplikace pokusí připojit ke správci front, který není definován ve členu subsystému pro jeho oblast IMS , aplikace se ukončí.

   z/OS Další informace o použití těchto maker naleznete v tématu Makra určená pro použití zákazníkem.
8. V systému IBM inejsou programy, které skončí abnormálně, automaticky odpojeny od správce front. Zapište aplikace, abyste umožnili možnost volání MQCONN nebo MQCONNX vracející kód dokončení MQCC_WARNING a kód příčiny MQRC_ALREADY_CONNECTED. Manipulátor připojení vrácený v této situaci použijte jako normální.

Vyvolání jazyka C

```
MQCONN (QMgrName, &Hconn, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQCHAR48  QMgrName;  /* Name of queue manager */
MQHCONN   Hconn;     /* Connection handle */
MQLONG    CompCode;  /* Completion code */
MQLONG    Reason;    /* Reason code qualifying CompCode */
```

Vyvolání COBOL

```
CALL 'MQCONN' USING QMGRNAME, HCONN, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQCONN (QMgrName, Hconn, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQCONN,(QMGRNAME,HCONN,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
QMGRNAME  DS  CL48  Name of queue manager
HCONN     DS  F     Connection handle
COMPCODE  DS  F     Completion code
REASON    DS  F     Reason code qualifying COMPCODE
```

Vyvolání jazyka Visual Basic

```
MQCONN QMgrName, Hconn, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Správce front Connect (rozšířený)

Volání MQCONNX připojí aplikační program ke správci front. Poskytuje manipulátor připojení správce front, který aplikace používá při následných voláních IBM MQ .

Volání MQCONNX je podobné volání MQCONN s tím rozdílem, že MQCONNX umožňuje zadat volby pro řízení způsobu, jakým volání funguje.

- Toto volání je podporováno na všech systémech IBM MQ a na klientech IBM MQ připojených k těmto systémům.

Připojení klienta nelze vytvořit pouze na instalaci serveru a lokální připojení nelze vytvořit pouze na instalaci klienta.

Syntaxe

MQCONNX (*QMgrName*, *ConnectOpts*, *Hconn*, *CompCode*, *Příčina*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Podrobnosti viz parametr **QMgrName** popsaný v části “MQCONN-Připojení správce front” na stránce 663 .

ConnectOpts
Typ: MQCNO-vstupní/výstupní

Podrobnosti viz “MQCNO-Volby připojení” na stránce 318.

Hconn (připojení)
Typ: MQHCONN-výstup

Tento manipulátor představuje připojení ke správci front. Zadejte jej pro všechna následná volání front zpráv vydaná aplikací. Přestane být platný, když je vydáno volání MQDISC nebo když je ukončena jednotka zpracování, která definuje rozsah popisovače.

Produkt IBM MQ nyní dodává knihovně mqm balíky klienta i balíky serveru. To znamená, že pokud je provedeno volání MQI nalezené v knihovně mqm, zkontroluje se typ připojení, zda se jedná o připojení klienta nebo serveru, a poté se provede správné základní volání. Uživatelská procedura, která je předána *Hconn* , proto může být nyní propojena s knihovnou mqm, ale použita v instalaci klienta.

Rozsah popisovače: Rozsah vráceného popisovače závisí na volání použitém pro připojení ke správci front (MQCONN nebo MQCONNX). Je-li použito volání MQCONNX, závisí obor manipulátoru také na volbě MQCNO_HANDLE_SHARE_ * zadané v poli *Options* struktury MQCNO.

- Pokud je volání MQCONN nebo je zadána volba MQCNO_HANDLE_SHARE_NONE, vrácený popisovač je *nesdílený* popisovač.

  Rozsah nesdíleného popisovače je nejmenší jednotka paralelního zpracování podporovaná platformou, na které je aplikace spuštěna (podrobnosti viz Tabulka 547 na stránce 672); popisovač není platný mimo jednotku paralelního zpracování, ze které bylo volání vydáno.
- Zadáte-li volbu MQCNO_HANDLE_SHARE_BLOCK nebo MQCNO_HANDLE_SHARE_NO_BLOCK, vrácený popisovač je *sdílený* popisovač.

  Rozsah sdíleného manipulátoru je proces, který vlastní podproces, ze kterého bylo volání vydáno; manipulátor lze použít z libovolného podprocesu, který patří do tohoto procesu. Ne všechny platformy podporují podprocesy.
- Pokud volání MQCONN nebo MQCONNX selže s kódem dokončení rovným MQCC_FAILED, hodnota Hconn není definována.

Tabulka 547. Rozsah nesdílených manipulátorů na různých platformách	
Platforma	**Rozsah nesdíleného popisovače**
z/OS	• CICS: úloha CICS • IMS: úloha až do dalšího synchronizačního bodu (kromě dílčích úloh úlohy). • z/OS dávka a TSO: úloha (kromě dílčích úloh úlohy)
IBM i	Úloha
AIX and Linux	Podproces
32bitové aplikace Windows	Podproces
64bitové aplikace Windows	Podproces

V systému z/OS pro aplikace CICS je vrácená hodnota:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

CompCode
Typ: MQLONG-výstup

Podrobnosti viz parametr **CompCode** popsaný v části "MQCONN-Připojení správce front" na stránce 663 .

Příčina
Typ: MQLONG-výstup

Volání MQCONN a MQCONNX mohou vrátit následující kódy. Seznam dalších kódů, které mohou být vráceny voláním MQCONNX, naleznete v následujících kódech.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_ALREADY_CONNECTED
(2002, X'7D2') Aplikace je již připojena.

MQRC_CLUSTER_EXIT_LOAD_ERROR
(2267, X'8DB') Nelze načíst uživatelskou proceduru pracovní zátěže klastru.

MQRC_SSL_ALREADY_INICIALIZOVÁNO
(2391, X' 957 ') SSL je již inicializováno.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_CONN_LOAD_ERROR
(2129, X'851 ') Nelze načíst modul připojení adaptéru.

MQRC_ADAPTER_DEFS_ERROR
(2131, X'853 ') Modul definice subsystému adaptéru je neplatný.

MQRC_ADAPTER_DEFS_LOAD_ERROR
(2132, X'854 ') Nelze načíst modul definice subsystému adaptéru.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ADAPTER_STORAGE_SHORTAGE
(2127, X'84F') Nedostatek paměti pro adaptér.

MQRC_ANOTHER_Q_MGR_CONNECTED
(2103, X'837 ') Již je připojen jiný správce front.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_INIT_ERROR
(2375, X' 947 ') Inicializace uživatelské procedury rozhraní API se nezdařila.

MQRC_API_EXIT_TERM_ERROR
(2376, X' 948 ') Ukončení uživatelské procedury rozhraní API se nezdařilo.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONN_ID_IN_USE
(2160, X'870 ') Identifikátor připojení se již používá.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_ERROR
(2273, X'8E1') Chyba při zpracování volání MQCONN.

MQRC_CONNECTION_NOT_AVAILABLE
(2568, X'A08') Vyskytuje se u volání MQCONN nebo MQCONNX, když správce front nemůže poskytnout připojení požadovaného typu připojení v aktuální instalaci. Připojení klienta nelze vytvořit pouze na instalaci serveru. Lokální připojení nelze vytvořit pouze v instalaci klienta.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Uklidení připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_CRYPTO_HARDWARE_ERROR
(2382, X'94E') Chyba konfigurace šifrovacího hardwaru.

MQRC_DUPLICATE_RECOV_COORD
(2163, X'873 ') Koordinátor zotavení existuje.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není v prostředí platné.

Dále ve volání MQCONNX předání řídicího bloku “MQCSP-parametry zabezpečení” na stránce 337 z aplikace CICS nebo IMS .

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOST_NENÍ k dispozici
(2538, X'9EA') Bylo vyvoláno volání MQCONN z klienta pro připojení ke správci front, ale pokus o přidělení konverzace vzdálenému systému se nezdařil.

MQRC_INSTALLATION_MISMATCH
(2583, X'A17') Neshoda mezi instalací správce front a vybranou knihovnou.

MQRC_KEY_REPOSITORY_ERROR
(2381, X'94D') Úložiště klíčů není platné.

MQRC_MAX_CONNS_LIMIT_REACHED
(2025, X'7E9') Byl dosažen maximální počet připojení.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_OPEN_FAILED
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uveden do klidového stavu.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

Chyba MQRC_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

MQRC_SSL_INITIALIZATION_ERROR
(2393, X' 959 ') Chyba inicializace SSL.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Volání MQCONNX může vrátit následující další kódy příčiny:

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_AIR_CHYBA
(2385, X' 951 ') Záznam ověřovacích informací není platný.

MQRC_AUTH_INFO_CONN_NAME_ERROR
(2387, X' 953 ') Název připojení ověřovacích informací není platný.

MQRC_AUTH_INFO_REC_COUNT_ERROR
(2383, X'94F') Počet záznamů ověřovacích informací není platný.

MQRC_AUTH_INFO_REC_ERROR
(2384, X' 950 ') Pole záznamu ověřovacích informací jsou neplatná.

MQRC_AUTH_INFO_TYPE_ERROR
(2386, X' 952 ') Typ ověřovacích informací není platný.

MQRC_CD_ERROR
(2277, X'8E5') Definice kanálu není platná.

MQRC_CLIENT_CONN_ERROR
(2278, X'8E6') Pole připojení klienta nejsou platná.

MQRC_CNO_ERROR
(2139, X'85B') Struktura voleb připojení je neplatná.

MQRC_CONN_TAG_IN_USE
(2271, X'8DF') Značka připojení je používána.

MQRC_CONN_TAG_NOT_USABLE
(2350, X'92E') Značka připojení není použitelná.

MQRC_CSP_ERROR
(2595, X'A23') Struktura MQCSP je neplatná.

V 9.3.4 **MQRC_FUNCTION_NOT_SUPPORTED**
(2298, X'8FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

MQRC_LDAP_PASSWORD_ERROR
(2390, X' 956 ') Heslo LDAP není platné.

MQRC_LDAP_USER_NAME_ERROR
(2388, X' 954 ') Pole jména uživatele LDAP nejsou platná.

MQRC_LDAP_USER_NAME_LENGTH_ERR
(2389, X' 955 ') Délka jména uživatele LDAP není platná.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_SCO_ERROR
(2380, X'94C') Struktura voleb konfigurace SSL není platná.

MQRC_SSL_CONFIG_ERROR
(2392, X' 958 ') Chyba konfigurace SSL.

MQRC_TOKEN_TIMESTAMP_NOT_VALID
(2064, X'810 ') Token ověření ještě není platný nebo mu vypršela platnost.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

Pro programovací jazyk Visual Basic platí následující bod:

- Parametr **ConnectOpts** je deklarován jako typ MQCNO. Pokud je aplikace spuštěna jako IBM MQ MQI clienta chcete určit parametry kanálu připojení klienta, deklarujte parametr **ConnectOpts** jako typ Any, aby aplikace mohla zadat strukturu MQCNOCD pro volání namísto struktury MQCNO. To však znamená, že parametr **ConnectOpts** nelze zkontrolovat, aby se zajistilo, že se jedná o správný datový typ.

Vyvolání jazyka C

```
MQCONNX (QMgrName, &ConnectOpts, &Hconn, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQCONNX' USING QMGRNAME, CONNECTOPTS, HCONN, COMPCODE,
REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQCONNX (QMgrName, ConnectOpts, Hconn, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQCONNX,(QMGRNAME,CONNECTOPTS,HCONN,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Vyvolání jazyka Visual Basic

```
MQCONNX QMgrName, ConnectOpts, Hconn, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Vytvoření popisovače zprávy

Volání MQCRTMH vrací popisovač zprávy.

Aplikace může použít volání MQCRTMH pro následná volání front zpráv:

- Pomocí volání MQSETMP nastavte vlastnost manipulátoru zprávy.
- Pomocí volání MQINQMP můžete zjistit hodnotu vlastnosti manipulátoru zprávy.
- Volání MQDLTMP slouží k odstranění vlastnosti manipulátoru zprávy.

Manipulátor zprávy lze použít pro volání MQPUT a MQPUT1 k přidružení vlastností manipulátoru zprávy k vlastnostem vkládané zprávy. Podobně zadáním popisovače zprávy ve volání MQGET lze přistupovat k vlastnostem načítané zprávy pomocí popisovače zprávy po dokončení volání MQGET.

K odstranění manipulátoru zprávy použijte příkaz MQDLTMH .

Syntaxe

MQCRTMH (*Hconn*, *CrtMsgHOpts*, *Hmsg*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX. Pokud připojení ke správci front přestane být platné a na manipulátoru zprávy nepracuje žádné volání IBM MQ , bude pro odstranění zprávy implicitně voláno volání MQDLTMH .

Případně můžete zadat následující hodnotu:

MQHC_UNASSOCIATED_HCONN
Manipulátor připojení nepředstavuje připojení k žádnému konkrétnímu správci front.

Při použití této hodnoty musí být manipulátor zprávy odstraněn s explicitním voláním funkce MQDLTMH , aby se uvolnila paměť, která je mu přidělena; IBM MQ manipulátor zprávy nikdy implicitně neodstraní.

Musí existovat alespoň jedno platné připojení ke správci front vytvořenému v podprocesu, který vytváří manipulátor zprávy, jinak volání selže s MQRC_HCONN_ERROR.

V prostředí s více instalacemi na jednom systému je hodnota MQHC_UNASSOCIATED_HCONN omezena na použití s první instalací načtenou do procesu. Kód příčiny MQRC_HMSG_NOT_AVAILABLE je vrácen, pokud je manipulátor zprávy dodán do jiné instalace.

V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*můžete zadat následující hodnotu:

MQHC_DEF_CONN
Výchozí manipulátor připojení

CrtMsgPočet operací HOpts
Typ: MQCMHO-vstup

Volby, které řídí akci MQCRTMH. Podrobnosti viz MQCMHO .

Hmsg
Typ: MQHMSG-výstup

Na výstupu je vrácen manipulátor zprávy, který lze použít k nastavení, dotazování a odstranění vlastností manipulátoru zprávy. Na počátku popisovač zprávy neobsahuje žádné vlastnosti.

Popisovač zprávy má také přidružený deskriptor zprávy. Na začátku obsahuje výchozí hodnoty. Hodnoty přidružených polí deskriptoru zpráv lze nastavit a dotazovat pomocí volání MQSETMP a MQINQMP. Volání MQDLTMP resetuje pole deskriptoru zprávy zpět na výchozí hodnotu.

Je-li parametr *Hconn* zadán jako hodnota MQHC_UNASSOCIATED_HCONN, lze vrácený popisovač zprávy použít pro volání MQGET, MQPUT nebo MQPUT1 s jakýmkoli připojením v rámci jednotky zpracování, ale může být používán pouze jedním voláním IBM MQ v daném okamžiku. Pokud se manipulátor používá, když se druhé volání IBM MQ pokusí použít stejný manipulátor zprávy, druhé volání IBM MQ selže s kódem příčiny MQRC_MSG_HANDLE_IN_USE.

Pokud parametr *Hconn* není MQHC_UNASSOCIATED_HCONN, lze vrácený popisovač zprávy použít pouze pro uvedené připojení.

Stejná hodnota parametru *Hconn* musí být použita pro následná volání MQI, kde je použit tento popisovač zprávy:

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMHBUF
- MQBUFMH

Vrácený popisovač zprávy přestane být platný, když je pro popisovač zprávy vydáno volání MQDLTMH, nebo když je ukončena jednotka zpracování, která definuje rozsah popisovače. Funkce MQDLTMH je volána implicitně, pokud je při vytvoření manipulátoru zprávy zadáno specifické připojení a připojení ke správci front přestane být platné, například pokud je volán modul MQDBC.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CMHO_ERROR
(2461, X'099D') Struktura voleb popisovače zprávy není platná.

MQRC_CONNECTION_BROKEN
(2273, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_HANDLE_NOT_AVAILABLE
(2017, X'07E1') Nejsou k dispozici žádné další popisovače.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HMSG_ERROR
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

C

```
MQCRTMH (Hconn, &CrtMsgHOpts, &Hmsg, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN  Hconn;       /* Connection handle */
MQCMHO   CrtMsgHOpts; /* Options that control the action of MQCRTMH */
MQHMSG   Hmsg;        /* Message handle */
MQLONG   CompCode;    /* Completion code */
MQLONG   Reason;      /* Reason code qualifying CompCode */
```

COBOL

```
CALL 'MQCRTMH' USING HCONN, CRTMSGHOPTS, HMSG, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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);
```

Deklarujte parametry následujícím způsobem:

```
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)
```

Deklarujte parametry následujícím způsobem:

```
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-zpětná volání řízení

Volání MQCTL provádí řídicí akce na zpětných voláních a manipuluje s objekty, které jsou otevřeny pro připojení.

Syntaxe

MQCTL (*Hconn*, *Operace*, *ControlOpts*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*můžete zadat následující speciální hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

Operace
Typ: MQLONG-vstup

Operace, která se zpracovává na zpětném volání definovaném pro uvedený popisovač objektu. Musíte zadat pouze jednu z následujících voleb:

MQOP_START

Spustit spotřebu zpráv pro všechny definované funkce spotřebitele zpráv pro určený manipulátor připojení.

Zpětná volání se spouštějí na podprocesu spuštěném systémem, který se liší od všech podprocesů aplikace.

Tato operace poskytuje řízení poskytovaného manipulátoru připojení systému. Jediná volání MQI, která mohou být vydána jiným podprocesem než podprocesem spotřebitele, jsou:

- MQCTL s operací MQOP_STOP
- MQCTL s operací MQOP_SUSPEND
- MQDISC-Provede MQCTL s operací MQOP_STOP před odpojením připojení HConn.

Funkce MQRC_HCONN_ASYNC_ACTIVE je vrácena, pokud je při spuštění manipulátoru připojení vydáno volání rozhraní API IBM MQ a volání nepochází z funkce spotřebitele zpráv.

Pokud spotřebitel zpráv zastaví připojení během volání MQCBCT_START_CALL, vrátí se volání MQCTL s kódem příčiny selhání MQRC_CONNECTION_STOPPED.

Tuto funkci lze zadat ve funkci spotřebitele. Pro stejné připojení jako rutina zpětného volání je jediným účelem zrušení dříve vydané operace MQOP_STOP.

Tato volba není podporována v následujících prostředích: CICS on z/OS nebo v případě, že je aplikace svázána s knihovnou IBM MQ bez podprocesů.

MQOP_START_WAIT
Spustit spotřebu zpráv pro všechny definované funkce spotřebitele zpráv pro určený manipulátor připojení.

Spotřebitelé zpráv jsou spouštěny ve stejném podprocesu a řízení není vráceno volajícímu modulu MQCTL, dokud:

- Uvolněno pomocí operací MQCTL MQOP_STOP nebo MQOP_SUSPEND, nebo
- Všechny rutiny spotřebitele byly odregistrovány nebo pozastaveny.

Jsou-li všichni spotřebitelé odregistrováni nebo pozastaveni, je vydána implicitní operace MQOP_STOP.

Tuto volbu nelze použít v rámci rutiny zpětného volání ani pro aktuální manipulátor připojení, ani pro žádný jiný manipulátor připojení. Dojde-li k pokusu o volání, vrátí se s hodnotou MQRC_ENVIRONMENT_ERROR.

Pokud kdykoli během operace MQOP_START_WAIT nejsou registrováni žádní nepozastavení spotřebitelé, volání selže s kódem příčiny MQRC_NO_CALLBACKS_ACTIVE.

Pokud je během operace MQOP_START_WAIT připojení pozastaveno, volání MQCTL vrátí varovný kód příčiny MQRC_CONNECTION_SUSPENDOVÁNO; připojení zůstane 'spuštěno'.

Aplikace se může rozhodnout vydat příkaz MQOP_STOP nebo MQOP_RESUME. V této instanci jsou bloky operace MQOP_RESUME.

Tato volba není podporována v jednom klientovi s podporou podprocesů.

MQOP_STOP
Zastavte spotřebu zpráv a počkejte, až všichni spotřebitelé dokončí své operace před dokončením této volby. Tato operace uvolní manipulátor připojení.

Je-li zadána z rutiny zpětného volání, tato volba se projeví až po ukončení rutiny. Po dokončení rutin spotřebitele pro zprávy, které již byly načteny, a po provedení volání zastavení (je-li požadováno) pro rutiny zpětného volání nejsou volány žádné další rutiny spotřebitele zpráv.

Je-li vydán mimo rutinu zpětného volání, řízení se nevrátí volajícímu, dokud nejsou dokončeny rutiny spotřebitele pro již přečtené zprávy a po provedení volání zastavení (je-li požadováno) pro zpětná volání. Samotná zpětná volání však zůstávají registrována.

Tato funkce nemá žádný vliv na zprávy dopředného čtení. Je třeba zajistit, aby spotřebitelé spouštěli příkaz MQCLOSE (MQCO_QUIESCE) z funkce zpětného volání, aby bylo možné určit, zda jsou k dispozici další zprávy, které mají být doručeny.

MQOP_SUSPEND
Pozastavit spotřebu zpráv. Tato operace uvolní manipulátor připojení.

To nemá žádný vliv na čtení před zprávami pro aplikaci. Chcete-li ukončit spotřebu zpráv na dlouhou dobu, zvažte zavření fronty a její opětovné otevření, až bude spotřeba pokračovat.

Je-li vydán z rutiny zpětného volání, neprojeví se, dokud se rutina neukončí. Po ukončení aktuální rutiny nebudou volány žádné další rutiny spotřebitele zpráv.

Je-li vydán mimo zpětné volání, ovládací prvek se nevrátí volajícímu, dokud není dokončena aktuální rutina spotřebitele a nejsou volána žádná další.

MQOP_RESUME
Obnovte spotřebu zpráv.

Tato volba je obvykle vydána z hlavního podprocesu aplikace, ale lze ji také použít z rutiny zpětného volání ke zrušení dřívějšího požadavku na pozastavení vydaného ve stejné rutině.

Pokud je MQOP_RESUME použit k obnovení MQOP_START_WAIT, pak se operace zablokuje.

ControlOpts
Typ: MQCTLO-vstup

Volby, které řídí akci MQCTL

Podrobnosti o struktuře viz MQCTLO .

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855 ') Nelze načíst moduly služeb převodu dat.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALLBACK_LINK_ERROR
(2487, X'9B7') Nelze volat rutinu zpětného volání

MQRC_CALLBACK_NOT_ REGISTROVÁNO
(2448, X' 990 ') Nelze zrušit registraci, pozastavit nebo obnovit, protože neexistuje žádné registrované zpětné volání

MQRC_CALLBACK_ROUTINE_ERROR
(2486, X'9B6') Při volání MQOP_REGISTER byly zadány hodnoty CallbackFunction i CallbackName .

Nebo byla zadána volba CallbackFunction nebo CallbackName , ale neodpovídá aktuálně registrované funkci zpětného volání.

MQRC_CALLBACK_TYPE_ERROR
(2483, X'9B3') Nesprávné pole typu CallBack.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CBD_ERROR
(2444, X'98C') Blok voleb je chybný.

MQRC_CBD_OPTIONS_ERROR
(2484, X'9B4') Chybné pole voleb MQCBD.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Není autorizováno pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Uklidení připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_CORREL_ID_ERROR
(2207, X'89F') Chyba identifikátoru korelace.

MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není v prostředí platné.

MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

MQRC_GET_INHIBITED
(2016, X'7E0') Získá blokováno pro frontu.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální konflikt pracovních jednotek.

MQRC_GMO_ERROR
(2186, X'88A') Struktura voleb Get-message není platná.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Rukojeť používaná pro globální jednotku práce.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOBJ_ERROR
(2019, X'7E3') popisovač objektu není platný.

MQRC_NEKONZISTENTNÍ_PROCHÁZENÍ
(2259, X'8D3') Nekonzistentní specifikace procházení.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

MQRC_INVALID_MSG_UNDER_CURSOR
(2246, X'8C6') Zpráva pod kurzorem není platná pro načtení.

MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s lokální jednotkou práce.

MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Volby shody nejsou platné.

MQRC_MAX_MSG_LENGTH_ERROR
(2485, X'9B5') Chybné pole MaxMsg

MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zpráv není platný.

MQRC_MODULE_ENTRY_NOT_FOUND
(2497, X'9C1') Zadaný vstupní bod funkce nebyl v modulu nalezen.

MQRC_MODULE_INVALID
(2496, X'9C0') Modul byl nalezen, ale je nesprávného typu (32 bit/64 bit) nebo se nejedná o platnou knihovnu DLL.

MQRC_MODULE_NOT_FOUND
(2495, X'9BF') Modul nebyl nalezen ve vyhledávací cestě nebo není autorizován k načtení.

MQRC_MSG_ID_ERROR
(2206, X'89E') Chyba identifikátoru zprávy.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Pořadové číslo zprávy není platné.

MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Použití tokenu zprávy není platné.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') Fronta není otevřena pro procházení.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') Fronta není otevřena pro vstup.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu se od doby otevření změnila.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OPERATION_ERROR
(2488, X'9B8') Chybný kód operace ve volání rozhraní API

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

MQRC_Q_DELETED
(2052, X'804 ') Fronta byla odstraněna.

MQRC_Q_INDEX_TYPE_ERROR
(2394, X'95A') Fronta má chybný typ indexu.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uveden do klidového stavu.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_SIGNAL_NEVYŘÍZENÉ
(2069, X'815 ') Pro tuto rukojeť je signál vynikající.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora synchronizační funkce není k dispozici.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Registrace v globální pracovní jednotce se nezdařila.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotky práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Interval čekání v MQGMO není platný.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Dodána nesprávná verze MQGMO.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Dodána nesprávná verze MQMD.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Rutiny zpětného volání musí zkontrolovat odezvy ze všech služeb, které vyvolávají, a pokud rutina zjistí podmínku, kterou nelze vyřešit, musí zadat příkaz MQCB MQOP_DEREGISTER, aby zabránila opakovaným voláním rutiny zpětného volání.

2. Pokud používáte asynchronní spotřebu v aplikaci, kde správce transakcí XA spravuje globální transakce, včetně aktualizací produktu IBM MQ, musíte zvážit následující další body:
   a. Není platné volání MQCTL (MQOP_START) pro **HConn**po jeho vytvoření po volání **xa_open**.

      Důvodem je, že produkt **HConn** byl připojen ke kontextu XA, a proto k němu nelze přistupovat v samostatném podprocesu nebo v podprocesech používaných mechanizmem asynchronního příjmu.
   b. Pokud v tomto scénáři zavoláte MQCTL (MQOP_START), volání selže s kódem příčiny MQRC_ASYNC_XA_CONFLICT (2350).
   c. Je platné volání MQCTL (MQOP_START_WAIT) pro **HConn**po jeho vytvoření po volání **xa_open**.

      Důvodem je, že tato metoda spuštění mechanizmu asynchronního příjmu způsobí, že všechna další zpětná volání pro **HConn** se spustí na podprocesu, kde je provedeno volání MQCTL. Proto není spojení mezi **HConn** a podprocesem ztraceno.
3. z/OS V systému z/OS, je-li operace MQOP_START:
   - Programy, které používají asynchronní rutiny zpětného volání, musí být oprávněny používat z/OS UNIX System Services (z/OS UNIX).
   - Programy jazykového prostředí (LE), které používají asynchronní rutiny zpětného volání, musí používat běhovou volbu LE POSIX(ON).
   - Programy jiné než LE, které používají asynchronní rutiny zpětného volání, nesmí používat rozhraní z/OS UNIX pthread_create (volatelná služba BPX1PTC).
4. z/OS Funkce MQCTL není v rámci adaptéru IMS podporována.

Poznámka: V systému CICSnení příkaz MQOP_START podporován. Místo toho použijte volání funkce MQOP_START_WAIT.

Vyvolání jazyka C

```
MQCTL (Hconn, Operation, &ControlOpts, &CompCode, &Reason)
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQCTL' USING HCONN, OPERATION, CTLOPTS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQCTL(Hconn, Operation, CtlOpts, CompCode, Reason)
```

Deklarujte parametry následujícím způsobem:

```
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-Odpojit správce front

Volání MQDISC přeruší připojení mezi správcem front a aplikačním programem a jedná se o inverzní volání MQCONN nebo MQCONNX.

- V systému z/OSmusí všechny aplikace, které používají asynchronní spotřebu zpráv, obsluhu událostí nebo zpětné volání, hlavní řídicí podproces před ukončením vydat volání MQDISC. Další podrobnosti viz Asynchronní spotřeba IBM MQ zpráv .
- V systému z/OSnemusí aplikace systému CICS při odpojování od správce front provádět toto volání.

  Pokud aplikace CICS provede toto volání, nemá žádný účinek, pokud nebylo provedeno dřívější volání MQCONNX, s uvedením jedné z následujících možností:

  MQCNO_SERIALIZE_CONN_TAG_Q_MGR
  MQCNO_SERIALIZE_CONN_TAG_QSG
  MQCNO_RESTRICT_CONN_TAG_Q_MGR nebo
  MQCNO_RESTRICT_CONN_TAG_QSG

  volby, v takovém případě jsou všechny aktuálně otevřené popisovače objektu uzavřeny.

Syntaxe

MQDISC (*Hconn*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup/výstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V aplikacích z/OS for CICS můžete vynechat volání MQCONN a zadat následující hodnotu pro *Hconn*:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

Při úspěšném dokončení volání správce front nastaví hodnotu *Hconn* na hodnotu, která není platným popisovačem pro dané prostředí. Tato hodnota je:

MQHC_UNUSABLE_HCONN
Nepoužitelný manipulátor připojení.

V systému z/OSje hodnota *Hconn* nastavena na hodnotu, která není definována.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících kódů:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_BACKED_OUT
(2003, X'7D3') Jednotka práce byla vrácena zpět.

MQRC_CONN_TAG_NOT_RELEASED
(2344, X' 928 ') Značka připojení nebyla uvolněna.

MQRC_OUTCOME_PENDING
(2124, X'84C') Výsledek operace potvrzení je nevyřízený.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_DISC_LOAD_ERROR
(2138, X'85A') Nelze načíst modul odpojení adaptéru.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_INIT_ERROR
(2375, X' 947 ') Inicializace uživatelské procedury rozhraní API se nezdařila.

MQRC_API_EXIT_TERM_ERROR
(2376, X' 948 ') Ukončení uživatelské procedury rozhraní API se nezdařilo.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_OUTCOME_MIXED (smíšený)
(2123, X'84B') Výsledek operace potvrzení nebo vrácení je smíšený.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Pokud je vyvoláno volání MQDISC, když má připojení stále otevřené objekty pod tímto připojením, správce front tyto objekty zavře s volbami zavření nastavenými na MQCO_NONE.
2. Pokud aplikace skončí s nepotvrzenými změnami v transakci, odebrání těchto změn závisí na tom, jak aplikace skončí:
   a. Pokud aplikace před ukončením vydá volání MQDISC:
      - V případě pracovní jednotky koordinované správcem front zadá správce front volání MQCMIT jménem aplikace. Jednotka práce je potvrzena, pokud je to možné, a vrácena zpět, pokud ne.
      - U externě koordinované jednotky práce nedochází ke změně stavu jednotky práce; správce front však obvykle označuje, že jednotka práce musí být potvrzena na žádost koordinátora jednotky práce.

        V systémech z/OS, CICS, IMS (jiné než dávkové programy DL/1) a RRS jsou podobné.

   b. Pokud aplikace skončí normálně, ale bez vyvolání volání MQDISC, přijatá akce závisí na prostředí:
      - V systému z/OSse s výjimkou aplikací MQ Java nebo MQ JMS vyskytují akce popsané v poznámce 2a .
      - Ve všech ostatních případech se vyskytnou akce popsané v poznámce 2c .

      Vzhledem k rozdílům mezi prostředími se ujistěte, že aplikace, které chcete přenést, buď potvrdí, nebo vrátí zpět jednotku práce před jejich ukončením.
   c. Pokud aplikace ukončí *abnormálně* bez volání MQDISC, jednotka práce se odvolá.
3. V systému z/OSplatí následující body:
   - Aplikace CICS nemusí pro odpojení od správce front vyvolat volání MQDISC, protože systém CICS se sám připojí ke správci front a volání MQDISC nemá na toto připojení žádný vliv.
   - CICS, IMS (jiné než dávkové programy DL/1) a aplikace RRS používají pracovní jednotky koordinované externím koordinátorem pracovní jednotky. Výsledkem je, že volání MQDISC neovlivní stav pracovní jednotky (pokud existuje), která existuje při vydání volání.

     Volání MQDISC *však* označuje konec použití značky připojení *ConnTag* , která byla přidružena k připojení dřívějším voláním MQCONNX vydaným aplikací. Pokud existuje aktivní pracovní jednotka, která odkazuje na značku připojení při vydání volání MQDISC, volání se dokončí s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_CONN_TAG_NOT_RELEASED. Značka připojení nebude k dispozici pro opětovné použití, dokud koordinátor externí jednotky práce nevyřeší jednotku práce.

Poznámka: V systému CICSnení příkaz MQOP_START podporován. Místo toho použijte volání funkce MQOP_START_WAIT.

Vyvolání jazyka C

```
MQDISC (&Hconn, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

Vyvolání COBOL

```
CALL 'MQDISC' USING HCONN, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQDISC (Hconn, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
dcl Hconn     fixed bin(31);  /* Connection handle */
dcl CompCode  fixed bin(31);  /* Completion code */
dcl Reason    fixed bin(31);  /* Reason code qualifying CompCode */
```

System/390 vyvolání assembleru

```
CALL MQDISC,(HCONN,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Vyvolání jazyka Visual Basic

```
MQDISC Hconn, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQDLTMH-Odstranění popisovače zprávy

Volání MQDLTMH odstraní manipulátor zprávy a je inverzní k volání MQCRTMH.

Syntaxe

MQDLTMH (*Hconn*, *Hmsg*, *DltMsgHOpts*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **Hmsg** .

Pokud byl manipulátor zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být v podprocesu odstraňovaném manipulátor zprávy ustanoveno platné připojení, jinak volání selže s hodnotou MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-vstup/výstup

Jedná se o popisovač zprávy, který má být odstraněn. Hodnota byla vrácena předchozím voláním MQCRTMH.

Při úspěšném dokončení volání je popisovač nastaven na neplatnou hodnotu pro dané prostředí. Tato hodnota je:

MQHM_UNUSABLE_HMSG
Nepoužitelný popisovač zprávy.

Popisovač zprávy nelze odstranit, pokud probíhá jiné volání IBM MQ , kterému byl předán stejný popisovač zprávy.

DltMsgPočet operací HOpts
Typ: MQDMHO-vstup

Podrobnosti viz MQDMHO .

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

MQRC_DMHO_ERROR
(2462, X'099E') Struktura voleb popisovače zprávy odstranění není platná.

MQRC_HMSG_ERROR
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Obsluha zprávy je již používána.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQDLTMH (Hconn, &Hmsg, &DltMsgHOpts, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQDLTMH' USING HCONN, HMSG, DLTMSGHOPTS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**  Connection handle
01   HCONN   PIC S9(9) BINARY.

**  Options that control the action of MQDLTMH
01   DLTMSGHOPTS.
COPY CMQDMHOL.

**  Completion code
01  COMPCODE   PIC S9(9) BINARY.

** Reason code qualifying COMPCODE
01   REASON      PIC S9(9) BINARY.
```

Vyvolání PL/I

```
call MQDLTMH (Hconn, Hmsg, DltMsgHOpts, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
dcl Hconn          /* Connection handle */
dcl Hmsg           /* Message handle */
```

```
dcl DltMsgHOpts like MQDMHO;    /* Options that control the action of MQDLTMH */
dcl CompCode       /* Completion code */
dcl Reason         /* Reason code qualifying CompCode */
```

Vyvolání High Level Assembler

```
CALL MQDLTMH,(HCONN,HMSG,DLTMSGHOPTS,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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-Odstranit vlastnost zprávy

Volání MQDLTMP odstraní vlastnost z manipulátoru zprávy a je inverzní k volání MQSETMP.

Syntaxe

MQDLTMP (*Hconn*, *Hmsg*, *DltPropOpts*, *Název*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být vytvořeno platné připojení pro podproces odstraňující popisovač zprávy, jinak volání selže s hodnotou MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-vstup

Jedná se o popisovač zprávy obsahující vlastnost, která má být odstraněna. Hodnota byla vrácena předchozím voláním MQCRTMH.

DltPropVolby
Typ: MQDMPO-vstup

Podrobnosti viz datový typ MQDMPO .

Název
Typ: MQCHARV-vstup

Název vlastnosti, která se má odstranit. Další informace o názvech vlastností viz Názvy vlastností .

V názvu vlastnosti nejsou povoleny zástupné znaky.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7') Vlastnost není k dispozici.

MQRC_RFH_FORMAT_ERROR
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nelze analyzovat.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'0852 ') Nelze načíst modul služby adaptéru.

MQRC_ASID_MISMATCH
(2157, X'086D') Primární a domovská ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

MQRC_DMPO_CHYBA
(2481, X'09B1') Struktura voleb vlastností zprávy pro odstranění je neplatná.

MQRC_HMSG_ERROR
(2460, X'099C') popisovač zprávy není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Obsluha zprávy je již používána.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Neplatný název vlastnosti.

MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'0893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQDLTMP (Hconn, Hmsg, &DltPropOpts, &Name, &CompCode, &Reason)
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQDLTMP' USING HCONN, HMSG, DLTPROPOPTS, NAME, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQDLTMP (Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQDLTMP,(HCONN,HMSG,DLTPROPOPTS,NAME,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
HCONN        DS         F     Connection handle
HMSG         DS         D     Message handle
DLTPROPOPTS  CMQDMPOA   ,     Options that control the action of MQDLTMP
NAME         CMQCHRVA   ,     Property name
COMPCODE     DS         F     Completion code
REASON       DS         F     Reason code qualifying COMPCODE
```

MQGET-Získat zprávu

Volání MQGET načte zprávu z lokální fronty, která byla otevřena pomocí volání MQOPEN.

Syntaxe

MQGET (*Hconn*, *Hobj*, *MsgDesc*, *GetMsgOpts*, *BufferLength*, *Buffer*, *DataLength*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

HOBJ
Typ: MQHOBJ-vstup

Tento manipulátor představuje frontu, ze které má být načtena zpráva. Hodnota *Hobj* byla vrácena předchozím voláním MQOPEN. Fronta musí být otevřena s jednou nebo více z následujících voleb (podrobnosti viz "MQOPEN-Otevření objektu" na stránce 734):

- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE
- MQOO_INPUT_AS_Q_DEF
- MQOO_BROWSE

MsgDesc
Typ: MQMD-vstup/výstup

Tato struktura popisuje atributy požadované zprávy a atributy načtené zprávy. Podrobnosti viz "MQMD-Deskriptor zpráv" na stránce 424.

Pokud je hodnota *BufferLength* menší než délka zprávy, správce front vyplní *MsgDesc* , zda je v parametru **GetMsgOpts** uvedena hodnota MQGMO_ACCEPT_TRUNCATED_MSG (viz pole MQGMO-Volby).

Pokud aplikace poskytuje version-1 MQMD, má vrácená zpráva předponu MQMDE pro data zprávy aplikace, ale pouze v případě, že jedno nebo více polí v MQMDE má jinou než výchozí hodnotu. Pokud mají všechna pole v MQMDE výchozí hodnoty, MQMDE se vynechá. Název formátu MQFMT_MD_EXTENSION v poli *Formát* v MQMD označuje přítomnost MQMDE.

Aplikace nemusí poskytovat strukturu MQMD, pokud je v poli *MsgHandle* zadán platný popisovač zprávy. Není-li v tomto poli uvedeno nic, deskriptor zprávy je převzat z deskriptoru přidruženého k popisovači zprávy.

Pokud aplikace poskytuje manipulátor zprávy spíše než strukturu MQMD a určuje MQGMO_PROPERTIES_FORCE_MQRFH2, volání selže s kódem příčiny MQRC_MD_ERROR. Volání také selže s kódem příčiny MQRC_MD_ERROR, pokud aplikace neposkytuje strukturu MQMD a určuje MQGMO_PROPERTIES_AS_Q_DEF a atribut fronty **PropertyControl** je MQPROP_FORCE_MQRFH2.

Pokud jsou zadány volby shody a používá se deskriptor zprávy přidružený k popisovači zprávy, vstupní pole použitá pro porovnání pocházejí z popisovače zprávy.

GetMsgOpty (volby)
Typ: MQGMO-vstup/výstup

Podrobnosti viz "MQGMO-Volby získání zprávy" na stránce 371.

BufferLength
Typ: MQLONG-vstup

Jedná se o délku oblasti *Buffer* v bajtech. Uveďte nulu pro zprávy, které nemají žádná data, nebo pokud má být zpráva odebrána z fronty a data vyřazena (v tomto případě musíte uvést MQGMO_ACCEPT_TRUNCATED_MSG).

Poznámka: Délka nejdelší zprávy, kterou lze číst z fronty, je dána atributem fronty **MaxMsgLength** ; viz "Atributy pro fronty" na stránce 836.

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-výstup

Toto je oblast, která má obsahovat data zprávy. Zarovnejte vyrovnávací paměť na hranici odpovídající povaze dat ve zprávě. 4bajtové zarovnání je vhodné pro většinu zpráv (včetně zpráv obsahujících struktury záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8bajtové zarovnání.

Je-li hodnota *BufferLength* menší než délka zprávy, přesune se co největší část zprávy do adresáře **Buffer**. K tomu dochází v případě, že je v parametru **GetMsgOpts** uvedena hodnota MQGMO_ACCEPT_TRUNCATED_MSG (další informace viz Pole MQGMO-Volby).

Znaková sada a kódování dat v souboru **Buffer** jsou dány poli *CodedCharSetId* a *Encoding* vrácenými v parametru **MsgDesc** . Pokud se tyto hodnoty liší od hodnot požadovaných příjemcem, musí příjemce převést data zprávy aplikace na znakovou sadu a požadované kódování. K převodu dat zprávy lze použít volbu MQGMO_CONVERT (v případě potřeby s uživatelem psanou uživatelskou procedurou). Podrobnosti o této volbě naleznete v části “MQGMO-Volby získání zprávy” na stránce 371 .

Poznámka: Všechny ostatní parametry volání MQGET jsou ve znakové sadě a kódování lokálního správce front (dané atributem správce front **CodedCharSetId** a hodnotou MQENC_NATIVE).

Pokud volání selže, obsah vyrovnávací paměti se mohl ještě změnit.

V programovacím jazyku C je parametr deklarován jako ukazatel-na-void: jako parametr lze zadat adresu libovolného typu dat.

Je-li parametr **BufferLength** nulový, *Buffer* se na něj neodkazuje; v tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo sestavovacím programem System/390 hodnotu null.

DataLength
Typ: MQLONG-výstup

Jedná se o délku dat aplikace *ve zprávě* v bajtech. Je-li tato hodnota větší než *BufferLength*, vrátí se v parametru **Buffer** pouze *BufferLength* bajtů (to znamená, že zpráva je oříznuta). Je-li hodnota nula, zpráva neobsahuje žádná data aplikace.

Pokud je hodnota *BufferLength* menší než délka zprávy, správce front stále dokončí zpracování *DataLength* , zda je v parametru **GetMsgOpts** uvedena hodnota MQGMO_ACCEPT_TRUNCATED_MSG (další informace viz pole MQGMO-Volby). To umožňuje aplikaci určit velikost vyrovnávací paměti požadované pro uložení dat zprávy a poté znovu zadat volání s vyrovnávací pamětí odpovídající velikosti.

Pokud je však zadána volba MQGMO_CONVERT a převedená data zprávy jsou příliš dlouhá na to, aby se vešla do souboru *Buffer*, je hodnota vrácená pro parametr *DataLength* :

- Délka *nepřevedených* dat pro formáty definované správcem front.

  V tomto případě, pokud povaha dat způsobí, že se během převodu rozšíří, musí aplikace přidělit vyrovnávací paměť větší, než je hodnota vrácená správcem front pro *DataLength*.
- Hodnota vrácená uživatelskou procedurou pro převod dat pro formáty definované aplikací.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina

Typ: MQLONG-výstup

Uvedené kódy příčiny jsou ty, které může správce front vrátit pro parametr **Reason** . Pokud aplikace určuje volbu MQGMO_CONVERT a je vyvolána uživatelská procedura pro převod některých nebo všech dat zprávy, rozhoduje uživatelská procedura o tom, jaká hodnota je vrácena pro parametr **Reason** . V důsledku toho jsou možné jiné hodnoty než ty, které jsou zdokumentovány.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_CONVERTED_MSG_TOO_BIG
(2120, X'848 ') Převedená data jsou příliš velká pro vyrovnávací paměť.

MQRC_CONVERTED_STRING_TOO_BIG
(2190, X'88E') Převedený řetězec je pro pole příliš velký.

MQRC_DBCS_ERROR
(2150, X'866 ') Řetězec DBCS není platný.

MQRC_FORMAT_ERROR
(2110, X'83E') Formát zprávy není platný.

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není dokončena.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není dokončena.

MQRC_INCONSISTENT_CCSIDS
(2243, X'8C3') Segmenty zpráv mají různé CCSID.

KÓDOVÁNÍ MQRC_INCONSISTENT_ENCODINGS
(2244, X'8C4') Segmenty zpráv mají odlišné kódování.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Neplatné použití tokenu zprávy.

MQRC_NO_MSG_LOCKED
(2209, X'8A1') Žádná zpráva není uzamčena.

MQRC_NOT_CONVERTED
(2119, X'847 ') Data zprávy nebyla převedena.

MQRC_OPTIONS_CHANGED
(nnnn, X'xxx ') Volby, které měly být konzistentní, byly změněny.

MQRC_PARTIALLY_CONVERTED
(2272, X'8E0') Data zprávy byla částečně převedena.

MQRC_SIGNAL_REQUEST_ACCEPTED
(2070, X'816 ') Nebyla vrácena žádná zpráva (ale byl přijat požadavek na signál).

MQRC_SOURCE_BUFFER_ERROR
(2145, X'861 ') Parametr zdrojové vyrovnávací paměti není platný.

MQRC_SOURCE_CCSID_ERROR
(2111, X'83F') Identifikátor zdrojové kódované znakové sady není platný.

MQRC_SOURCE_DECIMAL_ENC_ERROR
(2113, X'841 ') kódování Packed-decimal ve zprávě nebylo rozpoznáno.

MQRC_SOURCE_FLOAT_ENC_ERROR
(2114, X'842 ') kódování s pohyblivou řádovou čárkou ve zprávě nebylo rozpoznáno.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840 ') kódování zdrojového celého čísla nebylo rozpoznáno.

MQRC_SOURCE_LENGTH_ERROR
(2143, X'85F') Parametr délky zdroje není platný.

MQRC_TARGET_BUFFER_ERROR
(2146, X'862 ') Parametr cílové vyrovnávací paměti není platný.

MQRC_TARGET_CCSID_ERROR
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

MQRC_TARGET_DECIMAL_ENC_ERROR
(2117, X'845 ') kódování Packed-decimal určené příjemcem nebylo rozpoznáno.

MQRC_TARGET_FLOAT_ENC_ERROR
(2118, X'846 ') kódování s pohyblivou řádovou čárkou určené příjemcem nebylo rozpoznáno.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844 ') kódování cílového celého čísla nebylo rozpoznáno.

MQRC_TRUNCATED_MSG_ACCEPTED
(2079, X'81F') Vrácena zkrácená zpráva (zpracování dokončeno).

MQRC_TRUNCATED_MSG_FAILED
(2080, X'820 ') Vrácena zkrácená zpráva (zpracování nebylo dokončeno).

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855 ') Nelze načíst moduly služeb převodu dat.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BACKED_OUT
(2003, X'7D3') Jednotka práce byla vrácena zpět.

MQRC_BUFFER_ERROR
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Zařízení pro spojku není k dispozici.

MQRC_CF_STRUC_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility je používána.

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347, X'92B') Záhlaví seznamu struktury prostředku Coupling Facility je používáno.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Není autorizováno pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Uklidení připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_CORREL_ID_ERROR
(2207, X'89F') Chyba identifikátoru korelace.

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Db2 subsystém není k dispozici.

MQRC_GET_INHIBITED
(2016, X'7E0') Získá blokováno pro frontu.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální konflikt pracovních jednotek.

MQRC_GMO_ERROR
(2186, X'88A') Struktura voleb Get-message není platná.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Rukojeť používaná pro globální jednotku práce.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOBJ_ERROR
(2019, X'7E3') popisovač objektu není platný.

MQRC_NEKONZISTENTNÍ_PROCHÁZENÍ
(2259, X'8D3') Nekonzistentní specifikace procházení.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

MQRC_INVALID_MSG_UNDER_CURSOR
(2246, X'8C6') Zpráva pod kurzorem není platná pro načtení.

MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s lokální jednotkou práce.

MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Volby shody nejsou platné.

MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zpráv není platný.

MQRC_MSG_ID_ERROR
(2206, X'89E') Chyba identifikátoru zprávy.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Pořadové číslo zprávy není platné.

MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Použití tokenu zprávy není platné.

MQRC_NO_MSG_AVAILABLE
(2033, X'7F1') Není k dispozici žádná zpráva.

MQRC_NO_MSG_UNDER_CURSOR
(2034, X'7F2') Kurzor procházení není umístěn na zprávě.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') Fronta není otevřena pro procházení.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') Fronta není otevřena pro vstup.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu se od doby otevření změnila.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

MQRC_Q_DELETED
(2052, X'804 ') Fronta byla odstraněna.

MQRC_Q_INDEX_TYPE_ERROR
(2394, X'95A') Fronta má chybný typ indexu.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uveden do klidového stavu.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_SECOND_MARK_NOT_ALLOWED
(2062, X'80E') Zpráva je již označena.

MQRC_SIGNAL_NEVYŘÍZENÉ
(2069, X'815 ') Pro tuto rukojeť je signál vynikající.

MQRC_SIGNAL1_ERROR
(2099, X'833 ') Pole signálu není platné.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

MQRC_SYNCPOINT_LIMIT_DOSAŽENO
(2024, X'7E8') V rámci aktuální pracovní jednotky nelze zpracovat žádné další zprávy.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') podpora synchronizačního bodu není k dispozici.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Registrace v globální pracovní jednotce se nezdařila.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotky práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Interval čekání v MQGMO není platný.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Dodána nesprávná verze MQGMO.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Dodána nesprávná verze MQMD.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Načtená zpráva je obvykle odstraněna z fronty. K tomuto odstranění může dojít jako součást samotného volání MQGET nebo jako součást synchronizačního bodu.

   Volby procházení jsou: MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXT a MQGMO_BROWSE_MSG_UNDER_CURSOR.

2. Je-li zadána volba MQGMO_LOCK s jednou z voleb procházení, je procházená zpráva uzamknuta tak, aby byla viditelná pouze pro tento manipulátor.

   Je-li zadána volba MQGMO_UNLOCK, dojde k odemknutí dříve zamknuté zprávy. V tomto případě se nenačte žádná zpráva a parametry **MsgDesc**, **BufferLength**, **Buffer**a **DataLength** nejsou kontrolovány ani pozměněny.

3. V případě aplikací, které zadají volání MQGET, může být načtená zpráva ztracena, pokud je aplikace nestandardně ukončena nebo je připojení přerušeno během zpracování volání. K tomuto problému dochází, protože náhradník spuštěný na stejné platformě jako správce front, který zadává volání MQGET jménem aplikace, nemůže zjistit ztrátu aplikace, dokud náhradní zástupce nevrátí zprávu aplikaci po jejím odebrání z fronty. Tento problém se může vyskytnout jak pro trvalé zprávy, tak pro přechodné zprávy.

   Chcete-li eliminovat riziko ztráty zpráv tímto způsobem, vždy načtěte zprávy v rámci pracovních jednotek. To znamená zadáním volby MQGMO_SYNCPOINT ve volání MQGET a použitím volání MQCMIT nebo MQBACK k potvrzení nebo vrácení transakce po dokončení zpracování zprávy. Je-li zadána hodnota MQGMO_SYNCPOINT a klient se ukončí nestandardně nebo je připojení přerušeno, náhradní jednotka práce ve správci front bude odvolána a zpráva bude znovu uvedena do fronty. Další informace o synchronizačních bodech naleznete v tématu Aspekty synchronizačních bodů v IBM MQ aplikacích.

   Tato situace může nastat s klienty IBM MQ i s aplikacemi, které jsou spuštěny na stejné platformě jako správce front.

4. Pokud aplikace vloží posloupnost zpráv na konkrétnífronty v rámci jedné pracovní jednotky a poté úspěšně potvrdí tuto pracovní jednotku, budou zprávy k dispozici pro načtení následujícím způsobem:

   - Pokud se jedná o *nesdílenou frontu* (tj. lokální frontu), budou všechny zprávy v rámci pracovní jednotky k dispozici současně.
   - Pokud se jedná o frontu *sdílenou frontu*, budou zprávy v rámci pracovní jednotky k dispozici v pořadí, v jakém byly vloženy, ale ne všechny najednou. Je-li systém silně zatížen, je možné, že první zpráva v pracovní jednotce bude úspěšně načtena, ale volání MQGET pro druhou nebo následující zprávu v pracovní jednotce selže s MQRC_NO_MSG_AVAILABLE. Dojde-li k tomuto problému, aplikace musí chvíli počkat a poté operaci zopakovat.

5. Pokud aplikace vloží posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, pokud jsou splněny určité podmínky. Podrobnosti viz Poznámky k použití MQPUT . Jsou-li splněny podmínky, jsou zprávy předkládány přijímající žádosti v pořadí, v jakém byly odeslány, pokud:

   - Pouze jeden zásobník získává zprávy z fronty.

     Pokud existují dvě nebo více aplikací, které získávají zprávy z fronty, musí souhlasit s odesílatelem mechanizmu, který má být použit k identifikaci zpráv, které patří do posloupnosti. Například odesilatel může nastavit všechna pole `CorrelId` ve zprávách v posloupnosti na hodnotu, která byla jedinečná pro danou posloupnost zpráv.

- Příjemce záměrně nezmění pořadí načtení, například uvedením konkrétního `MsgId` nebo `CorrelId`.

Pokud odesílající aplikace vloží zprávy jako skupinu zpráv, budou zprávy prezentovány přijímající aplikaci ve správném pořadí, pokud přijímající aplikace ve volání MQGET zadá volbu MQGMO_LOGICAL_ORDER. Další informace o skupinách zpráv viz:

- MQMD-pole MsgFlags
- MQPMO_LOGICAL_ORDER
- MQGMO_LOGICAL_ORDER

Pokud uživatel získává zprávy ve skupině pod synchronizačním bodem, musí před pokusem o dokončení transakce zajistit zpracování celé skupiny.

6. Aplikace musí testovat kód zpětné vazby MQFB_QUIT v poli `Feedback` parametru **MsgDesc** a ukončit, pokud tuto hodnotu naleznou. Další informace viz pole MQMD-Zpětná vazba .
7. Pokud byla fronta určená parametrem `Hobj` otevřena s volbou MQOO_SAVE_ALL_CONTEXT a kód dokončení z volání MQGET je MQCC_OK nebo MQCC_WARNING, je kontext přidružený k popisovači fronty `Hobj` nastaven na kontext zprávy, která byla načtena (pokud není volba MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXT nebo MQGMO_BROWSE_MSG_UNDER_CURSOR označena jako nedostupná).

   Uložený kontext můžete použít pro následné volání MQPUT nebo MQPUT1 zadáním voleb MQPMO_PASS_IDENTITY_CONTEXT nebo MQPMO_PASS_ALL_CONTEXT. To umožňuje, aby byl kontext přijaté zprávy zcela nebo částečně přenesen do jiné zprávy (například když je zpráva postoupena do jiné fronty). Další informace o kontextu zprávy viz Kontext zprávy.
8. Pokud do parametru **GetMsgOpts** zahrnete volbu MQGMO_CONVERT, data zprávy aplikace se před umístěním dat do parametru **Buffer** převedou na reprezentaci požadovanou přijímající aplikací:
   - Pole `Format` v řídicích informacích ve zprávě identifikuje strukturu dat aplikace a pole `CodedCharSetId` a `Encoding` v řídicích informacích ve zprávě uvádí svůj identifikátor znakové sady a kódování.
   - Aplikace vydávající volání MQGET určuje v polích `CodedCharSetId` a `Encoding` v parametru **MsgDesc** identifikátor znakové sady a kódování, na které se mají převést data zprávy aplikace.

   Je-li převod dat zprávy nezbytný, provede převod buď samotný správce front, nebo uživatelská procedura, v závislosti na hodnotě pole `Format` v řídicích informacích zprávy:

   - Následující názvy formátů jsou formáty, které jsou převedeny správcem front; tyto formáty se nazývají "vestavěné" formáty:
     - MQFMT_ADMIN
     - MQFMT_CICS (pouze z/OS)
     - MQFMT_COMMAND_1
     - MQFMT_COMMAND_2
     - MQFMT_DEAD_LETTER_HEADER
     - MQFMT_DIST_HEADER
     - MQFMT_EVENT verze 1
     - MQFMT_EVENT verze 2 (pouze z/OS)
     - MQFMT_IMS
     - MQFMT_IMS_VAR_STRING
     - MQFMT_MD_EXTENSION
     - MQFMT_PCF
     - MQFMT_REF_MSG_HEADER
     - MQFMT_RF_HEADER
     - MQFMT_RF_HEADER_2

- MQFMT_STRING
- MQFMT_TRIGGER
- MQFMT_WORK_INFO_HEADER (pouze z/OS)
- MQFMT_XMIT_Q_HEADER

- Název formátu MQFMT_NONE je speciální hodnota, která označuje, že povaha dat ve zprávě není definována. V důsledku toho se správce front nepokusí o převod, když je zpráva načtena z fronty.

  Poznámka: Je-li ve volání MQGET pro zprávu s názvem formátu MQFMT_NONE zadána hodnota MQGMO_CONVERT a znaková sada nebo kódování zprávy se liší od znakové sady nebo kódování zprávy zadané v parametru **MsgDesc** , bude zpráva vrácena v parametru **Buffer** (bez dalších chyb), ale volání bude dokončeno s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_FORMAT_ERROR.

  MQFMT_NONE můžete použít buď tehdy, když povaha dat zprávy znamená, že nevyžadují převod, nebo když se odesílající a přijímající aplikace mezi sebou dohodly na formuláři, ve kterém mají být odeslána data zprávy.

- Všechny ostatní názvy formátů předávají zprávu uživatelské proceduře pro převod. Uživatelská procedura má stejný název jako formát, kromě přidání specifických pro prostředí. Názvy formátů zadané uživatelem nesmí začínat písmeny IBM MQ.

  Podrobnosti o uživatelské proceduře pro převod dat viz "Uživatelská procedura převodu dat" na stránce 905 .

Uživatelská data ve zprávě lze převést mezi všemi podporovanými znakovými sadami a kódováním. Mějte však na paměti, že pokud zpráva obsahuje jednu nebo více struktur záhlaví IBM MQ , nelze zprávu převést ze znakové sady nebo na znakovou sadu, která obsahuje dvoubajtové nebo vícebajtové znaky pro všechny znaky platné v názvech front. Kód příčiny MQRC_SOURCE_CCSID_ERROR nebo MQRC_TARGET_CCSID_ERROR je výsledkem tohoto pokusu a zpráva je vrácena nepřevedená. Příkladem takové znakové sady je znaková sada Unicode UTF-16 .

Při návratu z příkazu MQGET následující kód příčiny označuje, že zpráva byla úspěšně převedena:

- MQRC_NONE

Následující kód příčiny označuje, že zpráva mohla být úspěšně převedena; aplikace musí zkontrolovat pole `CodedCharSetId` a `Encoding` v parametru **MsgDesc** , aby zjistila:

- MQRC_TRUNCATED_MSG_ACCEPTED

Všechny ostatní kódy příčiny označují, že zpráva nebyla převedena.

Poznámka: Interpretace tohoto kódu příčiny je pravdivá pro převody prováděné uživatelskou procedurou pouze v případě, že uživatelská procedura odpovídá pokynům pro zpracování popsaným v části "Uživatelská procedura převodu dat" na stránce 905.

9. Při použití objektově orientovaného rozhraní k získání zpráv se můžete rozhodnout, že neuvedete vyrovnávací paměť pro uchování dat zprávy pro volání MQGET. Když získáte zprávu pomocí objektově orientované aplikace bez omezení velikosti vyrovnávací paměti pro příjem zpráv, aplikace neselže s MQRC_CONVERTED_MSG_TOO_BIG a přijme převedenou zprávu. To platí pro následující prostředí:
   - .NET, včetně plně spravovaných aplikací
   - C++
   - Java (IBM MQ classes for Java)

   Poznámka: Je-li pro všechny klienty hodnota `sharingConversations` nula a vyrovnávací paměť je příliš malá pro přijetí převedené zprávy, vrátí se nepřevedená zpráva s kódem příčiny MQRC_CONVERTED_MSG_TOO_BIG. Další informace o produktu `sharingConversations`naleznete v tématu Použití sdílení konverzací v klientské aplikaci.
10. Pro vestavěné formáty může správce front provést *výchozí převod* znakových řetězců ve zprávě, je-li zadána volba MQGMO_CONVERT. Výchozí převod umožňuje správci front při převodu řetězcových dat použít výchozí znakovou sadu určenou instalací, která se blíží skutečné znakové sadě. Výsledkem je, že volání MQGET může být úspěšné s kódem dokončení MQCC_OK,

namísto provedení s MQCC_WARNING a kódem příčiny MQRC_SOURCE_CCSID_ERROR nebo MQRC_TARGET_CCSID_ERROR.

Poznámka: Výsledkem použití přibližné znakové sady pro převod řetězcových dat je, že některé znaky mohou být nesprávně převedeny. Chcete-li se tomu vyhnout, použijte znaky v řetězci, které jsou společné jak pro skutečnou znakovou sadu, tak pro výchozí znakovou sadu.

Výchozí převod se vztahuje jak na data zprávy aplikace, tak na znaková pole ve strukturách MQMD a MQMDE:

- Výchozí převod dat zprávy aplikace se vyskytne pouze v případě, že jsou všechny následující příkazy pravdivé:
  - Aplikace určuje parametr MQGMO_CONVERT.
  - Zpráva obsahuje data, která musí být převedena buď ze znakové sady, nebo na znakovou sadu, která není podporována.
  - Při instalaci nebo restartování správce front byl povolen výchozí převod.
- Výchozí převod znakových polí ve strukturách MQMD a MQMDE se provádí podle potřeby, pokud je pro správce front povolen výchozí převod. Převod se provede i v případě, že aplikace ve volání MQGET nezadá volbu MQGMO_CONVERT.

11. Pro programovací jazyk Visual Basic platí následující body:
    - Pokud je velikost parametru **Buffer** menší než délka určená parametrem **BufferLength** , volání selže s kódem příčiny MQRC_STORAGE_NOT_AVAILABLE.
    - Parametr **Buffer** je deklarován jako typ `String`. Pokud data, která mají být načtena z fronty, nejsou typu `String`, použijteVolání MQGETAny namísto volání MQGET.

      Volání MQGETAny má stejné parametry jako volání MQGET s tím rozdílem, že parametr **Buffer** je deklarován jako typ `Any`, což umožňuje načtení libovolného typu dat. To však znamená, že `Buffer` nelze zkontrolovat, aby se zajistilo, že má velikost alespoň `BufferLength` bajtů.
12. Při povolení dopředného čtení nejsou podporovány všechny volby MQGET. Následující tabulka uvádí, které volby jsou povoleny a zda je lze změnit mezi voláními MQGET.

Tabulka 548. Volby MQGET jsou povoleny, když je povoleno dopředné čtení

	Povoleno, když je povoleno dopředné čtení a lze jej změnit mezi voláními MQGET	**Povoleno, když je povoleno dopředné čtení, ale nelze změnit mezi voláními MQGET** [a]	**Volby MQGET, které nejsou povoleny, když je povoleno dopředné čtení** [b]
Hodnoty MQGET MD	MsgId [c] CorrelId [c]	Kódování CodedCharSetId	
Volby 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
Hodnoty MQGMO		MsgHandle	

a. Pokud se tyto volby změní mezi voláními MQGET, vrátí se kód příčiny MQRC_OPTIONS_CHANGED.

b. Pokud se tyto volby zadaly při prvním volání MQGET, bude dopředné čtení zablokováno. Budou-li tyto volby zadány při následném volání MQGET, vrátí se kód příčiny MQRC_OPTIONS_ERROR.

c. Aplikace klienta si musí být vědomy toho, že pokud se hodnoty MsgId a CorrelId změní mezi voláními MQGET, zprávy s předchozími hodnota již mohly být odeslány na klienta a zůstávají ve vyrovnávací paměti klienta pro dopředné čtení, dokud nebudou zpracovány (nebo automaticky vyprázdněny).

d. První volání MQGET určuje, zda se mají zprávy procházet nebo získat z fronty, je-li povoleno dopředné čtení. Pokud se aplikace pokusí použít kombinaci procházení a získání, vrátí se kód příčiny MQRC_OPTIONS_CHANGED.

e. MQGMO_MSG_UNDER_CURSOR nelze použít s dopředným čtením. Zprávy lze procházet nebo získat, je-li dopředné čtení povoleno, ale ne kombinaci obojího.

13. Aplikace mohou destruktivně získávat nepotvrzené zprávy pouze v případě, že jsou tyto zprávy vloženy do stejné lokální pracovní jednotky jako příkaz get. Aplikace nemohou získat nepotvrzené zprávy nedestruktivně.
14. Zprávy pod kurzorem procházení lze načíst v jednotce práce. Tímto způsobem nelze načíst nepotvrzenou zprávu.

Vyvolání jazyka C

```
MQGET (Hconn, Hobj, &MsgDesc, &GetMsgOpts, BufferLength, Buffer,
      &DataLength, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQGET' USING HCONN, HOBJ, MSGDESC, GETMSGOPTS, BUFFERLENGTH,
BUFFER, DATALENGTH, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQGET (Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,
           DataLength, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
         CALL MQGET,(HCONN,HOBJ,MSGDESC,GETMSGOPTS,BUFFERLENGTH,
               BUFFER,DATALENGTH,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Vyvolání jazyka Visual Basic

```
MQGET Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,
DataLength, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Atributy dotazovaného objektu

Volání MQINQ vrátí pole celých čísel a sadu znakových řetězců obsahujících atributy objektu.

Následující typy objektů jsou platné:

- Správce front
- Fronta
- Seznam názvů
- Definice procesu

Syntaxe

MQINQ (*Hconn*, *Hobj*, *SelectorCount*, *Selektory*, *IntAttrCount*, *IntAttrs*, *CharAttrLength*, *CharAttrs*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN -vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním funkce MQCONN nebo MQCONNX .

V systému z/OS pro aplikace CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

HOBJ
Typ: MQHOBJ -vstup

Tento popisovač představuje objekt (libovolného typu) s požadovanými atributy. Popisovač musí být vrácen předchozím voláním MQOPEN , které určilo volbu MQOO_INQUIRE .

SelectorCount
Typ: MQLONG -vstup

Jedná se o počet selektorů, které jsou dodány v poli *Selectors* . Jedná se o počet atributů, které mají být vráceny. Nula je platná hodnota. Maximální povolený počet je 256.

Selektory.
Typ: MQLONG x *SelectorCount* -vstup

Toto je pole selektorů atributů **SelectorCount** ; každý selektor identifikuje atribut (celé číslo nebo znak) s hodnotou, která je povinná.

Každý selektor musí být platný pro typ objektu, který *Hobj* představuje, jinak volání selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_SELECTOR_ERROR.

Ve zvláštním případě front:

- Pokud není selektor platný pro fronty jakéhokoli typu, volání selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_SELECTOR_ERROR.
- Pokud selektor platí pouze pro fronty jiných typů než je typ objektu, volání uspěje s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_SELECTOR_NOT_FOR_TYPE.
- Je-li dotazovaná fronta frontou klastru, závisí platné selektory na tom, jak byla fronta vyřešena; další podrobnosti viz “Poznámky k použití” na stránce 722 .

Selektory můžete zadat v libovolném pořadí. Hodnoty atributů, které odpovídají selektorům celočíselných atributů (MQIA_* selektory), jsou vráceny v produktu *IntAttrs* ve stejném pořadí, v jakém se tyto selektory vyskytují v produktu *Selectors*. Hodnoty atributů, které odpovídají selektorům znakových atributů (MQCA_* selektory), jsou vráceny v produktu *CharAttrs* ve stejném pořadí, ve kterém se tyto selektory vyskytují. Selektory MQIA_* mohou být prokládány se selektory MQCA_* ; důležité je pouze relativní pořadí v rámci jednotlivých typů.

Poznámka:

1. Selektory celočíselných a znakových atributů jsou přiděleny ve dvou různých rozsazích; selektory MQIA_* jsou umístěny v rozsahu MQIA_FIRST až MQIA_LASTa selektory MQCA_* v rozsahu MQCA_FIRST až MQCA_LAST.

   Pro každý rozsah konstanty MQIA_LAST_USED a MQCA_LAST_USED definují nejvyšší hodnotu, kterou správce front přijímá.
2. Pokud se nejprve vyskytnou všechny selektory MQIA_* , lze k adresování odpovídajících prvků v polích *Selectors* a *IntAttrs* použít stejná čísla prvků.

3. Je-li parametr **SelectorCount** nulový, na *Selectors* se neodkazuje. V tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo v sestavovacím modulu S/390 hodnotu null.

Atributy, které lze zjišťovat, jsou uvedeny v následujících tabulkách. Pro selektory MQCA_* je konstanta, která definuje délku výsledného řetězce v *CharAttrs* v bajtech, uvedena v závorkách.

Následující tabulky uvádějí selektory podle objektů v abecedním pořadí:

- Selektory atributů Tabulka 549 na stránce 708 MQINQ pro fronty
- Tabulka 550 na stránce 710 MQINQ selektory atributů pro seznamy názvů
- Tabulka 551 na stránce 711 MQINQ selektory atributů pro definice procesů
- Selektory atributů Tabulka 552 na stránce 711 MQINQ pro správce front

Všechny selektory jsou podporovány na všech platformách IBM MQ , kromě těch, které jsou uvedeny ve sloupci **Poznámka** , jak je uvedeno níže:

NEz/OS
Podporováno na všech platformách **s výjimkou** z/OS

z/OS
Podporováno **pouze** na systému z/OS

Tabulka 549. MQINQ selektory atributů pro fronty

Selektor	Délka pole	Popis	Poznámka
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Datum poslední změny	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Čas poslední změny	
MQCA_BACKOUT_REQ_Q_NAME	MQ_Q_NAME_LENGTH	Nadměrný počet vrácený název fronty	
MQCA_BASE_Q_NAME	MQ_Q_NAME_LENGTH	Název fronty, na kterou se alias interpretuje	
MQCA_CF_STRUC_NAME	MQ_CF_STRUC_NAME_LENGTH	Název struktury prostředku Coupling Facility	z/OS
MQCA_CLUS_CHL_NAME	MQ_CHANNEL_NAME_LENGTH	Název odesílacího kanálu klastru, který používá tuto frontu jako přenosovou frontu.	
MQCA_CLUSTER_NAME	MQ_CLUSTER_NAME_LENGTH	Název klastru	
MQCA_CLUSTER_NAMELIST	MQ_NAMELIST_NAME_LENGTH	Seznam názvů klastru	
MQCA_CREATION_DATE	MQ_CREATION_DATE_LENGTH	Datum vytvoření fronty	
MQCA_CREATION_TIME	MQ_CREATION_TIME_LENGTH	Čas vytvoření fronty	
MQCA_CUSTOM	MQ_CUSTOM_LENGTH	Vlastní atribut pro nové funkce	
MQCA_INITIATION_Q_NAME	MQ_Q_NAME_LENGTH	Název inicializační fronty	
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	Název definice procesu	
MQCA_Q_DESC	MQ_Q_DESC_LENGTH	Popis fronty	
MQCA_Q_NAME	MQ_Q_NAME_LENGTH	Název fronty	

Tabulka 549. MQINQ selektory atributů pro fronty (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQCA_REMOTE_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	Název vzdáleného správce front	
MQCA_REMOTE_Q_NAME	MQ_Q_NAME_LENGTH	Název vzdálené fronty, jak je znám ve vzdáleném správci front	
MQCA_STORAGE_CLASS	MQ_STORAGE_CLASS_LENGTH	Název paměťové třídy	z/OS
MQCA_TRIGGER_DATA	MQ_TRIGGER_DATA_LENGTH	Data spouątěče	
MQCA_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	Jméno přenosové fronty	
MQIA_ACCOUNTING_Q	MQLONG	Řídí shromažďování dat evidence pro frontu.	NEz/OS
MQIA_BACKOUT_THRESHOLD	MQLONG	Práh vrácení	
MQIA_CLWL_Q_PRIORITY	MQLONG	Priorita fronty	
MQIA_CLWL_Q_RANK	MQLONG	Pořadí fronty	
MQIA_CLWL_USEQ	MQLONG	Použít vzdálené fronty	
MQIA_CURRENT_Q_DEPTH	MQLONG	Počet zpráv ve frontě	
MQIA_DEF_BIND	MQLONG	Výchozí vazba	
MQIA_DEF_INPUT_OPEN_OPTION	MQLONG	Výchozí volba otevření pro vstup	
MQIA_DEF_PERSISTENCE	MQLONG	Výchozí trvalost zpráv	
MQIA_DEF_PRIORITY	MQLONG	Výchozí priorita zpráv	
MQIA_DEFINITION_TYPE	MQLONG	Typ definice fronty	
MQIA_DIST_LISTS	MQLONG	Podpora seznamu distribuce	NEz/OS
MQIA_HARDEN_GET_BACKOUT	MQLONG	Zda se má zatvrdit počet vrácení	
MQIA_INDEX_TYPE	MQLONG	Typ indexu udržovaného pro frontu	z/OS
MQIA_INHIBIT_GET	MQLONG	Zda jsou povoleny operace získání	
MQIA_INHIBIT_PUT	MQLONG	Zda jsou povoleny operace vložení	
MQIA_MAX_MSG_LENGTH	MQLONG	Maximální délka zprávy	
MQIA_MAX_Q_DEPTH	MQLONG	Maximální počet zpráv povolených ve frontě	
MQIA_MSG_DELIVERY_SEQUENCE	MQLONG	Zda je priorita zprávy relevantní	
MQIA_NPM_CLASS	MQLONG	Úroveň spolehlivosti pro přechodné zprávy	
MQIA_OPEN_INPUT_COUNT	MQLONG	Počet volání MQOPEN , která mají otevřenou frontu pro vstup	
MQIA_OPEN_OUTPUT_COUNT	MQLONG	Počet volání MQOPEN , která mají otevřenou frontu pro výstup	

Tabulka 549. MQINQ selektory atributů pro fronty (pokračování)			
Selektor	**Délka pole**	**Popis**	**Poznámka**
MQIA_PROPERTY_CONTROL	MQLONG	Atribut řízení vlastností	
MQIA_Q_DEPTH_HIGH_EVENT	MQLONG	Řídicí atribut pro události vysoké hloubky fronty	NEz/OS
MQIA_Q_DEPTH_HIGH_LIMIT	MQLONG	Horní limit pro délku fronty	NEz/OS
MQIA_Q_DEPTH_LOW_EVENT	MQLONG	Atribut řízení pro události nízké hloubky fronty	NEz/OS
MQIA_Q_DEPTH_LOW_LIMIT	MQLONG	Dolní limit pro hloubku fronty	NEz/OS
MQIA_Q_DEPTH_MAX_EVENT	MQLONG	Atribut řízení pro maximální počet událostí délky fronty	NEz/OS
MQIA_Q_SERVICE_INTERVAL	MQLONG	Limit pro interval služby fronty	NEz/OS
MQIA_Q_SERVICE_INTERVAL_EVENT	MQLONG	Řídicí atribut pro události intervalu služby fronty	NEz/OS
MQIA_Q_TYPE	MQLONG	Typ fronty	
MQIA_QSG_DISP	MQLONG	Dispozice skupiny sdílení front	z/OS
MQIA_RETENTION_INTERVAL	MQLONG	Interval uchování fronty	
MQIA_SCOPE	MQLONG	Rozsah definice fronty	NEz/OS
MQIA_SHAREABILITY	MQLONG	Zda lze frontu sdílet pro vstup	
MQIA_STATISTICS_Q	MQLONG	Řídí shromažďování statistických dat pro frontu	NEz/OS
MQIA_TRIGGER_CONTROL	MQLONG	Řízení spouštěče	
MQIA_TRIGGER_DEPTH	MQLONG	Hloubka spouątěče	
MQIA_TRIGGER_MSG_PRIORITY	MQLONG	Prahová hodnota priority zpráv pro spouštěče	
MQIA_TRIGGER_TYPE	MQLONG	Typ spouątěče	
MQIA_USAGE	MQLONG	Pouľití	

Tabulka 550. MQINQ selektory atributů pro seznamy názvů			
Selektor	**Délka pole**	**Popis**	**Poznámka**
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Datum poslední změny	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Čas poslední změny	
MQCA_NAMELIST_DESC	MQ_NAMELIST_DESC_LENGTH	Popis seznamu názvů	

Tabulka 550. MQINQ selektory atributů pro seznamy názvů (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQCA_NAMELIST_NAME	MQ_NAMELIST_NAME_LENGTH	Název objektu seznamu názvů	
MQIA_NAMELIST_TYPE	MQLONG	Typ seznamu názvů	z/OS
MQCA_NAMES	MQ_Q_NAME_LENGTH x *Number of names in the list*	Jména v seznamu názvů	
MQIA_NAME_COUNT	MQLONG	Počet jmen v seznamu názvů	
MQIA_QSG_DISP	MQLONG	Dispozice skupiny sdílení front	z/OS

Tabulka 551. MQINQ selektory atributů pro definice procesů

Selektor	Délka pole	Popis	Poznámka
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Datum poslední změny	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Čas poslední změny	
MQCA_APPL_ID	MQ_PROCESS_APPL_ID_LENGTH	Identifikátor aplikace	
MQCA_ENV_DATA	MQ_PROCESS_ENV_DATA_LENGTH	Data prostředí	
MQCA_PROCESS_DESC	MQ_PROCESS_DESC_LENGTH	Popis definice procesu	
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	Název definice procesu	
MQCA_USER_DATA	MQ_PROCESS_USER_DATA_LENGTH	Data uživatele	
MQIA_APPL_TYPE	MQLONG	Typ aplikace	
MQIA_QSG_DISP	MQLONG	Dispozice skupiny sdílení front	z/OS

Tabulka 552. MQINQ selektory atributů pro správce front

Selektor	Délka pole	Popis	Poznámka
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Datum poslední změny	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Čas poslední změny	
MQCA_CHANNEL_AUTO_DEF_EXIT	MQ_EXIT_NAME_LENGTH	Název uživatelské procedury automatické definice kanálu	
MQCA_CHINIT_SERVICE_PARM		Vyhrazeno pro použití společností IBM	

Tabulka 552. MQINQ selektory atributů pro správce front (pokračování)			
Selektor	**Délka pole**	**Popis**	**Poznámka**
MQCA_CLUSTER_WORKLOAD_DATA	MQ_EXIT_DATA_LENGTH	Data předaná uživatelské proceduře pracovní zátěže klastru	
MQCA_CLUSTER_WORKLOAD_EXIT	MQ_EXIT_NAME_LENGTH	Název uživatelské procedury pracovní zátěže klastru	
MQCA_COMMAND_INPUT_Q_NAME	MQ_Q_NAME_LENGTH	Název vstupní fronty systémového příkazu	
MQCA_CUSTOM	MQ_CUSTOM_LENGTH	Vlastní atribut pro nové funkce	
MQCA_DEAD_LETTER_Q_NAME	MQ_Q_NAME_LENGTH	Název fronty nedoručených zpráv	
MQCA_DEF_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	Výchozí název přenosové fronty	
MQCA_DNS_GROUP	MQ_DNS_GROUP_NAME_LENGTH	Název skupiny pro modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, ke které se má připojit. Tento název se použije při použití služby správy zátěže pro dynamické názvy domén.	z/OS
MQCA_IGQ_USER_ID	MQ_USER_ID_LENGTH	Identifikátor uživatele řazení do fronty v rámci skupiny	z/OS
V 9.3.0 V 9.3.0 MQCA_INITIAL_KEY	MQ_INITIAL_KEY_LENGTH	Počáteční klíč pro systém ochrany hesla	Vrátí hodnotu ********, pokud není prázdná, nebo prázdnou hodnotu, pokud se používá výchozí počáteční klíč.
MQCA_INSTALLATION_DESC	MQ_INSTALLATION_DESC_LENGTH	Popis přidružené instalace	Není z/OS. NEIBM i

Tabulka 552. MQINQ selektory atributů pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQCA_INSTALLATION_NAME	MQ_INSTALLATION_NAME_LENGTH	Název instalace přidružené ke správci front	Není z/OS. NEIBM i
MQCA_INSTALLATION_PATH	MQ_INSTALLATION_PATH_LENGTH	Cesta, kde je nainstalován přidružený soubor IBM MQ	Není z/OS. NEIBM i
MQCA_LU_GROUP_NAME	MQ_LU_NAME_LENGTH	Generické jméno LU pro modul listener 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front, jež má být použita.	z/OS
MQCA_LU_NAME	MQ_LU_NAME_LENGTH	Název LU, která má být použita pro odchozí přenosy LU 6.2 . Nastavte tento název na stejnou logickou jednotku, kterou modul listener používá pro příchozí přenosy.	z/OS
MQCA_LU62_ARM_SUFFIX	MQ_ARM_SUFFIX_LENGTH	Přípona SYS1.PARMLIB člena APPCPM *xx*, který nominuje LUADD pro tento inicializátor kanálu.	z/OS
MQCA_PARENT	MQ_Q_MGR_NAME_LENGTH	Název hierarchicky připojeného správce front, který je nominován jako nadřízený objekt tohoto správce front	
MQCA_Q_MGR_DESC	MQ_Q_MGR_DESC_LENGTH	Popis správce front	
MQCA_Q_MGR_IDENTIFIER	MQ_Q_MGR_IDENTIFIER_LENGT	Identifikátor správce front (H)	
MQCA_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	Název lokálního správce front	
MQCA_QSG_NAME	MQ_QSG_NAME_LENGTH	Název skupiny sdílení front	z/OS
MQCA_REPOSITORY_NAME	MQ_CLUSTER_NAME_LENGTH	Název klastru, pro který správce front poskytuje služby úložiště	
MQCA_REPOSITORY_NAMELIST	MQ_NAMELIST_NAME_LENGTH	Název objektu seznamu názvů obsahujícího názvy klastrů, pro které správce front poskytuje služby úložiště	

Tabulka 552. MQINQ selektory atributů pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
V 9.3.0 V 9.3.0 MQCA_SSL_KEY_REPO_PASSWORD	MQ_SSL_ENCRYP_KEY_REPO_PWD_LEN	Heslo úložiště klíčů	Vrací ********, pokud není prázdné, nebo prázdné, pokud není nastaveno Šifrováno, když je nastaveno před uložením.
MQCA_TCP_NAME	MQ_TCP_NAME_LENGTH	Název systému TCP/IP, který používáte	z/OS
MQIA_ACCOUNTING_CONN_OVERRIDE	MQLONG	Přepsat nastavení evidence	NEz/OS
MQIA_ACCOUNTING_INTERVAL	MQLONG	Jak často zapisovat mezilehlé účetní záznamy	NEz/OS
MQIA_ACCOUNTING_MQI	MQLONG	Řídí shromažďování účetních informací pro data MQI	NEz/OS
MQIA_ACCOUNTING_Q	MQLONG	Řídí shromažďování informací o monitorování účtů pro fronty.	NEz/OS
MQIA_ACTIVE_CHANNELS	MQLONG	Maximální počet kanálů, které mohou být kdykoli aktivní	z/OS
MQIA_ADOPTNEWMCA_CHECK	MQLONG	Prvky, které se kontrolují, aby se určilo, zda se má adoptovat agent MCA. Kontrola se provede, když je zjištěn nový příchozí kanál se stejným názvem jako MCA, který je již aktivní.	z/OS

Tabulka 552. MQINQ selektory atributů pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_ADOPTNEWMCA_INTERVAL	MQLONG	Doba v sekundách, po kterou nový kanál čeká na ukončení osiřelého kanálu	NEz/OS
MQIA_ADOPTNEWMCA_TYPE	MQLONG	Zda se má automaticky restartovat osiřelá instance MCA konkrétního typu kanálu, když je zjištěn nový příchozí požadavek kanálu odpovídající parametrům AdoptNewMCACheck	z/OS
MQIA_AUTHORITY_EVENT	MQLONG	Atribut řízení pro události oprávnění	NEz/OS
MQIA_BRIDGE_EVENT	MQLONG	Řídicí atribut pro události mostu IMS	z/OS
MQIA_CHANNEL_AUTO_DEF	MQLONG	Řídicí atribut pro automatickou definici kanálu	NEz/OS
MQIA_CHANNEL_AUTO_DEF_EVENT	MQLONG	Řídicí atribut pro události automatické definice kanálu	NEz/OS
MQIA_CHANNEL_EVENT	MQLONG	Atribut řízení pro události kanálu	
MQIA_CHINIT_ADAPTERS	MQLONG	Počet dílčích úloh adaptéru, které se mají použít pro zpracování volání IBM MQ	z/OS
MQIA_CHINIT_DISPATCHERS	MQLONG	Počet dispečerů, kteří mají být použiti pro inicializátor kanálu	z/OS
MQIA_CHINIT_TRACE_AUTO_START	MQLONG	Zda spustit trasování inicializátoru kanálu automaticky	z/OS
MQIA_CHINIT_TRACE_TABLE_SIZE	MQLONG	Velikost prostoru trasovacích dat (v MB) inicializátoru kanálu	z/OS
MQIA_CLUSTER_WORKLOAD_LENGTH	MQLONG	Délka pracovní zátěže klastru.	
MQIA_CLWL_MRU_CHANNELS	MQLONG	Počet naposledy použitých kanálů pro vyrovnávání pracovní zátěže klastru	
MQIA_CLWL_USEQ	MQLONG	Použít vzdálené fronty	
MQIA_CODED_CHAR_SET_ID	MQLONG	Identifikátor znakové sady	
MQIA_COMMAND_EVENT	MQLONG	Řídicí atribut pro události příkazu	
MQIA_COMMAND_LEVEL	MQLONG	Úroveň příkazů podporovaná správcem front	
MQIA_CONFIGURATION_EVENT	MQLONG	Atribut řízení pro události konfigurace	NEz/OS
MQIA_DEF_CLUSTER_XMIT_Q_TYPE	MQLONG	Výchozí typ přenosové fronty, kterou budou používat odesílací kanály klastru.	
MQIA_DIST_LISTS	MQLONG	Podpora seznamu distribuce	NEz/OS

Tabulka 552. MQINQ selektory atributů pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_DNS_WLM	MQLONG	Zda se modul listener protokolu TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, registruje ve správci pracovní zátěže pro služby dynamického názvu domény.	z/OS
MQIA_EXPIRY_INTERVAL	MQLONG	Interval mezi skenováním zpráv s vypršenou platností	z/OS
MQIA_GROUP_UR	MQLONG	Určete, zda jsou pro tohoto správce front povoleny jednotky zotavení GROUP. Dispozice pro jednotku zotavení GROUP je k dispozici pouze v případě, že je správce front členem skupiny sdílení front.	z/OS
MQIA_IGQ_PUT_AUTHORITY	MQLONG	Oprávnění k zařazování do fronty v rámci skupiny	z/OS
MQIA_INHIBIT_EVENT	MQLONG	Řídicí atribut pro blokování událostí	NEz/OS
MQIA_INTRA_GROUP_queuing	MQLONG	Podpora řazení do front v rámci skupiny	z/OS
MQIA_LISTENER_TIMER	MQLONG	Časový interval (v sekundách) mezi pokusy produktu IBM MQ restartovat modul listener, pokud došlo k selhání APPC nebo TCP/IP.	z/OS
MQIA_LOCAL_EVENT	MQLONG	Atribut řízení pro lokální události	NEz/OS
MQIA_LOGGER_EVENT	MQLONG	Řídicí atribut pro blokování událostí	NEz/OS
MQIA_LU62_CHANNELS	MQLONG	Maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni pomocí přenosového protokolu LU 6.2	z/OS
MQIA_MSG_MARK_BROWSE_INTERVAL	MQLONG	Časový interval (v milisekundách), po kterém může správce front automaticky odebrat značku z procházení zpráv. **Upozornění:** Tuto hodnotu byste neměli nastavovat pod výchozí hodnotou 5000.	
MQIA_MAX_CHANNELS	MQLONG	Maximální počet kanálů, které mohou být aktuální (včetně kanálů připojení serveru s připojenými klienty)	z/OS
MQIA_MAX_HANDLES	MQLONG	Maximální počet popisovačů	
MQIA_MAX_MSG_LENGTH	MQLONG	Maximální délka zprávy	

Tabulka 552. MQINQ selektory atributů pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_MAX_PRIORITY	MQLONG	Maximální priorita	
MQIA_MAX_UNCOMMITTED_MSGS	MQLONG	Maximální počet nepotvrzených zpráv v rámci pracovní jednotky	
MQIA_OUTBOUND_PORT_MAX	MQLONG	S volbou MQIA_OUTBOUND_PORT_MINdefinuje rozsah čísel portů, která se mají použít při vázání odchozích kanálů.	z/OS
MQIA_OUTBOUND_PORT_MIN	MQLONG	S volbou MQIA_OUTBOUND_PORT_MAXdefinuje rozsah čísel portů, která se mají použít při vázání odchozích kanálů.	z/OS
MQIA_PERFORMANCE_EVENT	MQLONG	Atribut řízení pro události výkonu	NEz/OS
MQIA_PLATFORM	MQLONG	Platforma, na které je umístěn správce front	
MQIA_PROT_POLICY_CAPABILITY	MQLONG	Označuje, zda jsou pro správce front k dispozici funkce zabezpečení produktu Advanced Message Security .	
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	MQLONG	Počet pokusů o opětovné zpracování nezdařené zprávy příkazu pod synchronizačním bodem	
MQIA_PUBSUB_MODE	MQLONG	Zda je spuštěn stroj publikování/odběru a rozhraní publikování/odběru ve frontě. Aplikace, které se mají publikovat nebo odebírat pomocí rozhraní API, vyžadují stroj publikování/odběru. Fronty, které jsou monitorovány rozhraním pro publikování/odběr ve frontě, vyžadují, aby bylo spuštěno rozhraní pro publikování/odběr ve frontě.	
MQIA_PUBSUB_NP_MSG	MQLONG	Zda zrušit (nebo uchovat) nedoručenou vstupní zprávu	
MQIA_PUBSUB_NP_RESP	MQLONG	Ovládá chování nedoručených odpovědí zpráv.	
MQIA_PUBSUB_SYNC_PT	MQLONG	Zda jsou pod synchronizačním bodem zpracovávány pouze trvalé (nebo všechny) zprávy.	
MQIA_QMGR_CFCONLOS	MQLONG	Určuje akci, která má být provedena v případě, že správce front ztratí připojení ke struktuře administrace nebo ke strukturám prostředku CF s volbou CFCONLOS nastavenou na hodnotu ASQMGR .	z/OS

Tabulka 552. MQINQ selektory atributů pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_RECEIVE_TIMEOUT	MQLONG	Přibližně jak dlouho kanál TCP/IP čeká na příjem dat, včetně prezenčních signálů, od svého partnera, než se vrátí do neaktivního stavu. Hodnota je číselná, kvalifikovaná pomocí MQIA_RECEIVE_TIMEOUT_TYPE.	z/OS
MQIA_RECEIVE_TIMEOUT_MIN	MQLONG	Minimální doba, po kterou kanál TCP/IP čeká na přijetí dat, včetně prezenčních signálů, od svého partnera, před návratem do neaktivního stavu	z/OS
MQIA_RECEIVE_TIMEOUT_TYPE	MQLONG	Přibližně jak dlouho kanál TCP/IP čeká na příjem dat, včetně prezenčních signálů, od svého partnera, než se vrátí do neaktivního stavu. MQIA_RECEIVE_TIMEOUT_TYPE je kvalifikátor použitý pro MQIA_RECEIVE_TIMEOUT.	z/OS
MQIA_REMOTE_EVENT	MQLONG	Atribut řízení pro vzdálené události	NEz/OS
MQIA_SECURITY_CASE	MQLONG	Případ bezpečnostních profilů	z/OS
MQIA_SSL_EVENT	MQLONG	Atribut řízení pro události kanálu	
MQIA_SSL_FIPS_REQUIRED	MQLONG	Pro šifrování používat pouze algoritmy s certifikací FIPS	
MQIA_SSL_RESET_COUNT	MQLONG	Počet resetů klíče TLS	
MQIA_START_STOP_EVENT	MQLONG	Řídicí atribut pro události zastavení spuštění	NEz/OS
MQIA_STATISTICS_AUTO_CLUSSDR	MQLONG	Řídí shromažďování statistických monitorovacích informací pro odesílací kanály klastru.	
MQIA_STATISTICS_CHANNEL	MQLONG	Řídí shromažďování statistických dat pro kanály.	
MQIA_STATISTICS_INTERVAL	MQLONG	Jak často zapisovat data monitorování statistiky	NEz/OS
MQIA_STATISTICS_MQI	MQLONG	Řídí shromažďování statistických informací o monitorování pro správce front.	NEz/OS
MQIA_STATISTICS_Q	MQLONG	Řídí shromažďování statistických dat pro fronty	NEz/OS
MQIA_SYNCPOINT	MQLONG	dostupnost synchronizačního bodu	

Tabulka 552. MQINQ selektory atributů pro správce front (pokračování)			
Selektor	**Délka pole**	**Popis**	**Poznámka**
MQIA_TCP_CHANNELS	MQLONG	Maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni pomocí přenosového protokolu TCP/IP	z/OS
MQIA_TCP_KEEP_ALIVE	MQLONG	Zda použít prostředek KEEPALIVE protokolu TCP ke kontrole, zda je druhý konec připojení stále k dispozici	z/OS
MQIA_TCP_STACK_TYPE	MQLONG	Zda může inicializátor kanálu použít pouze adresní prostor TCP/IP uvedený v parametru TCPNAME, nebo zda se může volitelně připojit k libovolné vybrané adrese TCP/IP.	z/OS
MQIA_TRACE_ROUTE_RECORDING	MQLONG	Řídí záznam informací o trasovací trase.	z/OS
MQIA_TREE_LIFE_TIME	MQLONG	Životnost nepoužívaných neadministrativních témat	
MQIA_TRIGGER_INTERVAL	MQLONG	Interval spouštěče	

IntAttrCount
Typ: MQLONG -vstup

Jedná se o počet prvků v poli *IntAttrs* . Nula je platná hodnota.

Je-li IntAttrPočet alespoň počet selektorů MQIA_* v parametru **Selectors** , vrátí se všechny požadované celočíselné atributy.

IntAttrs
Typ: MQLONG x *IntAttrCount* -výstup

Toto je pole celočíselných hodnot atributu *IntAttrCount* .

Hodnoty celočíselných atributů jsou vráceny ve stejném pořadí jako selektory MQIA_* v parametru **Selectors** . Pokud pole obsahuje více prvků, než je počet selektorů MQIA_* , přebytečné prvky se nezmění.

Pokud *Hobj* představuje frontu, ale selektor atributů neplatí pro tento typ fronty, vrátí se specifická hodnota MQIAV_NOT_APPLICABLE . Vrací se pro odpovídající prvek v poli *IntAttrs* .

Pokud je parametr **IntAttrCount** nebo **SelectorCount** nulový, na *IntAttrs* se neodkazuje. V tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo v sestavovacím modulu S/390 hodnotu null.

CharAttrDélka
Typ: MQLONG -vstup

Jedná se o délku parametru **CharAttrs** v bajtech.

CharAttratributu CharAttr musí být alespoň součtem délek požadovaných znakových atributů (viz Selektory). Nula je platná hodnota.

CharAttrs
Typ: MQCHAR x *CharAttrLength* -výstup

Jedná se o vyrovnávací paměť, ve které jsou vráceny atributy znaků, které jsou zřetězeny dohromady. Délka vyrovnávací paměti je dána parametrem **CharAttrLength** .

Atributy znaků jsou vráceny ve stejném pořadí jako selektory MQCA_* v parametru **Selectors** . Délka každého řetězce atributu je pevná pro každý atribut (viz Selektory) a hodnota v něm je v případě potřeby doplněna mezerami. Můžete vytvořit vyrovnávací paměť větší, než je potřeba, aby obsahovala všechny požadované atributy znaků a výplň. Bajty za poslední vrácenou hodnotou atributu se nezměnily.

Pokud *Hobj* představuje frontu, ale selektor atributů se na tento typ fronty nevztahuje, vrátí se znakový řetězec sestávající výhradně z hvězdiček (*). Hvězdička je vrácena jako hodnota tohoto atributu v souboru *CharAttrs*.

Pokud je parametr *CharAttrLength* nebo **SelectorCount** nulový, na *CharAttrs* se neodkazuje. V tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo v sestavovacím modulu S/390 hodnotu null.

CompCode
Typ: MQLONG -výstup

Kód dokončení:

MQCC_OK
Úspěšné dokončení.

MQCC_WARNING
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000') Není důvod k hlášení.

Pokud je *CompCode* MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
(2008, X'7D8') Pro atributy znaků není povolen dostatek prostoru.

MQRC_INT_ATTR_COUNT_TOO_SMALL
(2022, X'7E6') Pro celočíselné atributy není povolen dostatek prostoru.

MQRC_SELECTOR_NOT_FOR_TYPE
(2068, X'814') Selektor nelze použít pro typ fronty.

Pokud je *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X'946') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CF_STRUC_FAILED
(2373, X'945') Struktura prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility se používá.

MQRC_CHAR_ATTR_LENGTH_ERROR
(2006, X'7D6') Délka znakových atributů není platná.

MQRC_CHAR_ATTRS_ERROR
(2007, X'7D7') Řetězec znakových atributů není platný.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Neautorizováno pro připojení.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Probíhá ukončování připojení.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOBJ_ERROR
(2019, X'7E3') popisovač objektu není platný.

MQRC_INT_ATTR_COUNT_ERROR
(2021, X'7E5') Počet celočíselných atributů není platný.

MQRC_INT_ATTRS_ARRAY_ERROR
(2023, X'7E7') Celé pole atributů není platné.

MQRC_NOT_OPEN_FOR_INQUIRE
(2038, X'7F6') Fronta není otevřena pro zjišťování.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu se od otevření změnila.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objekt poškozen.

MQRC_PAGESET_ERROR
(2193, X'891') Chyba při přístupu k datové sadě sady stránek.

MQRC_Q_DELETED
(2052, X'804') Fronta byla odstraněna.

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_STOPPING
(2162, X'872') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Nedostatek dostupných systémových prostředků.

MQRC_SELECTOR_COUNT_ERROR
(2065, X'811') Počet selektorů není platný.

MQRC_SELECTOR_ERROR
(2067, X'813') Selektor atributů není platný.

MQRC_SELECTOR_LIMIT_EXCEEDED
(2066, X'812') Počet selektorů je příliš velký.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') K dispozici je nedostatečná paměť.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno uživatelským programem.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny .

Poznámky k použití

1. Vrácené hodnoty jsou snímkem vybraných atributů. Neexistuje žádná záruka, že atributy zůstanou stejné, než aplikace bude moci jednat na základě vrácených hodnot.
2. Když otevřete modelovou frontu, vytvoří se dynamická lokální fronta. Dynamická lokální fronta se vytvoří i v případě, že otevřete modelovou frontu a dotazujete se na její atributy.

   Atributy dynamické fronty jsou z velké části stejné jako atributy modelové fronty v době vytvoření dynamické fronty. Pokud poté použijete volání MQINQ v této frontě, vrátí správce front atributy dynamické fronty, nikoli atributy modelové fronty. Podrobnosti o atributech modelové fronty, které jsou zděděny dynamickou frontou, naleznete v části Tabulka 561 na stránce 839 .
3. Pokud je dotazovaný objekt alias fronta, hodnoty atributů vrácené voláním MQINQ jsou atributy alias fronty. Nejedná se o atributy základní fronty nebo tématu, na které se alias interpretuje.
4. Pokud je dotazovaný objekt frontou klastru, atributy, které lze dotazovat, závisí na tom, jak je fronta otevřena:

   - Můžete otevřít frontu klastru pro dotazování plus jednu nebo více operací vstupu, procházení nebo nastavení. Chcete-li tak učinit, musí existovat lokální instance fronty klastru, aby bylo otevření úspěšné. V tomto případě jsou atributy, které lze zjišťovat, atributy, které jsou platné pro lokální fronty.

     Pokud je fronta klastru otevřená pro zjišťování bez zadání vstupu, procházení nebo nastavení, volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_SELECTOR_NOT_FOR_TYPE (2068), pokud se pokusíte zjistit atributy, které jsou platné pouze pro lokální fronty, a ne pro fronty klastru.
   - Můžete otevřít frontu klastru pro zjišťování při předávání názvu správce základní fronty připojeného správce front.

     Chcete-li tak učinit, musí existovat lokální instance fronty klastru, aby bylo otevření úspěšné. Není-li správce základní fronty předán, vrátí volání kód dokončení MQCC_WARNING a kód příčiny MQRC_SELECTOR_NOT_FOR_TYPE (2068), pokud se pokusíte zjistit atributy, které jsou platné pouze pro lokální fronty, a nikoli pro fronty klastru.
   - Je-li fronta klastru otevřena pouze pro dotazování nebo pro dotazování a výstup, lze zjistit pouze uvedené atributy. Atribut **QType** má v tomto případě hodnotu MQQT_CLUSTER :
     - MQCA_Q_DESC
     - MQCA_Q_NAME
     - MQIA_DEF_BIND
     - MQIA_DEF_PERSISTENCE
     - MQIA_DEF_PRIORITY
     - MQIA_INHIBIT_PUT
     - MQIA_Q_TYPE

     Frontu klastru můžete otevřít bez pevné vazby. Můžete ji otevřít pomocí parametru MQOO_BIND_NOT_FIXED uvedeného ve volání MQOPEN . Případně uveďte MQOO_BIND_AS_Q_DEFa nastavte atribut **DefBind** fronty na MQBND_BIND_NOT_FIXED. Pokud otevřete frontu klastru bez pevné vazby, následná volání MQINQ pro tuto frontu mohou zjišťovat různé instance fronty klastru. Je však typická pro všechny instance, které mají stejné hodnoty atributů.
   - Objekt alias fronty lze definovat pro klastr. Protože TARGTYPE a TARGET nejsou atributy klastru, proces provádějící proces MQOPEN ve frontě aliasů neví o objektu, na který se alias interpretuje.

Během počátečního MQOPENse alias fronty interpretuje na správce front a frontu v klastru. Rozpoznávání názvů probíhá znovu ve vzdáleném správci front a právě zde je rozpoznána položka TARGTPYE alias fronty.

Pokud se alias fronty interpretuje jako alias tématu, publikování zpráv vložených do alias fronty proběhne v tomto vzdáleném správci front.

Viz Fronty klastru

5. Možná budete chtít zjistit několik atributů a poté některé z nich nastavit pomocí volání MQSET . Chcete-li dotazovat a nastavovat efektivně, umístěte atributy, které mají být nastaveny, na začátek polí selektoru. Pokud tak učiníte, stejná pole se sníženým počtem lze použít pro MQSET.
6. Pokud se vyskytne více než jedna varovná situace (viz parametr **CompCode**), vrácený kód příčiny je první z následujícího seznamu, který platí:
   a. MQRC_SELECTOR_NOT_FOR_TYPE
   b. MQRC_INT_ATTR_COUNT_TOO_SMALL
   c. MQRC_CHAR_ATTRS_TOO_SHORT
7. Následující téma obsahuje informace o atributech objektu:
   - “Atributy pro fronty” na stránce 836
   - “Atributy pro seznamy názvů” na stránce 869
   - “Atributy pro definice procesu” na stránce 871
   - “Atributy pro správce front” na stránce 799

Vyvolání jazyka C

```
MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
     CALL 'MQINQ' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,
                        INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,
                        CHARATTRS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
          IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
         CALL MQINQ,(HCONN,HOBJ,SELECTORCOUNT,SELECTORS,INTATTRCOUNT,  X
               INTATTRS,CHARATTRLENGTH,CHARATTRS,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Vyvolání jazyka Visual Basic

```
MQINQ Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Vlastnost dotazové zprávy

Volání MQINQMP vrací hodnotu vlastnosti zprávy.

Syntaxe

MQINQMP (*Hconn*, *Hmsg*, *InqPropOpts*, *Název*, *PropDesc*, *Typ*, *ValueLength*, *Hodnota*, *DataLength*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **Hmsg** .

Pokud byl manipulátor zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být v podprocesu s dotazem na vlastnost manipulátoru zprávy vytvořeno platné připojení, jinak volání selže s hodnotou MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-vstup

Jedná se o popisovač zprávy, který má být dotazován. Hodnota byla vrácena předchozím voláním funkce **MQCRTMH** .

InqPropOpty
Typ: MQIMPO-vstup/výstup

Podrobnosti viz datový typ MQIMPO .

Název
Typ: MQCHARV-vstup/výstup

Název vlastnosti, která se má dotázat.

Pokud nelze nalézt žádnou vlastnost s tímto názvem, volání se nezdaří s příčinou MQRC_PROPERTY_NOT_AVAILABLE.

Na konci názvu vlastnosti můžete použít zástupný znak procenta (%). Zástupný znak odpovídá žádnému nebo více znakům, včetně znaku tečky (.). To umožňuje aplikaci zjišťovat hodnotu mnoha vlastností. Vyvolejte MQINQMP s volbou MQIMPO_INQ_FIRST, abyste získali první odpovídající vlastnost, a znovu s volbou MQIMPO_INQ_NEXT, abyste získali další odpovídající vlastnost. Nejsou-li k dispozici žádné další odpovídající vlastnosti, volání se nezdaří s hodnotou MQRC_PROPERTY_NOT_AVAILABLE. Je-li pole *ReturnedName* struktury InqPropOpts inicializováno s adresou nebo offsetem pro vrácený název vlastnosti, je tato operace dokončena při návratu z MQINQMP s názvem vlastnosti, pro kterou byla nalezena shoda. Pokud je pole *VSBufSize* položky *ReturnedName* ve struktuře Opts InqPropkratší než délka vráceného názvu vlastnosti, je kód dokončení nastaven na MQCC_FAILED s příčinou MQRC_PROPERTY_NAME_TOO_BIG.

Vlastnosti, které mají známá synonyma, jsou vráceny následujícím způsobem:

1. Vlastnosti s předponou "mqps." jsou vráceny jako název vlastnosti IBM MQ . Například "MQTopicString" je vrácený název spíše než "mqps.Top"
2. Vlastnosti s předponou "jms." nebo "mcd." jsou vráceny jako název pole záhlaví JMS , například "JMSExpiration" je vrácený název spíše než "jms.Exp".

3. Vlastnosti s předponou "usr." jsou vráceny bez této předpony, například "Color" je vrácena spíše než "usr.Color".

Vlastnosti se synonymy jsou vráceny pouze jednou.

V programovacím jazyku C jsou definovány následující proměnné maker pro zjišťování všech vlastností a poté všech vlastností, které začínají na "usr.":

MQPROP_INQUIRE_ALL
Informujte se o všech vlastnostech zprávy.

MQPROP_INQUIRE_ALL lze použít následujícím způsobem:

```
MQCHARV Name = {MQPROP_INQUIRE_ALL};
```

MQPROP_INQUIRE_ALL_USR
Dotázat se na všechny vlastnosti zprávy, která začíná "usr.". Vrácené jméno je vráceno bez "usr." předpona.

Pokud je zadána hodnota MQIMP_INQ_NEXT, ale název se od předchozího volání změnil, nebo se jedná o první volání, je hodnota MQIMPO_INQ_FIRST odvozena.

Další informace o použití názvů vlastností viz Názvy vlastností a Omezení názvů vlastností .

PropDesc
Typ: MQPD-výstup

Tato struktura se používá k definování atributů vlastnosti, včetně toho, co se stane, když vlastnost není podporována, do jakého kontextu zprávy vlastnost patří a do jakých zpráv by měla být vlastnost zkopírována. Podrobnosti o této struktuře viz MQPD .

Typ
Typ: MQLONG-vstup/výstup

Při návratu z volání MQINQMP je tento parametr nastaven na datový typ *Hodnota*. Datový typ může být libovolný z následujících:

MQTYPE_BOOLEAN
Logická hodnota.

MQTYPE_BYTE_STRING
bajtový řetězec.

MQTYPE_INT8
8bitové celé číslo se znaménkem.

MQTYPE_INT16
16bitové celé číslo se znaménkem.

MQTYPE_INT32
32bitové celé číslo se znaménkem.

MQTYPE_INT64
64bitové celé číslo se znaménkem.

MQTYPE_FLOAT32
32bitové číslo s pohyblivou řádovou čárkou.

MQTYPE_FLOAT64
64bitové číslo s pohyblivou řádovou čárkou.

MQTYPE_STRING
Znakový řetězec.

MQTYPE_NULL
Vlastnost existuje, ale má hodnotu null.

Pokud není datový typ hodnoty vlastnosti rozpoznán, vrátí se hodnota MQTYPE_STRING a do oblasti *Hodnota* se umístí řetězcová reprezentace hodnoty. Řetězcovou reprezentaci datového typu

lze nalézt v poli *TypeString* parametru *InqPropOpts* . Je vrácen varovný kód dokončení s příčinou MQRC_PROP_TYPE_NOT_SUPPORTED.

Je-li navíc zadána volba MQIMPO_CONVERT_TYPE, je požadován převod hodnoty vlastnosti. Jako vstup použijte *Typ* , chcete-li určit datový typ, jako který má být vlastnost vrácena. Podrobnosti o převodu datových typů naleznete v popisu volby MQIMPO_CONVERT_TYPE struktury MQIMPO .

Pokud nepožadujete převod typu, můžete na vstupu použít následující hodnotu:

MQTYPE_AS_SET
Hodnota vlastnosti je vrácena bez převodu jejího datového typu.

ValueLength
Typ: MQLONG-vstup

Délka oblasti hodnot v bajtech. Zadejte nulu pro vlastnosti, pro které nepožadujete vrácenou hodnotu. Může se jednat o vlastnosti navržené aplikací tak, aby měly hodnotu null nebo prázdný řetězec. Zadejte také nulu, pokud byla zadána volba MQIMPO_QUERY_LENGTH ; v tomto případě není vrácena žádná hodnota.

Hodnota
Typ: MQBYTEx *ValueLength* -výstup

Jedná se o oblast, která má obsahovat požadovanou hodnotu vlastnosti. Vyrovnávací paměť by měla být zarovnána na hranici odpovídající vracené hodnotě. Pokud tak neučiníte, může dojít k chybě při pozdějším přístupu k hodnotě.

Je-li *ValueLength* menší než délka hodnoty vlastnosti, přesune se co největší část hodnoty vlastnosti do pole *Hodnota* a volání selže s kódem dokončení MQCC_FAILED a příčinou MQRC_PROPERTY_VALUE_TOO_BIG.

Znaková sada dat v poli *Hodnota* je dána polem ReturnedCCSID v parametru InqPropOpts. Kódování dat v poli *Hodnota* je dáno polem ReturnedEncoding v parametru InqPropOpts.

V programovacím jazyku C je parametr deklarován jako ukazatel-na-void; jako parametr lze zadat adresu libovolného typu dat.

Je-li parametr *ValueLength* nulový, *Hodnota* se neodkazuje a její hodnota předaná programy napsané v sestavovacím modulu C nebo System/390 může být null.

DataLength
Typ: MQLONG-výstup

Jedná se o délku (v bajtech) skutečné hodnoty vlastnosti vrácené v oblasti *Hodnota* .

Pokud je *DataLength* menší než délka hodnoty vlastnosti, *DataLength* je stále vyplněn při návratu z volání MQINQMP. To umožňuje aplikaci určit velikost vyrovnávací paměti potřebné k uložení hodnoty vlastnosti a poté znovu zadat volání s vyrovnávací pamětí odpovídající velikosti.

Také mohou být vráceny následující hodnoty.

Je-li parametr *Type* nastaven na MQTYPE_STRING nebo MQTYPE_BYTE_STRING:

MQVL_EMPTY_STRING
Vlastnost existuje, ale neobsahuje žádné znaky ani bajty.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina

Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_PROP_NAME_NOT_CONVERTED
(2492, X'09BC') Vrácený název vlastnosti nebyl převeden.

MQRC_PROP_VALUE_NOT_CONVERTED
(2466, X'09A2') Hodnota vlastnosti nebyla převedena.

MQRC_PROP_TYPE_NOT_SUPPORTED
(2467, X'09A3') Datový typ vlastnosti není podporován.

MQRC_RFH_FORMAT_ERROR
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nelze analyzovat.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'0852 ') Nelze načíst modul služby adaptéru.

MQRC_ASID_MISMATCH
(2157, X'086D') Primární a domovská ASID se liší.

MQRC_BUFFER_ERROR
(2004, X'07D4') Parametr hodnoty není platný.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Parametr délky hodnoty není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

MQRC_DATA_LENGTH_ERROR
(2010, X'07DA') Parametr délky dat není platný.

MQRC_IMPO_CHYBA
(2464, X'09A0') Struktura voleb vlastností dotazové zprávy je neplatná.

MQRC_HMSG_ERROR
(2460, X'099C') popisovač zprávy není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Obsluha zprávy je již používána.

MQRC_OPTIONS_ERROR
(2046, X'07F8') Volby jsou neplatné nebo nekonzistentní.

MQRC_PD_ERROR
(2482, X'09B2') Struktura deskriptoru vlastnosti není platná.

MQRC_PROP_CONV_NOT_SUPPORTED
(2470, X'09A6') Převod ze skutečného na požadovaný datový typ není podporován.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Neplatný název vlastnosti.

MQRC_PROPERTY_NAME_TOO_BIG
(2465, X'09A1') Název vlastnosti je příliš velký pro vrácenou vyrovnávací paměť názvů.

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7) Vlastnost není k dispozici.

MQRC_PROPERTY_VALUE_TOO_BIG
(2469, X'09A5') Hodnota vlastnosti je pro oblast Hodnota příliš velká.

MQRC_PROP_NUMBER_FORMAT_ERROR
(2472, X'09A8') Došlo k chybě formátu čísla v datech hodnoty.

MQRC_PROPERTY_TYPE_ERROR
(2473, X'09A9') Neplatný požadovaný typ vlastnosti.

MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'0871 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'0893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQINQMP (Hconn, Hmsg, &InqPropOpts, &Name, &PropDesc, &Type,
ValueLength, Value, &DataLength, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQINQMP' USING HCONN, HMSG, INQMSGOPTS, NAME, PROPDESC, TYPE,
VALUELENGTH, VALUE, DATALENGTH, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQINQMP (Hconn, Hmsg, InqPropOpts, Name, PropDesc, Type,
ValueLength, Value, DataLength, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQINQMP,(HCONN,HMSG,INQMSGOPTS,NAME,PROPDESC,TYPE,
VALUELENGTH,VALUE,DATALENGTH,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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-Převést popisovač zprávy do vyrovnávací paměti

Volání MQMHBUF převádí manipulátor zprávy na vyrovnávací paměť a jedná se o inverzní volání MQBUFMH.

Syntaxe

MQMHBUF (*Hconn*, *Hmsg*, *MsgHBufOpts*, *Name*, *MsgDesc*, *BufferLength*, *Buffer*, *DataLength*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)

Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **Hmsg** .

Pokud byl manipulátor zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být v podprocesu odstraňovaném manipulátor zprávy zavedeno platné připojení. Není-li navázáno platné připojení, volání se nezdaří s hodnotou MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-vstup

Jedná se o popisovač zprávy, pro který je vyžadována vyrovnávací paměť. Hodnota byla vrácena předchozím voláním MQCRTMH.

MsgHBufOpty (volby)
Typ: MQMHBO-vstup

Struktura MQMHBO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou vyrovnávací paměti vytvářeny z manipulátorů zpráv.

Podrobnosti viz "MQMHBO-Možnosti zpracování zprávy do vyrovnávací paměti" na stránce 480.

Název
Typ: MQCHARV-vstup

Název vlastnosti nebo vlastností, které mají být vloženy do vyrovnávací paměti.

Pokud není nalezena žádná vlastnost odpovídající názvu, volání selže s hodnotou MQRC_PROPERTY_NOT_AVAILABLE.

Chcete-li do vyrovnávací paměti vložit více než jednu vlastnost, můžete použít zástupný znak. K tomu použijte zástupný znak '%' na konci názvu vlastnosti. Tento zástupný znak odpovídá žádnému nebo více znakům, včetně znaku '.' Znak.

V programovacím jazyku C jsou definovány následující proměnné makra pro zjišťování všech vlastností a všech vlastností, které začínají na 'usr':

MQPROP_INQUIRE_ALL
Vložit všechny vlastnosti zprávy do vyrovnávací paměti

MQPROP_INQUIRE_ALL_USR
Vložte všechny vlastnosti zprávy, které začínají znaky 'usr.' do vyrovnávací paměti.

Další informace o použití názvů vlastností viz Názvy vlastností a Omezení názvů vlastností .

MsgDesc
Typ: MQMD-vstup/výstup

Struktura *MsgDesc* popisuje obsah oblasti vyrovnávací paměti.

Ve výstupu jsou pole *Encoding*, *CodedCharSetId* a *Format* nastavena tak, aby správně popisovala kódování, identifikátor znakové sady a formát dat v oblasti vyrovnávací paměti, jak jsou zapsána voláním.

Data v této struktuře jsou ve znakové sadě a kódování aplikace.

BufferLength
Typ: MQLONG-vstup

BufferLength je délka oblasti vyrovnávací paměti v bajtech.

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-výstup

Buffer definuje oblast, která má obsahovat vlastnosti zprávy. Vyrovnávací paměť musíte zarovnat na 4bajtovou hranici.

Pokud je hodnota *BufferLength* menší než délka požadovaná pro uložení vlastností v souboru *Buffer*, MQMHBUF selže s hodnotou MQRC_PROPERTY_VALUE_TOO_BIG.

Obsah vyrovnávací paměti se může změnit i v případě, že volání selže.

DataLength
Typ: MQLONG-výstup

DataLength je délka vrácených vlastností ve vyrovnávací paměti v bajtech. Je-li hodnota nula, žádné vlastnosti neodpovídají hodnotě uvedené v souboru *Name* a volání selže s kódem příčiny MQRC_PROPERTY_NOT_AVAILABLE.

Pokud je hodnota *BufferLength* menší než délka požadovaná pro uložení vlastností do vyrovnávací paměti, volání MQMHBUF selže s MQRC_PROPERTY_VALUE_TOO_BIG, ale hodnota je stále zadána do *DataLength*. To umožňuje aplikaci určit velikost vyrovnávací paměti potřebnou pro přizpůsobení vlastností a poté znovu zadat volání s požadovaným *BufferLength*.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_MHBO_ERROR
(2501, X'095C') Struktura voleb popisovače zprávy do vyrovnávací paměti je neplatná.

MQRC_BUFFER_ERROR
(2004, X'07D4') Parametr vyrovnávací paměti není platný.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONNECTION_BROKEN
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

MQRC_DATA_LENGTH_ERROR
(2010, X'07DA') Parametr délky dat není platný.

MQRC_HMSG_ERROR
(2460, X'099C') popisovač zprávy není platný.

MQRC_MD_ERROR
(2026, X'07EA') Deskriptor zpráv není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Obsluha zprávy je již používána.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Název vlastnosti je neplatný.

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7') Vlastnost není k dispozici.

MQRC_PROPERTY_VALUE_TOO_BIG
(2469, X'09A5') Hodnota BufferLength je příliš malá na to, aby obsahovala zadané vlastnosti.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQMHBUF (Hconn, Hmsg, &MsgHBufOpts, &Name, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Poznámky k použití

MQMHBUF převádí manipulátor zprávy na vyrovnávací paměť.

Můžete jej použít s uživatelskou procedurou rozhraní MQGET API pro přístup k určitým vlastnostem, pomocí rozhraní API vlastností zpráv a pak je předat zpět do vyrovnávací paměti aplikaci, která je navržena tak, aby používala záhlaví MQRFH2 namísto popisovačů zpráv.

Toto volání je inverzní volání MQBUFMH, které lze použít k analýze vlastností zprávy z vyrovnávací paměti do manipulátoru zprávy.

Vyvolání COBOL

```
     CALL 'MQMHBUF' USING HCONN, HMSG, MSGHBUFOPTS, NAME, MSGDESC,
                          BUFFERLENGTH, BUFFER, DATALENGTH, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQMHBUF (Hconn, Hmsg, MsgHBufOpts, Name, MsgDesc, BufferLength, Buffer,
DataLength, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQMHBUF,(HCONN,HMSG,MSGHBUFOPTS,NAME,MSGDESC,BUFFERLENGTH,
               BUFFER,DATALENGTH,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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-Otevření objektu

Volání MQOPEN zavádí přístup k objektu.

Následující typy objektů jsou platné:

- Fronta (včetně distribučních seznamů)
- Seznam názvů
- Definice procesu
- Správce front
- Téma

Syntaxe

MQOPEN (*Hconn*, *ObjDesc*, *Volby*, *Hobj*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota Hconn byla vrácena předchozím voláním MQCONN nebo MQCONNX.

z/OS V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

ObjDesc
Typ: MQOD-vstup/výstup

Toto je struktura, která identifikuje objekt, který se má otevřít; podrobnosti viz "MQOD-Popisovač objektu" na stránce 482 .

Pokud je pole ObjectName v parametru **ObjDesc** název modelové fronty, dynamická lokální fronta. je vytvořen s atributy modelové fronty; to se stane bez ohledu na volby, které zadáte v parametru **Options** . Následné operace používající funkci Hobj vrácenou voláním MQOPEN jsou prováděny v nové dynamické frontě, nikoli v modelové frontě. To platí i pro volání MQINQ a MQSET. Název modelové fronty v parametru **ObjDesc** je nahrazen názvem vytvořené dynamické fronty. Typ dynamické fronty je určen hodnotou atributu **DefinitionType** modelové fronty (viz "Atributy pro fronty" na stránce 836). Informace o volbách zavření použitelných pro dynamické fronty naleznete v popisu volání MQCLOSE.

Volby
Typ: MQLONG-vstup

Musíte uvést alespoň jednu z následujících voleb:

- MQOO_BROWSE
- MQOO_INPUT_ * (pouze jeden z nich)
- MQOO_DOTAZOVAT
- MQOO_OUTPUT
- MQOO_SET
- MQOO_BIND_ * (pouze jeden z nich)

Podrobnosti o těchto volbách naleznete v následující tabulce. Další volby lze zadat podle potřeby. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace). Neplatné kombinace jsou uvedeny; všechny ostatní kombinace jsou platné. Povoleny jsou pouze volby, které lze použít pro typ objektu určený parametrem ObjDesc .

Tabulka 553. Platné volby MQOPEN pro fronty a témata

Volba	Alias [1]	Lokální a model	Vzdálený	Nelokální klastr	Distribuční seznam	Téma
MQOO_INPUT_AS_Q_DEF	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_INPUT_SHARED	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_INPUT_EXCLUSIVE	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_OUTPUT	Ano	Ano	Ano	Ano	Ano	Ano
MQOO_BROWSE	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_CO_OP	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_INQUIRE	Ano	Ano	[2]	Ano	Ne	Ne
MQOO_SET	Ano	Ano	[2]	Ne	Ne	Ne

Tabulka 553. Platné volby MQOPEN pro fronty a témata (pokračování)						
Volba	**Alias [1]**	**Lokální a model**	**Vzdálený**	**Nelokální klastr**	**Distribuční seznam**	**Téma**
MQOO_BIND_ON_OPEN [3]	Ano	Ano	Ano	Ano	Ano	Ne
MQOO_BIND_NOT_FIXED [3]	Ano	Ano	Ano	Ano	Ano	Ne
MQOO_BIND_ON_GROUP [3]	Ano	Ano	Ano	Ano	Ano	Ne
MQOO_BIND_AS_Q_DEF [3]	Ano	Ano	Ano	Ano	Ano	Ne
MQOO_SAVE_ALL_CONTEXT	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_PASS_IDENTITY_CONTEXT	Ano	Ano	Ano	Ano	Ano	[4]
MQOO_PASS_ALL_CONTEXT	Ano	Ano	Ano	Ano	Ano	Ano
MQOO_SET_IDENTITY_CONTEXT	Ano	Ano	Ano	Ano	Ano	[4]
MQOO_SET_ALL_CONTEXT	Ano	Ano	Ano	Ano	Ano	Ano
MQOO_NO_READ_AHEAD	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_READ_AHEAD	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_READ_AHEAD_AS_Q_DEF	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_ALTERNATE_USER_AUTHORITY	Ano	Ano	Ano	Ano	Ano	Ano
MQOO_FAIL_IF_QUIESCING	Ano	Ano	Ano	Ano	Ano	Ano
MQOO_RESOLVE_LOCAL_Q	Ano	Ano	Ano	Ano	Ne	Ne
MQOO_RESOLVE_LOCAL_TOPIC	Ne	Ne	Ne	Ne	Ne	Ano
MQOO_NO_MULTICAST	Ne	Ne	Ne	Ne	Ne	Ano

Notes:

1. Platnost voleb pro aliasy závisí na platnosti volby pro frontu, do které se alias převádí.
2. Tato volba je platná pouze pro lokální definici vzdálené fronty.
3. Tuto volbu lze zadat pro libovolný typ fronty, ale ignoruje se, pokud fronta není frontou klastru. Atribut fronty **DefBind** však přepíše základní frontu, i když alias fronta není v klastru.
4. Tyto atributy lze použít s tématem, ale ovlivňují pouze kontext nastavený pro zachovanou zprávu, nikoli pole kontextu odeslaná libovolnému odběrateli.

Volby přístupu: Následující volby řídí typ operací, které lze na objektu provést:

MQOO_INPUT_AS_Q_DEF
Otevřít frontu pro získání zpráv pomocí výchozího nastavení definovaného frontou.

Fronta je otevřena pro použití s následnými voláními MQGET. Typ přístupu je buď sdílený, nebo výlučný, v závislosti na hodnotě atributu fronty **DefInputOpenOption** ; podrobnosti viz “Atributy pro fronty” na stránce 836 .

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami.

MQOO_INPUT_SHARED
Chcete-li získat zprávy se sdíleným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání může být úspěšné, pokud je fronta aktuálně otevřena touto nebo jinou aplikací s MQOO_INPUT_SHARED, ale nezdaří se s kódem příčiny MQRC_OBJECT_IN_USE, pokud je fronta aktuálně otevřena s MQOO_INPUT_EXCLUSIVE.

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami.

MQOO_INPUT_EXCLUSIVE
Chcete-li získat zprávy s výlučným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání selže s kódem příčiny MQRC_OBJECT_IN_USE, pokud je fronta aktuálně otevřena touto nebo jinou aplikací pro vstup libovolného typu (MQOOO_INPUT_SHARED nebo MQOO_INPUT_EXCLUSIVE).

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami.

MQOO_OUTPUT

Otevřená fronta pro vložení zpráv nebo téma či řetězec tématu pro publikování zpráv.

Fronta nebo téma je otevřeno pro použití s následnými voláními MQPUT.

Volání MQOPEN s touto volbou může být úspěšné i v případě, že je atribut fronty **InhibitPut** nastaven na hodnotu MQQA_PUT_INHIBITED (ačkoli následná volání MQPUT selžou, když je atribut nastaven na tuto hodnotu).

Tato volba je platná pro všechny typy front, včetně distribučních seznamů a témat.

Pro tyto volby platí následující poznámky:

- Lze zadat pouze jednu z těchto voleb.
- Volání MQOPEN s jednou z těchto voleb může být úspěšné i v případě, že je atribut fronty **InhibitGet** nastaven na hodnotu MQQA_GET_INHIBITED (ačkoli následná volání MQGET selžou, když je atribut nastaven na tuto hodnotu).
- Je-li fronta definována jako nesdílená (tj. atribut fronty **Shareability** má hodnotu MQQA_NOT_SHAREABLE), budou pokusy o otevření fronty pro sdílený přístup považovány za pokusy o otevření fronty s výlučným přístupem.
- Pokud je fronta aliasů otevřena s jednou z těchto voleb, test pro výlučné použití (nebo pro to, zda má jiná aplikace výlučné použití) je proti základní frontě, do které se alias převádí.
- Tyto volby nejsou platné, pokud **ObjectQMgrName** je název aliasu správce front. To platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro aliasy správce front je název lokálního správce front.

MQOO_BROWSE
Chcete-li procházet zprávy, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET s jednou z následujících voleb:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_BROWSE_MSG_UNDER_CURSOR

To je povoleno i v případě, že je fronta aktuálně otevřena pro MQOO_INPUT_EXCLUSIVE. Volání MQOPEN s volbou MQOO_BROWSE vytvoří kurzor procházení a umístí jej logicky před první zprávu ve frontě; další informace viz pole MQGMO-Volby .

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami. Není také platný, pokud ObjectQMgrName je název aliasu správce front. To platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro aliasy správce front je název lokálního správce front.

MQOO_CO_OP
Otevřeno jako spolupracující člen sady držadel.

Tato volba je platná pouze s volbou MQOO_BROWSE. Je-li zadána bez příkazu MQOO_BROWSE, MQOPEN vrací hodnotu MQRC_OPTIONS_ERROR.

Vrácený manipulátor je považován za člena spolupracující sady manipulátorů pro následná volání MQGET s jednou z následujících voleb:

- MQGMO_MARK_BROWSE_CO_OP

- MQGMO_UNMARKED_BROWSE_MSG
- MQGMO_UNMARK_BROWSE_CO_OP

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami.

MQOO_DOTAZOVAT
Otevřete objekt pro dotazování atributů.

Fronta, seznam názvů, definice procesu nebo správce front jsou otevřeny pro použití s následnými voláními MQINQ.

Tato volba je platná pro všechny typy objektů kromě rozdělovníků. Je-li `ObjectQMgrName` název aliasu správce front, není platný. To platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro aliasy správce front je název lokálního správce front.

MQOO_SET
Chcete-li nastavit atributy, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQSET.

Tato volba je platná pro všechny typy front kromě rozdělovníků. Není platné, pokud `ObjectQMgrName` je název lokální definice vzdálené fronty; to platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro aliasy správce front je název lokálního správce front.

Volby vazby: Následující volby platí, když je otevíraný objekt frontou klastru; tyto volby řídí vazbu popisovače fronty na instanci fronty klastru:

MQOO_BIND_ON_OPEN
Lokální správce front sváže při otevření fronty manipulátor fronty s instancí cílové fronty. V důsledku toho jsou všechny zprávy vkládané pomocí tohoto popisovače odesílány do stejné instance cílové fronty a stejnou přenosovou cestou.

Tato volba je platná pouze pro fronty a ovlivňuje pouze fronty klastru. Je-li zadána pro frontu, která není frontou klastru, volba se ignoruje.

MQOO_BIND_NOT_FIXED
Tímto se zastaví lokální správce front, který váže manipulátor fronty k instanci cílové fronty. Výsledkem je, že následná volání MQPUT používající tento manipulátor odesílají zprávy do různých instancí cílové fronty nebo do stejné instance, ale různými cestami. Umožňuje také pozdější změnu vybrané instance lokálním správcem front, vzdáleným správcem front nebo agentem kanálu zpráv (MCA) v souladu s podmínkami sítě.

Poznámka: Klientské a serverové aplikace, které potřebují vyměnit řadu zpráv za účelem dokončení transakce, nesmí používat MQOO_BIND_NOT_FIXED (nebo MQOO_BIND_AS_Q_DEF, když má `DefBind` hodnotu MQBND_BIND_NOT_FIXED), protože následné zprávy v řadě mohou být odeslány různým instancím serverové aplikace.

Pokud je pro frontu klastru zadána volba MQOOO_BROWSE nebo jedna z voleb MQOO_INPUT_*, je správce front nucen vybrat lokální instanci fronty klastru. V důsledku toho je vazba manipulátoru fronty pevná, a to i v případě, že je zadána volba MQOO_BIND_NOT_FIXED.

Je-li zadána volba MQOOO_INQUIRE spolu s parametrem MQOO_BIND_NOT_FIXED, mohou následující volání MQINQ, která používají tento manipulátor, zjišťovat různé instance fronty klastru, ačkoli obvykle mají všechny instance stejné hodnoty atributů.

MQOO_BIND_NOT_FIXED je platný pouze pro fronty a ovlivňuje pouze fronty klastru. Je-li zadána pro frontu, která není frontou klastru, volba se ignoruje.

MQOO_BIND_ON_GROUP
Umožňuje aplikaci požadovat, aby byla skupina zpráv přidělena ke stejné cílové instanci.

Tato volba je platná pouze pro fronty a ovlivňuje pouze fronty klastru. Je-li zadána pro frontu, která není frontou klastru, volba se ignoruje.

MQOO_BIND_AS_Q_DEF
Lokální správce front váže manipulátor fronty způsobem definovaným atributem fronty **DefBind** . Hodnota tohoto atributu je buď MQBND_BIND_ON_OPEN, MQBND_BIND_NOT_FIXED, nebo MQBND_BIND_ON_GROUP.

Volba MQOO_BIND_AS_Q_DEF je výchozí, není-li zadána volba MQOO_BIND_ON_OPEN, MQOO_BIND_NOT_FIXED nebo MQOO_BIND_ON_GROUP.

MQOO_BIND_AS_Q_DEF pomáhá s dokumentací programu. Není zamýšleno, aby tato volba byla použita s některou z dalších dvou voleb vazby, ale protože její hodnota je nula, nelze takové použití zjistit.

Volby kontextu: Následující volby řídí zpracování kontextu zprávy:

MQOO_SAVE_ALL_CONTEXT
Informace o kontextu jsou přidruženy k tomuto popisovači fronty. Tato informace je nastavena z kontextu libovolné zprávy načtené pomocí tohoto popisovače. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Tyto kontextové informace lze předat do zprávy, která je poté vložena do fronty pomocí volání MQPUT nebo MQPUT1 . Viz volby MQPMO_PASS_IDENTITY_CONTEXT a MQPMO_PASS_ALL_CONTEXT popsané v části “MQPMO-Volby vložení zprávy” na stránce 502.

Dokud nebude zpráva úspěšně načtena, nelze kontext předat do zprávy vkládané do fronty.

Zpráva načtená pomocí jedné z voleb procházení MQGMO_BROWSE_ * nemá uloženy informace o kontextu (ačkoli jsou pole kontextu v parametru **MsgDesc** nastavena po procházení).

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami. Musí být zadána jedna z voleb MQOO_INPUT_ *.

MQOO_PASS_IDENTITY_CONTEXT
To umožňuje zadání volby MQPMO_PASS_IDENTITY_CONTEXT v parametru **PutMsgOpts** , když je zpráva vložena do fronty; to poskytuje zprávě informace o kontextu identity ze vstupní fronty, která byla otevřena pomocí volby MQOOO_SAVE_ALL_CONTEXT. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Musí být zadána volba MQOO_OUTPUT.

Tato volba je platná pro všechny typy front, včetně rozdělovníků.

MQOO_PASS_ALL_CONTEXT
To umožňuje zadat volbu MQPMO_PASS_ALL_CONTEXT v parametru **PutMsgOpts** , když je zpráva vložena do fronty; to poskytuje zprávě informace o identitě a původu kontextu ze vstupní fronty, která byla otevřena pomocí volby MQOOO_SAVE_ALL_CONTEXT. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Tato volba určuje hodnotu MQOO_PASS_IDENTITY_CONTEXT, kterou proto není nutné zadat. Musí být zadána volba MQOO_OUTPUT.

Tato volba je platná pro všechny typy front, včetně rozdělovníků.

MQOO_SET_IDENTITY_CONTEXT
To umožňuje zadání volby MQPMO_SET_IDENTITY_CONTEXT v parametru **PutMsgOpts** při vložení zprávy do fronty; tato volba poskytuje zprávě informace o kontextu identity obsažené v parametru **MsgDesc** určeném ve volání MQPUT nebo MQPUT1 . Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Tato volba určuje hodnotu MQOO_PASS_IDENTITY_CONTEXT, kterou proto není nutné zadat. Musí být zadána volba MQOO_OUTPUT.

Tato volba je platná pro všechny typy front, včetně rozdělovníků.

MQOO_SET_ALL_CONTEXT
To umožňuje zadání volby MQPMO_SET_ALL_CONTEXT v parametru **PutMsgOpts** při vložení zprávy do fronty; tato volba poskytuje zprávě informace o identitě a původu kontextu obsažené v parametru **MsgDesc** určeném ve volání MQPUT nebo MQPUT1 . Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Tato volba zahrnuje následující volby, které proto nemusí být uvedeny:

- MQOO_PASS_IDENTITY_CONTEXT
- MQOO_PASS_ALL_CONTEXT
- MQOO_SET_IDENTITY_CONTEXT

Musí být zadána volba MQOO_OUTPUT.

Tato volba je platná pro všechny typy front, včetně rozdělovníků.

Volby dopředného čtení:

Při volání MQOPEN s parametrem MQOO_READ_AHEAD povolí klient IBM MQ čtení napřed pouze v případě, že jsou splněny určité podmínky. Tyto podmínky zahrnují:

- Aplikace klienta musí být kompilována a propojena s použitím podprocesových knihoven klienta IBM MQ MQI.
- Kanál klienta musí používat protokol TCP/IP.
- Kanál musí mít nenulové nastavení SharingConversations (SHARECNV) v definici kanálu klienta i serveru.

Následující volby řídí, zda jsou přechodné zprávy odesílány klientovi před tím, než je aplikace požádá. Pro volby dopředného čtení platí následující poznámky:

- Lze zadat pouze jednu z těchto voleb.
- Tyto volby jsou platné pouze pro lokální, alias a modelové fronty. Nejsou platné pro vzdálené fronty, distribuční seznamy, témata nebo správce front.
- Tyto volby lze použít pouze v případě, že jsou určeny také volby MQOO_BROWSE, MQOO_INPUT_SHARED a MQOO_INPUT_EXCLUSIVE, avšak není chybou tyto volby zadat s volbou MQOO_INQUIRE nebo MQOO_SET.
- Pokud aplikace není spuštěna jako klient IBM MQ , jsou tyto volby ignorovány.

MQOO_NO_READ_AHEAD
Přechodné zprávy nejsou klientovi odeslány dříve, než je aplikace požádá.

MQOO_READ_AHEAD
Přechodné zprávy se odesílají klientovi dříve, než je aplikace požádá.

MQOO_READ_AHEAD_AS_Q_DEF
Chování dopředného čtení je určeno výchozím atributem dopředného čtení otevřené fronty. Toto je výchozí hodnota.

Další volby: Následující volby řídí kontrolu autorizace, co se stane, když se správce front uvede do klidového stavu, zda se má interpretovat název lokální fronty a výběrové vysílání:

Oprávnění MQOO_ALTERNATE_USER_AUTHORITY
Pole *AlternateUserId* v parametru **ObjDesc** obsahuje identifikátor uživatele, který se má použít k ověření tohoto volání MQOPEN. Volání může být úspěšné pouze v případě, že je tento produkt *AlternateUserId* autorizován k otevření objektu s uvedenými volbami přístupu, bez ohledu na to, zda je k tomu autorizován identifikátor uživatele, pod kterým je aplikace spuštěna. To se však nevztahuje na žádné zadané kontextové volby, které jsou vždy kontrolovány proti identifikátoru uživatele, pod kterým je aplikace spuštěna.

Tato volba je platná pro všechny typy objektů.

MQOO_FAIL_IF_QUIESCING
Volání MQOPEN se nezdaří, pokud je správce front ve stavu uvedení do klidového stavu.

z/OS V systému z/OSpro aplikaci CICS nebo IMS tato volba také vynutí selhání volání MQOPEN, pokud je připojení ve stavu uvedení do klidového stavu.

Tato volba je platná pro všechny typy objektů.

Informace o kanálech klienta viz IBM MQ MQI clients.

MQOO_RESOLVE_LOCAL_Q
Vyplňte ResolvedQName ve struktuře MQOD názvem lokální fronty, která byla otevřena. Podobně je název ResolvedQMgrvyplněn názvem lokálního správce front, který je hostitelem lokální fronty. Je-li struktura MQOD nižší než verze 3, je hodnota MQOO_RESOLVE_LOCAL_Q ignorována bez vracené chyby.

Lokální fronta je vždy vrácena, když je otevřena buď lokální, alias, nebo modelová fronta, ale nejedná se například o případ, kdy je otevřena vzdálená fronta nebo jiná než lokální fronta klastru bez volby MQOO_RESOLVE_LOCAL_Q; názvy ResolvedQName a ResolvedQMgrjsou vyplněny názvy RemoteQName a RemoteQMgr, které jsou nalezeny v definici vzdálené fronty, nebo podobně s vybranou vzdálenou frontou klastru.

Pokud při otevírání například vzdálené fronty zadáte hodnotu MQOO_RESOLVE_LOCAL_Q, bude jako přenosová fronta, do které jsou vkládány zprávy, zadána hodnota ResolvedQName . Název ResolvedQMgrje vyplněn názvem lokálního správce front, který je hostitelem přenosové fronty.

Máte-li oprávnění k procházení, vstupu nebo výstupu ve frontě, máte potřebné oprávnění k zadání tohoto příznaku ve volání MQOPEN. Není potřeba žádné speciální oprávnění.

Tato volba je platná pouze pro fronty a správce front.

MQOO_RESOLVE_LOCAL_TOPIC
Vyplňte ResolvedQName ve struktuře MQOD názvem otevřeného administrativního tématu.

VÝBĚROVÉ vysílání MQOO_NO_MULTICAST
Publikační zprávy nejsou odesílány pomocí výběrového vysílání.

Tato volba je platná pouze s volbou MQOO_OUTPUT. Pokud je uveden bez MQOO_OUTPUT, MQOPEN se vrátí s MQRC_OPTIONS_ERROR.

Tato volba je platná pouze pro téma.

HOBJ
Typ: MQHOBJ-výstup

Tento popisovač představuje přístup, který byl ustanoven k objektu. Musí být uveden při následných voláních IBM MQ , která pracují s objektem. Přestává být platný, když je vydáno volání MQCLOSE nebo když je ukončena jednotka zpracování, která definuje rozsah popisovače.

Rozsah vráceného manipulátoru objektu je stejný jako rozsah manipulátoru připojení uvedeného ve volání. Informace o rozsahu popisovače viz parametr MQCONN-Hconn .

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_MULTIPLE_PŘÍČINA
(2136, X'858 ') Bylo vráceno více kódů příčiny.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ALIAS_BASE_Q_TYPE_ERROR
(2001, X'7D1') Základní fronta aliasů není platný typ.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Zařízení pro spojku není k dispozici.

MQRC_CF_STRUC_AUTH_FAILED
(2348, X'92C') Kontrola autorizace struktury prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_ERROR
(2349, X'92D') Struktura prostředku Coupling Facility není platná.

MQRC_CF_STRUC_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility je používána.

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347, X'92B') Záhlaví seznamu struktury prostředku Coupling Facility je používáno.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

MQRC_CLUSTER_EXIT_ERROR
(2266, X'8DA') Selhala uživatelská procedura pracovní zátěže klastru.

MQRC_CLUSTER_PUT_INHIBITED
(2268, X'8DC') Volání Put bylo zablokováno pro všechny fronty v klastru.

MQRC_CLUSTER_RESOLUTION_ERROR
(2189, X'88D') Nezdařilo se rozpoznání názvu klastru.

MQRC_CLUSTER_RESOURCE_ERROR
(2269, X'8DD') Chyba prostředku klastru.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Není autorizováno pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Uklidení připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Db2 subsystém není k dispozici.

MQRC_DEF_XMIT_Q_TYPE_ERROR
(2198, X'896 ') Výchozí přenosová fronta není lokální.

MQRC_DEF_XMIT_Q_USAGE_ERROR
(2199, X'897 ') Výchozí chyba použití přenosové fronty.

MQRC_DYNAMIC_Q_NAME_ERROR
(2011, X'7DB') Název dynamické fronty není platný.

MQRC_HANDLE_NOT_AVAILABLE
(2017, X'7E1') Nejsou k dispozici žádné další popisovače.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOBJ_ERROR
(2019, X'7E3') popisovač objektu není platný.

MQRC_MULTIPLE_PŘÍČINA
(2136, X'858 ') Bylo vráceno více kódů příčiny.

MQRC_NAME_IN_USE
(2201, X'899 ') Název se používá.

MQRC_NAME_NOT_VALID_FOR_TYPE
(2194, X'892 ') Název objektu není platný pro typ objektu.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_OBJECT_ALREADY_EXISTS
(2100, X'834 ') Objekt existuje.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OBJECT_IN_USE
(2042, X'7FA') Objekt je již otevřen s konfliktními volbami.

MQRC_OBJECT_LEVEL_NEKOMPATIBILNÍ
(2360, X' 938 ') Úroveň objektu není kompatibilní.

MQRC_OBJECT_NAME_ERROR
(2152, X'868 ') Název objektu není platný.

MQRC_OBJECT_NOT_UNIQUE
(2343, X' 927 ') Objekt není jedinečný.

MQRC_OBJECT_Q_MGR_NAME_ERROR
(2153, X'869 ') Název správce front objektů není platný.

MQRC_OBJECT_RECORDS_ERROR
(2155, X'86B') Záznamy objektů nejsou platné.

MQRC_OBJECT_STRING_ERROR
(2441, X'0989 ') Pole řetězce objektu není platné

MQRC_OBJECT_TYPE_ERROR
(2043, X'7FB') Typ objektu není platný.

MQRC_OD_ERROR
(2044, X'7FC') Struktura deskriptoru objektu není platná.

MQRC_OPTION_NOT_VALID_FOR_TYPE
(2045, X'7FD') Volba není platná pro typ objektu.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

MQRC_PAGESET_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_Q_DELETED
(2052, X'804 ') Fronta byla odstraněna.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uveden do klidového stavu.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_Q_TYPE_ERROR
(2057, X'809 ') Typ fronty není platný.

MQRC_RECS_PRESENT_ERROR
(2154, X'86A') Počet přítomných záznamů není platný.

MQRC_REMOTE_Q_NAME_ERROR
(2184, X'888 ') Název vzdálené fronty není platný.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Záznamy odpovědí nejsou platné.

Chyba MQRC_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

MQRC_SELECTOR_SYNTAX_ERROR
2459 (X'099B') Bylo vydáno volání MQOPEN, MQPUT1 nebo MQSUB, ale byl zadán řetězec výběru, který obsahoval chybu syntaxe.

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Volání bylo odmítnuto uživatelskou procedurou pracovní zátěže klastru.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082, X'822 ') Neznámý alias základní fronty.

MQRC_UNKNOWN_DEF_XMIT_Q
(2197, X'895 ') Neznámá výchozí přenosová fronta.

MQRC_UNKNOWN_OBJECT_NAME
(2085, X'825 ') Neznámý název objektu.

MQRC_UNKNOWN_OBJECT_Q_MGR
(2086, X'826 ') Neznámý správce front objektů.

MQRC_UNKNOWN_REMOTE_Q_MGR
(2087, X'827 ') Neznámý vzdálený správce front.

MQRC_UNKNOWN_XMIT_Q
(2196, X'894 ') Neznámá přenosová fronta.

MQRC_WRONG_CF_LEVEL
(2366, X'93E') Struktura prostředku Coupling Facility má chybnou úroveň.

MQRC_XMIT_Q_TYPE_ERROR
(2091, X'82B') Přenosová fronta není lokální.

MQRC_XMIT_Q_USAGE_ERROR
(2092, X'82C') Přenosová fronta s chybným použitím.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Obecné poznámky k použití

1. Otevřený objekt je jeden z následujících:
   - Fronta pro:
     - Získat nebo procházet zprávy (pomocí volání MQGET)
     - Vložit zprávy (pomocí volání MQPUT)
     - Dotaz na atributy fronty (pomocí volání MQINQ)
     - Nastavit atributy fronty (pomocí volání MQSET)

     Pokud je pojmenovaná fronta modelovou frontou, vytvoří se dynamická lokální fronta. Viz parametr **ObjDesc** popsaný v části "MQOPEN-Otevření objektu" na stránce 734.

     Distribuční seznam je speciální typ objektu fronty, který obsahuje seznam front. Může být otevřen pro vložení zpráv, ale ne pro získání nebo procházení zpráv, nebo pro dotazování nebo nastavení atributů. Další podrobnosti viz poznámka k použití 8.

     Fronta, která má typ `QSGDISP(GROUP)` , je speciálním typem definice fronty, který nelze použít s voláními MQOPEN nebo MQPUT1 .
   - Seznam názvů pro zjišťování názvů front v seznamu (pomocí volání MQINQ).
   - Definice procesu pro zjišťování atributů procesu (pomocí volání MQINQ).
   - Správce front, který se má dotazovat na atributy lokálního správce front (pomocí volání MQINQ).
   - Téma pro publikování zprávy (pomocí volání MQPUT)
2. Aplikace může otevřít stejný objekt více než jednou. Pro každé otevření se vrátí jiný popisovač objektu. Každý vrácený popisovač lze použít pro funkce, pro které bylo provedeno odpovídající otevření.
3. Je-li otevíraný objekt jinou frontou než frontou klastru, veškeré rozpoznávání názvů v lokálním správci front proběhne v době volání MQOPEN. To může zahrnovat:
   - Rozlišení názvu lokální definice vzdálené fronty na název vzdáleného správce front a názvu, pod kterým je fronta ve vzdáleném správci front známa.
   - Rozlišení názvu vzdáleného správce front na název lokální přenosové fronty
   - z/OS Pouze v systému z/OS se jedná o rozlišení názvu vzdáleného správce front na název sdílené přenosové fronty používané agentem IGQ (použije se pouze v případě, že lokální a vzdálení správci front patří do stejné skupiny sdílení front).
   - Rozlišení aliasu na název základní fronty nebo objektu tématu.

   Mějte však na paměti, že následná volání MQINQ nebo MQSET pro manipulátor se vztahují pouze k názvu, který byl otevřen, a nikoli k objektu, který je výsledkem po rozpoznání názvu. Pokud je například otevřený objekt alias, atributy vrácené voláním MQINQ jsou atributy aliasu, nikoli atributy základní fronty nebo objektu tématu, na které se alias interpretuje.

   Je-li otevíraný objekt frontou klastru, může k rozpoznání názvu dojít v době volání MQOPEN nebo může být odloženo na pozdější dobu. Bod, ve kterém dochází k rozpoznání, je řízen volbami MQOO_BIND_ * určenými ve volání MQOPEN:

- MQOO_BIND_ON_OPEN
- MQOO_BIND_NOT_FIXED
- MQOO_BIND_AS_Q_DEF
- MQOO_BIND_ON_GROUP

Další informace o rozlišování názvů pro fronty klastru naleznete v tématu Rozlišení názvů .

4. Volání MQOPEN s volbou MQOO_BROWSE vytvoří kurzor procházení pro použití s voláními MQGET, která určují popisovač objektu a jednu z voleb procházení. To umožňuje skenovat frontu bez změny jejího obsahu. Zprávu nalezenou procházením lze odebrat z fronty pomocí volby MQGMO_MSG_UNDER_CURSOR.

   Pro jednu aplikaci může být aktivní více kurzorů procházení zadáním několika požadavků MQOPEN pro stejnou frontu.

5. Aplikacím spuštěným monitorem spouštěčů je předán název fronty, která je přidružena k aplikaci při jejím spuštění. Tento název fronty lze zadat v parametru **ObjDesc** pro otevření fronty. Další podrobnosti viz “MQTMC2 -zpráva spouštěče 2 (znakový formát)” na stránce 608 .

Volby dopředného čtení

Při volání MQOPEN s parametrem MQOO_READ_AHEAD povolí klient IBM MQ čtení napřed pouze v případě, že jsou splněny určité podmínky. Tyto podmínky zahrnují:

- Aplikace klienta musí být kompilována a propojena s použitím podprocesových knihoven klienta IBM MQ MQI.
- Kanál klienta musí používat protokol TCP/IP.
- Kanál musí mít nenulové nastavení SharingConversations (SHARECNV) v definici kanálu klienta i serveru.

Následující poznámky platí pro použití voleb dopředného čtení.

1. Volby dopředného čtení jsou použitelné pouze v případě, že je zadána pouze jedna z voleb MQOO_BROWSE, MQOO_INPUT_SHARED a MQOO_INPUT_EXCLUSIVE. Pokud jsou zadány volby dopředného čtení s volbami MQOO_ INQUIRE nebo MQOO_SET, není vyvolána chyba.
2. Dopředné čtení není při požadavku povoleno, pokud volby použité v prvním volání MQGET nejsou podporovány pro použití s dopředným čtením. Dopředné čtení je také zakázáno, když se klient připojuje ke správci front, který nepodporuje dopředné čtení.
3. Pokud aplikace není spuštěna jako klient IBM MQ , volby dopředného čtení se ignorují.

Fronty klastru

Následující poznámky platí pro použití front klastru.

1. Když je fronta klastru poprvé otevřena a lokální správce front není správcem front úplného úložiště, získá lokální správce front informace o frontě klastru ze správce front úplného úložiště. Když je síť zaneprázdněná, může lokálnímu správci front trvat několik sekund, než obdrží potřebné informace od správce front úložiště. V důsledku toho může aplikace vydávající volání MQOPEN čekat až 10 sekund před návratem řízení z volání MQOPEN. Pokud lokální správce front v této době neobdrží potřebné informace o frontě klastru, volání se nezdaří s kódem příčiny MQRC_CLUSTER_RESOLUTION_ERROR.
2. Při otevření fronty klastru a existenci více instancí fronty v klastru závisí otevřená instance na volbách zadaných ve volání MQOPEN:
   - Pokud uvedené volby zahrnují některou z následujících možností:
     - MQOO_BROWSE
     - MQOO_INPUT_AS_Q_DEF
     - MQOO_INPUT_EXCLUSIVE
     - MQOO_INPUT_SHARED

– MQOO_SET

instance otevřené fronty klastru musí být lokální instance. Pokud neexistuje žádná lokální instance fronty, volání MQOPEN se nezdaří.

- Pokud zadané volby nezahrnují žádnou z dříve popsaných voleb, ale zahrnují jednu nebo obě z následujících možností:
  – MQOO_DOTAZOVAT
  – MQOO_OUTPUT

  Otevřená instance je lokální instance, pokud existuje, a vzdálená instance jinak (pokud používáte výchozí nastavení CLWLUSEQ). Instance zvolená správcem front však může být změněna uživatelskou procedurou pracovní zátěže klastru (pokud existuje).

3. Pokud existuje odběr pro frontu, ale není potvrzen úplným úložištěm, objekt není přítomen v klastru a volání selže s kódem příčiny MQRC_OBJECT_NAME.

Další informace o frontách klastru naleznete v tématu Fronty klastru.

Distribuční seznamy

Pro použití distribučních seznamů platí následující poznámky.

Distribuční seznamy jsou podporovány v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

1. Pole ve struktuře MQOD musí být při otevírání distribučního seznamu nastavena takto:
   - `Version` musí být MQOD_VERSION_2 nebo vyšší.
   - `ObjectType` musí být MQOT_Q.
   - `ObjectName` musí být prázdný nebo prázdný řetězec.
   - `ObjectQMgrName` musí být prázdný nebo prázdný řetězec.
   - `RecsPresent` musí být větší než nula.
   - Jedna z hodnot `ObjectRecOffset` a `ObjectRecPtr` musí být nula a druhá nenulová.
   - Nenulová hodnota může být nejvýše jedna z hodnot `ResponseRecOffset` a `ResponseRecPtr` .
   - Musí existovat záznamy objektů `RecsPresent` , adresované buď `ObjectRecOffset` , nebo `ObjectRecPtr`. Záznamy objektů musí být nastaveny na názvy cílových front, které se mají otevřít.
   - Je-li jedna z hodnot `ResponseRecOffset` a `ResponseRecPtr` nenulová, musí existovat `RecsPresent` záznamů odpovědí. Jsou nastaveny správcem front, pokud je volání dokončeno s kódem příčiny MQRC_MULTIPLE_REASON.

   Produkt version-2 MQOD lze také použít k otevření jedné fronty, která není v distribučním seznamu, a to tak, že zajistíte, že `RecsPresent` bude nula.
2. V parametru **`Options`** jsou platné pouze následující volby otevření:
   - MQOO_OUTPUT
   - MQOO_PASS_ * _CONTEXT
   - MQOO_SET_ * _CONTEXT
   - Oprávnění MQOO_ALTERNATE_USER_AUTHORITY
   - MQOO_FAIL_IF_QUIESCING

3. Cílové fronty v rozdělovníku mohou být lokální, alias nebo vzdálené fronty, ale nemohou být modelovými frontami. Je-li zadána modelová fronta, otevření této fronty se nezdaří s kódem příčiny MQRC_Q_TYPE_ERROR. To však nebrání úspěšnému otevření jiných front v seznamu.
4. Parametry kódu dokončení a kódu příčiny jsou nastaveny takto:
   - Pokud jsou všechny otevřené operace pro fronty v rozdělovníku úspěšné nebo selhávají stejným způsobem, jsou parametry kódu dokončení a kódu příčiny nastaveny tak, aby popisovaly společný výsledek. V tomto případě nejsou nastaveny záznamy odpovědí MQRR (jsou-li poskytovány aplikací).

     Pokud je například každé otevření úspěšné, kód dokončení je nastaven na MQCC_OK a kód příčiny je nastaven na MQRC_NONE; pokud se každé otevření nezdaří, protože neexistuje žádná z front, parametry jsou nastaveny na MQCC_FAILED a MQRC_UNKNOWN_OBJECT_NAME.
   - Pokud operace otevření pro fronty v rozdělovníku nejsou všechny úspěšné nebo selhávají stejným způsobem:
     - Parametr kódu dokončení je nastaven na hodnotu MQCC_WARNING, pokud bylo alespoň jedno otevření úspěšné, a na hodnotu MQCC_FAILED, pokud se vše nezdařilo.
     - Parametr kódu příčiny je nastaven na hodnotu MQRC_MULTIPLE_REASON.
     - Záznamy odezvy (jsou-li poskytovány aplikací) jsou nastaveny na jednotlivé kódy dokončení a kódy příčiny pro fronty v rozdělovníku.
5. Po úspěšném otevření distribučního seznamu lze manipulátor `Hobj` vrácený voláním použít v následných voláních MQPUT pro vložení zpráv do front v distribučním seznamu a ve volání MQCLOSE pro vzdání se přístupu k distribučnímu seznamu. Jedinou platnou volbou zavření pro distribuční seznam je MQCO_NONE.

   Volání MQPUT1 lze také použít k vložení zprávy do distribučního seznamu. Struktura MQOD definující fronty v seznamu je určena jako parametr pro toto volání.
6. Každý úspěšně otevřený cíl v distribučním seznamu se při kontrole, zda aplikace překročila povolený maximální počet manipulátorů, počítá jako samostatný manipulátor (viz atribut správce front **`MaxHandles`**). To platí i v případě, že se dvě nebo více míst určení v rozdělovníku vyřeší na stejnou fyzickou frontu. Pokud by volání MQOPEN nebo MQPUT1 pro distribuční seznam způsobilo překročení počtu manipulátorů používaných aplikací `MaxHandles`, volání se nezdaří s kódem příčiny MQRC_HANDLE_NOT_AVAILABLE.
7. Každé místo určení, které je úspěšně otevřeno, má hodnotu svého atributu **`OpenOutputCount`** zvýšenou o jednu. Pokud se dva nebo více cílů v distribučním seznamu vyřeší na stejnou fyzickou frontu, má tato fronta svůj atribut **`OpenOutputCount`** zvýšený o počet cílů v distribučním seznamu, které se na tuto frontu vyřeší.
8. Jakákoli změna definic front, která by způsobila, že by se manipulátor stal neplatným, kdyby byly fronty otevřeny jednotlivě (například změna v cestě řešení), nezpůsobí, že se manipulátor rozdělovníku stane neplatným. Při použití manipulátoru distribučního seznamu v následném volání MQPUT však dojde k selhání pro tuto konkrétní frontu.
9. Distribuční seznam může obsahovat pouze jeden cíl.

Vzdálené fronty

Následující poznámky platí pro použití vzdálených front.

Vzdálenou frontu lze zadat jedním ze dvou způsobů v parametru **`ObjDesc`** tohoto volání.

- Zadáním názvu lokální definice vzdálené fronty pro parametr `ObjectName` . V tomto případě produkt `ObjectQMgrName` odkazuje na lokálního správce front a lze jej zadat jako mezery nebo (v programovacím jazyce C) jako prázdný řetězec.

  Ověření zabezpečení provedené lokálním správcem front ověřuje, zda má uživatel oprávnění k otevření lokální definice vzdálené fronty.
- Zadáním názvu vzdálené fronty tak, jak je známa vzdálenému správci front, do parametru `ObjectName` . V tomto případě je `ObjectQMgrName` název vzdáleného správce front.

Ověření zabezpečení provedené lokálním správcem front ověřuje, zda má uživatel oprávnění k odesílání zpráv do přenosové fronty v důsledku procesu rozpoznávání názvů.

V obou případech:

- Lokální správce front neodesílá žádné zprávy vzdálenému správci front, aby zkontroloval, zda je uživatel oprávněn vkládat zprávy do fronty.
- Když zpráva dorazí do vzdáleného správce front, může ji vzdálený správce front odmítnout, protože uživatel, který zprávu spustil, nemá příslušnou autorizaci.

Další informace naleznete v polích `ObjectName` a `ObjectQMgrName` popsaných v části “MQOD-Popisovač objektu” na stránce 482 .

Objekty

Zabezpečení

Následující poznámky se týkají aspektů zabezpečení použití MQOPEN.

Správce front provádí kontroly zabezpečení při zadání volání MQOPEN, aby ověřil, zda má identifikátor uživatele, pod kterým je aplikace spuštěna, před povolením přístupu odpovídající úroveň oprávnění. Kontrola oprávnění se provádí na názvu objektu, který se otevírá, a ne na názvu nebo názvech, které jsou výsledkem vyřešení názvu.

Pokud je otevíraným objektem alias fronta, která odkazuje na objekt tématu, provede správce front kontrolu zabezpečení názvu alias fronty před provedením kontroly zabezpečení pro dané téma, jako by byl objekt tématu použit přímo.

Pokud je otevíraný objekt objektem tématu, ať už se jedná o objekt tématu `ObjectName` samostatně nebo pomocí `ObjectString` (s nebo bez báze `ObjectName`), správce front provede kontrolu zabezpečení pomocí výsledného řetězce tématu, převzatého z objektu tématu uvedeného v souboru `ObjectNamea` v případě potřeby jej zřetězení s objektem tématu poskytnutým v produktu `ObjectString`, a poté vyhledá nejbližší objekt tématu ve stromu témat nebo nad ním, aby provedl kontrolu zabezpečení. Nemusí se jednat o stejný objekt tématu, který byl zadán v souboru `ObjectName`.

Je-li otevíraný objekt modelovou frontou, provede správce front úplnou kontrolu zabezpečení pro název modelové fronty i pro název vytvořené dynamické fronty. Je-li výsledná dynamická fronta otevřena explicitně, provede se další kontrola zabezpečení prostředků podle názvu dynamické fronty.

z/OS V systému z/OSprovádí správce front kontroly zabezpečení pouze v případě, že je povoleno zabezpečení. Další informace o kontrole zabezpečení naleznete v tématu Nastavení zabezpečení v systému z/OS .

Atributy

Následující poznámky se vztahují k atributům.

Atributy objektu se mohou měnit, zatímco aplikace má objekt otevřený. V mnoha případech si to aplikace nevšimne, ale pro určité atributy označí správce front popisovač jako neplatný. Tyto atributy jsou:

- Jakýkoli atribut, který ovlivňuje rozlišení názvu objektu. To platí bez ohledu na použité volby otevření a zahrnuje následující:
  - Změna atributu **BaseQName** alias fronty, která je otevřená.
  - Změna atributu **TargetType** alias fronty, která je otevřená.
  - Změna atributů fronty **RemoteQName** nebo **RemoteQMgrName** pro jakýkoli manipulátor otevřený pro tuto frontu nebo pro frontu, která se prostřednictvím této definice interpretuje jako alias správce front.
  - Jakákoli změna, která způsobí, že se aktuálně otevřený popisovač pro vzdálenou frontu interpretuje jako jiná přenosová fronta, nebo se vůbec neinterpretuje jako jedna. Může například zahrnovat:

- Změna atributu **XmitQName** lokální definice vzdálené fronty, bez ohledu na to, zda je definice používána pro frontu nebo pro alias správce front.
- z/OS Pouze v systému z/OS se jedná o změnu hodnoty atributu správce front **IntraGroupqueuing** nebo o změnu definice sdílené přenosové fronty (SYSTEM.QSG.TRANSMIT.QUEUE) použité agentem IGQ.

Existuje jedna výjimka: vytvoření nové přenosové fronty. Popisovač, který by byl vyřešen pro tuto frontu, kdyby byl přítomen při otevření popisovače, ale místo něj by byl převeden na výchozí přenosovou frontu, není neplatný.

– Změna atributu správce front **DefXmitQName** . V tomto případě jsou všechny otevřené popisovače, které se převedly na dříve pojmenovanou frontu (která se na ni převedla pouze proto, že se jednalo o výchozí přenosovou frontu), označeny jako neplatné. Popisovače, které byly vyřešeny pro tuto frontu z jiných důvodů, nejsou ovlivněny.

- Atribut fronty **Shareability** , pokud existují dva nebo více manipulátorů, které momentálně poskytují přístup MQOO_INPUT_SHARED pro tuto frontu nebo pro frontu, která se interpretuje pro tuto frontu. Pokud ano, všechny popisovače, které jsou otevřené pro tuto frontu nebo pro frontu, která se interpretuje pro tuto frontu, jsou označeny jako neplatné, bez ohledu na volby otevření.

  z/OS V systému z/OSjsou dříve popsané manipulátory označeny jako neplatné, pokud jeden nebo více manipulátorů aktuálně poskytuje přístup k frontě MQOO_INPUT_SHARED nebo MQOOO_INPUT_EXCLUSIVE.
- Atribut fronty **Usage** pro všechny manipulátory, které jsou otevřené pro tuto frontu, nebo pro frontu, která se interpretuje pro tuto frontu, bez ohledu na volby otevření.

Je-li manipulátor označen jako neplatný, všechna následná volání (jiná než MQCLOSE) používající tento manipulátor selžou s kódem příčiny MQRC_OBJECT_CHANGED. Aplikace musí zadat volání MQCLOSE (s použitím původního manipulátoru) a poté znovu otevřít frontu. Všechny nepotvrzené aktualizace oproti starému popisovači z předchozích úspěšných volání mohou být i nadále potvrzeny nebo vráceny zpět, jak vyžaduje logika aplikace.

Pokud změna atributu způsobí, že k tomu dojde, použijte speciální vynucenou verzi volání.

Vyvolání jazyka C

```
MQOPEN (Hconn, &ObjDesc, Options, &Hobj, &CompCode,
       &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQOPEN' USING HCONN, OBJDESC, OPTIONS, HOBJ, COMPCODE, REASON
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQOPEN (Hconn, ObjDesc, Options, Hobj, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQOPEN,(HCONN,OBJDESC,OPTIONS,HOBJ,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Vyvolání jazyka Visual Basic

Windows

```
MQOPEN Hconn, ObjDesc, Options, Hobj, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Vložit zprávu

Volání MQPUT vloží zprávu do fronty nebo distribučního seznamu nebo do tématu. Fronta, distribuční seznam nebo téma již musí být otevřené.

Syntaxe

MQPUT (*Hconn*, *Hobj*, *MsgDesc*, *PutMsgOpts*, *BufferLength*, *Buffer*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota `Hconn` byla vrácena předchozím voláním MQCONN nebo MQCONNX.

z/OS V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

HOBJ
Typ: MQHOBJ-vstup

Tento popisovač představuje frontu, do které je zpráva přidána, nebo téma, do kterého je zpráva publikována. Hodnota `Hobj` byla vrácena předchozím voláním MQOPEN, které určilo volbu MQOO_OUTPUT.

MsgDesc
Typ: MQMD-vstup/výstup

Tato struktura popisuje atributy odesílané zprávy a přijímá informace o zprávě po dokončení požadavku na vložení. Podrobnosti viz "MQMD-Deskriptor zpráv" na stránce 424.

Pokud aplikace poskytuje version-1 MQMD, data zprávy mohou mít předponu se strukturou MQMDE, která určuje hodnoty pro pole, která existují v produktu version-2 MQMD, ale nikoli version-1. Pole *Formát* v deskriptoru MQMD musí být nastaveno na hodnotu MQFMT_MD_EXTENSION, aby se označila přítomnost MQMDE. Další informace viz část "MQMDE-Rozšíření deskriptoru zpráv" na stránce 473.

Aplikace nemusí poskytovat strukturu MQMD, pokud je v polích `OriginalMsgHandle` nebo `NewMsgHandle` struktury MQPMO uveden platný popisovač zprávy. Není-li v jednom z těchto polí uvedeno nic, deskriptor zprávy je převzat z deskriptoru přidruženého k popisovači zprávy.

Pokud používáte nebo plánujete používat uživatelské procedury rozhraní API, doporučujeme, abyste explicitně dodali strukturu MQMD a nepoužívali deskriptory zpráv přidružené k popisovačům zpráv. Důvodem je skutečnost, že uživatelská procedura rozhraní API přidružená k volání MQPUT nebo MQPUT1 nemůže zjistit, které hodnoty MQMD správce front používá k dokončení požadavku MQPUT nebo MQPUT1 .

PutMsgVolba
Typ: MQPMO-vstup/výstup

Podrobnosti viz "MQPMO-Volby vložení zprávy" na stránce 502.

BufferLength
Typ: MQLONG-vstup

Délka zprávy v souboru `Buffer`. Nula je platná a označuje, že zpráva neobsahuje žádná data aplikace. Horní limit pro `BufferLength` závisí na různých faktorech:

- Je-li cílem lokální fronta nebo je-li interpretován jako lokální fronta, závisí horní limit na tom, zda:
  - Lokální správce front podporuje segmentaci.
  - Odesílající aplikace určuje příznak, který umožňuje správci front segmentovat zprávu. Tento příznak je MQMF_SEGMENTATION_ALLOWED a lze jej zadat buď v produktu version-2 MQMD, nebo v prostředí MQMDE použitém s produktem version-1 MQMD.

  Jsou-li splněny obě tyto podmínky, nesmí hodnota `BufferLength` překročit 999 999 999 999 minus hodnota pole `Offset` v deskriptoru MQMD. Nejdelší logická zpráva, kterou lze vložit, je proto 999 999 999 bajtů (když je `Offset` nula). Omezení prostředků stanovená operačním systémem nebo prostředím, ve kterém je aplikace spuštěna, však mohou mít za následek nižší limit.

Není-li splněna jedna nebo obě předchozí podmínky, nesmí hodnota BufferLength překročit menší hodnotu atributu **MaxMsgLength** fronty a atributu **MaxMsgLength** správce front.

- Pokud je cílem vzdálená fronta nebo je přeložena do vzdálené fronty, platí podmínky pro lokální fronty, ale u každého správce front, přes kterého musí zpráva projít, aby se dostala do cílové fronty; zejména:

  1. Lokální přenosová fronta používaná k dočasnému uložení zprávy v lokálním správci front
  2. Intermediační přenosové fronty (pokud existují) používané k ukládání zpráv ve správcích front na trase mezi lokálními a cílovými správci front
  3. Cílová fronta ve správci cílové fronty

  Nejdelší zpráva, kterou lze vložit, je proto řízena nejrestriktivnějším z těchto front a správců front.

  Když je zpráva v přenosové frontě, další informace jsou uloženy spolu s daty zprávy, což snižuje množství dat aplikace, která lze přenášet. V této situaci odečtěte bajty MQ_MSG_HEADER_LENGTH od hodnot MaxMsgLength přenosových front při určování limitu pro BufferLength.

  Poznámka: Při vložení zprávy lze diagnostikovat synchronně pouze selhání splnění podmínky 1 (s kódem příčiny MQRC_MSG_TOO_BIG_FOR_Q nebo MQRC_MSG_TOO_BIG_FOR_Q_MGR). Nejsou-li splněny podmínky 2 nebo 3, je zpráva přesměrována do fronty nedoručených zpráv (nedoručených zpráv), a to buď do intermediačního správce front, nebo do cílového správce front. Pokud k tomu dojde, vygeneruje se zpráva sestavy, pokud ji odesilatel požadoval.

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-vstup

Jedná se o vyrovnávací paměť obsahující data aplikace, která mají být odeslána. Vyrovnávací paměť musí být zarovnána na hranici odpovídající povaze dat ve zprávě. 4bajtové zarovnání je vhodné pro většinu zpráv (včetně zpráv obsahujících struktury záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8bajtové zarovnání.

Pokud Buffer obsahuje znaková nebo číselná data, nastavte pole CodedCharSetId a Encoding v parametru **MsgDesc** na hodnoty odpovídající datům; to umožní přijímači zprávy převést data (v případě potřeby) na znakovou sadu a kódování používané přijímačem.

Poznámka: Všechny ostatní parametry volání MQPUT musí být ve znakové sadě a kódování lokálního správce front (dané atributem správce front **CodedCharSetId** a hodnotou MQENC_NATIVE).

V programovacím jazyku C je parametr deklarován jako ukazatel-na-void; jako parametr lze zadat adresu libovolného typu dat.

Je-li parametr **BufferLength** nulový, Buffer se na něj neodkazuje; v tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo sestavovacím programem System/390 hodnotu null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje CompCode.

Pokud je CompCode MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr CompCode hodnotu MQCC_VAROVÁNÍ:

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není dokončena.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není dokončena.

MQRC_NEKONZISTENTNÍ_PERZISTENCE
(2185, X'889 ') Nekonzistentní specifikace perzistence.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

MQRC_MULTIPLE_PŘÍČINA
(2136, X'858 ') Bylo vráceno více kódů příčiny.

MQRC_PRIORITY_PŘEKRAČUJE_MAXIMUM
(2049, X'801 ') Priorita zprávy překračuje maximální podporovanou hodnotu.

MQRC_UNKNOWN_REPORT_OPTION
(2104, X'838 ') Volba (y) sestavy v deskriptoru zprávy nebyla rozpoznána.

Má-li parametr CompCode hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ALIAS_TARGTYPE_CHANGED
(2480, X'09B0') Cílový typ odběru se změnil z fronty na objekt tématu.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BACKED_OUT
(2003, X'7D3') Jednotka práce byla vrácena zpět.

MQRC_BUFFER_ERROR
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CALL_PŘERUŠENO
(2549, X'9F5') Zpracování příkazu MQPUT nebo MQCMIT bylo přerušeno a zpracování opětovného připojení nemůže znovu vytvořit určitý výsledek.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Zařízení pro spojku není k dispozici.

MQRC_CF_STRUC_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility je používána.

MQRC_CFGR_ERROR
(2416, X' 970 ') Struktura parametru skupiny PCF MQCFGR v datech zprávy je neplatná.

MQRC_CFH_ERROR
(2235, X'8BB') Struktura záhlaví PCF není platná.

MQRC_CFIF_ERROR
(2414, X'96E') Struktura parametrů celočíselného filtru PCF v datech zprávy je neplatná.

MQRC_CFIL_ERROR
(2236, X'8BC') Struktura parametru seznamu celých čísel PCF nebo struktura parametrů seznamu celých čísel PCIF*64 není platná.

MQRC_CFIN_ERROR
(2237, X'8BD') Struktura celočíselných parametrů PCF nebo struktura celočíselných parametrů PCIF*64 není platná.

MQRC_CFSF_ERROR
(2415, X'96F') Struktura parametru filtru řetězce PCF v datech zprávy je neplatná.

MQRC_CFSL_ERROR
(2238, X'8BE') Struktura parametru seznamu řetězců PCF není platná.

MQRC_CFST_ERROR
(2239, X'8BF') Struktura parametru řetězce PCF není platná.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

MQRC_CLUSTER_EXIT_ERROR
(2266, X'8DA') Selhala uživatelská procedura pracovní zátěže klastru.

MQRC_CLUSTER_RESOLUTION_ERROR
(2189, X'88D') Nezdařilo se rozpoznání názvu klastru.

MQRC_CLUSTER_RESOURCE_ERROR
(2269, X'8DD') Chyba prostředku klastru.

MQRC_COD_NOT_VALID_FOR_XCF_Q
(2106, X'83A') Volba sestavy COD není platná pro frontu XCF.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Není autorizováno pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Uklidení připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_CONTENT_ERROR
2554 (X'09FA') Obsah zprávy nelze analyzovat, aby se zjistilo, zda má být zpráva doručena odběrateli s rozšířeným selektorem zpráv.

MQRC_CONTEXT_HANDLE_ERROR
(2097, X'831 ') popisovač fronty, na který se odkazuje, neuloží kontext.

MQRC_CONTEXT_NOT_AVAILABLE
(2098, X'832 ') Kontext není k dispozici pro popisovač fronty, na který se odkazuje.

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

MQRC_DH_ERROR
(2135, X'857 ') Struktura záhlaví distribuce není platná.

MQRC_DLH_ERROR
(2141, X'85D') Struktura záhlaví nedoručených položek není platná.

MQRC_EPH_ERROR
(2420, X' 974 ') Vestavěná struktura PCF není platná.

MQRC_EXPIRY_ERROR
(2013, X'7DD') Doba vypršení platnosti není platná.

MQRC_FEEDBACK_CHYBA
(2014, X'7DE') Kód zpětné vazby není platný.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální konflikt pracovních jednotek.

MQRC_GROUP_ID_ERROR
(2258, X'8D2') Identifikátor skupiny není platný.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Rukojeť používaná pro globální jednotku práce.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HEADER_CHYBA
(2142, X'85E') Struktura záhlaví MQ není platná.

MQRC_HOBJ_ERROR
(2019, X'7E3') popisovač objektu není platný.

MQRC_IIH_CHYBA
(2148, X'864 ') Struktura záhlaví informací IMS není platná.

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není dokončena.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není dokončena.

MQRC_NEKONZISTENTNÍ_PERZISTENCE
(2185, X'889 ') Nekonzistentní specifikace perzistence.

MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s lokální jednotkou práce.

MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zpráv není platný.

MQRC_MDE_ERROR
(2248, X'8C8') Rozšíření deskriptoru zprávy není platné.

MQRC_MISSING_REPLY_TO_Q
(2027, X'7EB') Chybí fronta pro odpověď nebo byla použita hodnota MQPMO_SUPPRESS_REPLYTO.

MQRC_MISSING_WIH
(2332, X'91C') Data zprávy nezačínají na MQWIH.

MQRC_MSG_FLAGS_ERROR
(2249, X'8C9') Příznaky zprávy nejsou platné.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Pořadové číslo zprávy není platné.

MQRC_MSG_TOO_BIG_FOR_Q
(2030, X'7EE') Délka zprávy je větší než maximum pro frontu.

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031, X'7EF') Délka zprávy pro správce front je větší než maximum.

MQRC_MSG_TYPE_ERROR
(2029, X'7ED') Typ zprávy v deskriptoru zprávy není platný.

MQRC_MULTIPLE_PŘÍČINA
(2136, X'858 ') Bylo vráceno více kódů příčiny.

MQRC_NO_DESTINATIONS_AVAILABLE
(2270, X'8DE') Nejsou k dispozici žádné cílové fronty.

MQRC_NOT_OPEN_FOR_OUTPUT
(2039, X'7F7') Fronta není otevřena pro výstup.

MQRC_NOT_OPEN_FOR_PASS_ALL
(2093, X'82D') Fronta není otevřena pro průchod celého kontextu.

MQRC_NOT_OPEN_FOR_PASS_IDENT
(2094, X'82E') Fronta není otevřena pro průchozí kontext identity.

MQRC_NOT_OPEN_FOR_SET_ALL
(2095, X'82F') Fronta není otevřena pro nastavení celého kontextu.

MQRC_NOT_OPEN_FOR_SET_IDENT
(2096, X'830 ') Fronta není otevřena pro nastavení kontextu identity.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu se od doby otevření změnila.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OFFSET_ERROR
(2251, X'8CB') Offset segmentu zprávy není platný.

MQRC_OPEN_FAILED
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_ORIGINAL_LENGTH_ERROR
(2252, X'8CC') Původní délka není platná.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

MQRC_PAGESET_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_PCF_ERROR
(2149, X'865 ') Struktury PCF nejsou platné.

MQRC_PERSISTENCE_ERROR
(2047, X'7FF') Perzistence není platná.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

MQRC_PMO_ERROR
(2173, X'87D') Struktura voleb vkládání zpráv je neplatná.

MQRC_PMO_RECORD_FLAGS_ERROR
(2158, X'86E') Příznaky záznamu vložení zprávy jsou neplatné.

MQRC_PRIORITY_ERROR
(2050, X'802 ') Priorita zprávy není platná.

MQRC_PUBLICATION_FAILURE
(2502, X'9C6') Publikace nebyla doručena žádnému z odběratelů.

MQRC_PUT_INHIBITED
(2051, X'803 ') Volání Put bylo zablokováno pro frontu, pro frontu, do které je tato fronta přeložena, nebo pro téma.

MQRC_PUT_MSG_RECORDS_ERROR
(2159, X'86F') Záznamy vkládající zprávy jsou neplatné.

MQRC_PUT_NOT_ZACHOVÁNO
(2479, X'09AF') Publikace nebyla uchována

MQRC_Q_DELETED
(2052, X'804 ') Fronta byla odstraněna.

MQRC_Q_FULL
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uveden do klidového stavu.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_Q_SPACE_NOT_AVAILABLE
(2056, X'808 ') Není k dispozici místo na disku pro frontu.

MQRC_RECONNECT_FAILED
(2548, X'9F4') Po opětovném připojení došlo k chybě při obnovování popisovačů pro znovu připojitelné připojení.

MQRC_RECS_PRESENT_ERROR
(2154, X'86A') Počet přítomných záznamů není platný.

MQRC_REPORT_OPTIONS_ERROR
(2061, X'80D') Volby sestavy v deskriptoru zprávy jsou neplatné.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Záznamy odpovědí nejsou platné.

MQRC_RFH_ERROR
(2334, X'91E') Struktura MQRFH nebo MQRFH2 není platná.

MQRC_RMH_ERROR
(2220, X'8AC') Struktura záhlaví referenční zprávy není platná.

MQRC_SEGMENT_DÉLKA_NULA
(2253, X'8CD') Délka dat v segmentu zprávy je nula.

MQRC_SEGMENTS_NOT_SUPPORTED
(2365, X'93D') Segmenty nejsou podporovány.

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') Existuje možný odběratel pro publikování, ale správce front nemůže zkontrolovat, zda má odeslat publikování odběrateli.

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Volání bylo odmítnuto uživatelskou procedurou pracovní zátěže klastru.

MQRC_STORAGE_CLASS_ERROR
(2105, X'839 ') Chyba paměťové třídy.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

MQRC_SYNCPOINT_LIMIT_DOSAŽENO
(2024, X'7E8') V rámci aktuální pracovní jednotky nelze zpracovat žádné další zprávy.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora synchronizační funkce není k dispozici.

MQRC_TM_ERROR
(2265, X'8D9') Struktura zprávy spouštěče není platná.

MQRC_TMC_ERROR
(2191, X'88F') Struktura zprávy spouštěče znaků je neplatná.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Registrace v globální pracovní jednotce se nezdařila.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotky práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

MQRC_WIH_ERROR
(2333, X'91D') Struktura MQWIH není platná.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Dodána nesprávná verze MQMD.

MQRC_XQH_ERROR
(2260, X'8D4') Struktura záhlaví přenosové fronty není platná.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití tématu

1. Pro použití témat platí následující poznámky:

   a. Při použití příkazu MQPUT k publikování zpráv v tématu, kde jednomu nebo více odběratelům daného tématu nelze danou publikaci poskytnout kvůli problému s frontou odběratele (například je plná), je kód příčiny vrácený volání MQPUT a chování při doručení závislé na nastavení atributů PMSGDLV nebo NPMSGDLV v tématu TOPIC. Doručení publikace do fronty nedoručených zpráv v případě, že je zadána volba MQRO_DEAD_LETTER_Q, nebo zrušení zprávy v případě, že je zadána volba MQRO_DISCARD_MSG, je považováno za úspěšné doručení zprávy. Nejsou-li doručena žádná publikování, příkaz MQPUT se vrátí s hodnotou MQRC_PUBLICATION_FAILURE. K tomu může dojít v následujících případech:

      - Zpráva je publikována do TOPIC s PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavenou na ALL a jakýkoli odběr (trvalý nebo ne) má frontu, která nemůže přijmout publikování.
      - Zpráva je publikována do TOPIC s parametrem PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastaveným na ALLDUR a trvalý odběr má frontu, která nemůže přijmout publikování.

      Příkaz MQPUT se může vrátit s hodnotou MQRC_NONE, i když publikování nebylo možné některým odběratelům doručit v následujících případech:

      - Zpráva je publikována do TOPIC s PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavenou na ALLAVAIL a jakýkoli odběr, trvalý nebo ne, má frontu, která nemůže přijmout publikování.
      - Zpráva je publikována do TOPIC s parametrem PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastaveným na ALLDUR a netrvalý odběr má frontu, která nemůže přijmout publikování.

      Pomocí atributu tématu USEDLQ můžete určit, zda má být fronta nedoručených zpráv použita v případě, že zprávy publikování nelze doručit do správné fronty odběratele. Další informace o použití funkce USEDLQ naleznete v tématu DEFINE TOPIC.

   b. Pokud pro používané téma nejsou žádní odběratelé, publikovaná zpráva nebude odeslána do žádné fronty a bude vyřazena. Nezáleží na tom, zda je zpráva trvalá nebo dočasná, nebo zda má neomezené vypršení platnosti nebo má čas vypršení platnosti, je stále vyřazena, pokud nejsou

žádní odběratelé. Výjimkou je situace, kdy má být zpráva uchována. V takovém případě, přestože není odeslána do žádné fronty odběratelů, je uložena pro téma, které má být doručeno novým odběrům nebo odběratelům, kteří žádají o zachovaná publikování pomocí MQSUBRQ.

MQPUT a MQPUT1

Ke vložení zpráv do fronty můžete použít volání MQPUT i MQPUT1 . Volání, které se má použít, závisí na okolnostech.

- Použijte volání MQPUT k umístění více zpráv do stejné fronty.

  Nejprve je zadáno volání MQOPEN uvádějící volbu MQOO_OUTPUT následované jedním nebo více požadavky MQPUT na přidání zpráv do fronty; nakonec je fronta uzavřena voláním MQCLOSE. To poskytuje lepší výkon než opakované použití volání MQPUT1 .
- Volání MQPUT1 slouží k vložení pouze jedné zprávy do fronty.

  Toto volání zapouzdřuje volání MQOPEN, MQPUT a MQCLOSE do jednoho volání a minimalizuje počet volání, která je třeba zadat.

Cílové fronty

Pro použití cílových front platí následující poznámky:

1. Pokud aplikace vloží posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, pokud jsou splněny podrobné podmínky. Některé podmínky platí pro lokální i vzdálené cílové fronty; jiné podmínky platí pouze pro vzdálené cílové fronty.

   Podmínky platné pro lokální a vzdálené cílové fronty

   - Všechna volání MQPUT jsou ve stejné pracovní jednotce, nebo žádná z nich není v pracovní jednotce.

     Uvědomte si, že když jsou zprávy vkládány do určité fronty v rámci jedné pracovní jednotky, mohou být zprávy z jiných aplikací prokládány s posloupností zpráv ve frontě.
   - Všechna volání MQPUT se provádějí pomocí stejného popisovače objektu *Hobj*.

     V některých prostředích je posloupnost zpráv zachována i při použití různých manipulátorů objektů, pokud jsou volání provedena ze stejné aplikace. Význam *stejné aplikace* je určen prostředím:

     – z/OS V systému z/OSje aplikace:

       - V případě systému CICSse jedná o úlohu CICS .
       - V případě IMSse jedná o úlohu
       - V případě dávky z/OS se jedná o úlohu

     – IBM i V systému IBM ije úlohou aplikace.

     – ALW V systému AIX, Linux, and Windowsje aplikací podproces.
   - Všechny zprávy mají stejnou prioritu.
   - Zprávy nejsou vloženy do fronty klastru se zadanou hodnotou MQOO_BIND_NOT_FIXED (nebo s hodnotou MQOO_BIND_AS_AS_Q_DEF v platnosti, když má atribut fronty DefBind hodnotu MQBND_BIND_NOT_FIXED).

   Další podmínky, které platí pro vzdálené cílové fronty

   - Existuje pouze jedna cesta od odesílajícího správce front k cílovému správci front.

     Pokud se některé zprávy v posloupnosti mohou nacházet v jiné cestě (například kvůli opětovné konfiguraci, vyvážení provozu nebo výběru cesty na základě velikosti zprávy), pořadí zpráv v cílovém správci front nelze zaručit.
   - Zprávy nejsou dočasně umisťovány do front nedoručených zpráv ve správcích front pro odesílání, zprostředkující nebo cílové fronty.

Pokud je jedna nebo více zpráv dočasně vloženo do fronty nedoručených zpráv (například proto, že přenosová fronta nebo cílová fronta jsou dočasně zaplněné), mohou zprávy do cílové fronty dorazit mimo pořadí.

- Zprávy jsou buď všechny trvalé, nebo všechny dočasné.

  Pokud má kanál na trase mezi odesílajícími a cílovými správci front nastaven atribut **NonPersistentMsgSpeed** na hodnotu MQNPMS_FAST, mohou přechodné zprávy skákat před trvalými zprávami, což vede k tomu, že pořadí trvalých zpráv vzhledem k dočasným zprávám nebude zachováno. Pořadí vzájemně relativních trvalých zpráv a přechodných zpráv vzájemně relativních je však zachováno.

Nejsou-li tyto podmínky splněny, můžete použít skupiny zpráv k zachování pořadí zpráv, ale to vyžaduje, aby odesílající i přijímající aplikace používaly podporu seskupení zpráv. Další informace o skupinách zpráv viz:

- MQMD-pole MsgFlags
- MQPMO_LOGICAL_ORDER
- MQGMO_LOGICAL_ORDER

Distribuční seznamy

Pro použití distribučních seznamů platí následující poznámky.

Distribuční seznamy jsou podporovány v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

1. Zprávy můžete do distribučního seznamu vložit buď pomocí MQPMO version-1 , nebo pomocí MQPMO version-2 . Pokud použijete volbu version-1 MQPMO (nebo version-2 MQPMO s hodnotou `RecsPresent` rovnou nule), aplikace nemůže poskytnout žádné záznamy vložených zpráv ani záznamy odpovědí. Nemůžete identifikovat fronty, které zaznamenají chyby, pokud je zpráva úspěšně odeslána do některých front v rozdělovníku a ne do jiných.

   Pokud aplikace poskytuje záznamy vkládaných zpráv nebo záznamy odpovědí, nastavte pole `Version` na hodnotu MQPMO_VERSION_2.

   Můžete také použít version-2 MQPMO k odeslání zpráv do jedné fronty, která není v distribučním seznamu, a to tak, že zajistíte, že `RecsPresent` bude nula.
2. Parametry kódu dokončení a kódu příčiny jsou nastaveny takto:
   - Pokud jsou všechny operace vložení do front v rozdělovníku úspěšné nebo selhávají stejným způsobem, jsou parametry kódu dokončení a kódu příčiny nastaveny tak, aby popisovaly společný výsledek. V tomto případě nejsou nastaveny záznamy odpovědí MQRR (jsou-li poskytovány aplikací).

     Pokud je například každé vložení úspěšné, kód dokončení a kód příčiny jsou nastaveny na MQCC_OK a MQRC_NONE; pokud se každé vložení nezdaří, protože jsou blokována všechna fronta pro vložení, parametry jsou nastaveny na MQCC_FAILED a MQRC_PUT_INHIBITED.
   - Pokud vložení do front v rozdělovníku nejsou všechna úspěšná nebo selhávají stejným způsobem:
     - Parametr kódu dokončení je nastaven na hodnotu MQCC_WARNING, pokud bylo alespoň jedno vložení úspěšné, a na hodnotu MQCC_FAILED, pokud se vše nezdařilo.
     - Parametr kódu příčiny je nastaven na hodnotu MQRC_MULTIPLE_REASON.

– Záznamy odezvy (jsou-li poskytovány aplikací) jsou nastaveny na jednotlivé kódy dokončení a kódy příčiny pro fronty v rozdělovníku.

Pokud vložení do místa určení selže, protože otevření pro toto místo určení se nezdařilo, pole v záznamu odpovědi jsou nastavena na MQCC_FAILED a MQRC_OPEN_FAILED; toto místo určení je zahrnuto v souboru `InvalidDestCount`.

3. Pokud se cíl v rozdělovníku interpretuje jako lokální fronta, zpráva se umístí do této fronty v normální formě (tj. ne jako zpráva rozdělovníku). Pokud se více než jeden cíl interpretuje na stejnou lokální frontu, umístí se do fronty jedna zpráva pro každý takový cíl.

   Pokud se cíl v rozdělovníku interpretuje jako vzdálená fronta, zpráva se umístí do příslušné přenosové fronty. Pokud se několik míst určení vyřeší do stejné přenosové fronty, může být do přenosové fronty umístěna jedna zpráva distribučního seznamu obsahující tato místa určení, a to i v případě, že tato místa určení nebyla sousedící se seznamem míst určení poskytnutým aplikací. To však lze provést pouze v případě, že přenosová fronta podporuje zprávy distribučního seznamu (viz DistLists).

   Pokud přenosová fronta nepodporuje distribuční seznamy, jedna kopie zprávy v normálním formátu se umístí do přenosové fronty pro každé místo určení, které používá tuto přenosovou frontu.

   Pokud je rozdělovník s daty zpráv aplikace příliš velký pro přenosovou frontu, zpráva rozdělovníku se rozdělí na menší zprávy rozdělovníku, z nichž každá obsahuje méně cílů. Pokud se data zpráv aplikace vejdou pouze do fronty, nelze zprávy distribučního seznamu vůbec použít a správce front vygeneruje jednu kopii zprávy v normální podobě pro každé místo určení, které používá danou přenosovou frontu.

   Pokud mají různá místa určení jinou prioritu zpráv nebo perzistenci zpráv (k tomu může dojít v případě, že aplikace určuje hodnotu MQPRI_PRIORITY_AS_Q_DEF nebo MQPER_PERSISTENCE_AS_Q_DEF), nejsou zprávy uloženy ve stejné zprávě distribučního seznamu. Místo toho správce front generuje tolik zpráv distribučního seznamu, kolik je potřeba pro přizpůsobení různých hodnot priority a perzistence.

4. Vložení do rozdělovníku může mít za následek:

   - jedna zpráva rozdělovníku nebo
   - Počet menších zpráv distribučního seznamu, nebo
   - směs zpráv distribučního seznamu a běžných zpráv, nebo
   - Pouze normální zprávy.

   Který z výše uvedených skutečností závisí na tom, zda:

   - Místa určení v seznamu jsou lokální, vzdálená nebo smíšená.
   - Místa určení mají stejnou prioritu zpráv a perzistenci zpráv.
   - Přenosové fronty mohou obsahovat zprávy distribučního seznamu.
   - Maximální délka zpráv přenosových front je dostatečně velká, aby se vešel do zprávy ve formě rozdělovníku.

   Avšak bez ohledu na to, která z výše uvedených zpráv se vyskytne, každá *fyzická* výsledná zpráva (tj. každá normální zpráva nebo zpráva distribučního seznamu vyplývající z vložení) se počítá jako *jedna* zpráva, když:

   - Kontrola, zda aplikace překročila povolený maximální počet zpráv v transakci (viz atribut správce front **MaxUncommittedMsgs**).
   - Kontrola, zda jsou splněny podmínky spouštěče.
   - Zvýšení hloubky fronty a kontrola, zda by byla překročena maximální hloubka fronty fronty.

5. Jakákoli změna definic front, která by způsobila, že by se manipulátor stal neplatným, kdyby byly fronty otevřeny jednotlivě (například změna v cestě řešení), nezpůsobí, že se manipulátor rozdělovníku stane neplatným. Při použití manipulátoru distribučního seznamu v následném volání MQPUT však dojde k selhání pro tuto konkrétní frontu.

Záhlaví

Pokud je zpráva vložena s jednou nebo více strukturami záhlaví IBM MQ na začátek dat zprávy aplikace, provede správce front určité kontroly struktur záhlaví, aby ověřil, zda jsou platné. Pokud správce front zjistí chybu, volání selže s příslušným kódem příčiny. Provedené kontroly se liší v závislosti na konkrétních strukturách, které jsou přítomny:

- Kontroly se provádějí pouze v případě, že je pro volání MQPUT nebo MQPUT1 použita volba version-2 nebo novější MQMD. Při použití deskriptoru MQMD version-1 se kontroly neprovádějí, a to ani v případě, že je na začátku dat zprávy přítomno prostředí MQMDE.
- Struktury, které nejsou podporovány lokálním správcem front, a struktury, které následují po prvním MQDLH ve zprávě, nejsou ověřeny.
- Struktury MQDH a MQMDE jsou zcela ověřeny správcem front.
- Ostatní struktury jsou částečně ověřeny správcem front (ne všechna pole jsou zaškrtnuta).

Obecné kontroly prováděné správcem front zahrnují následující:

- Pole `StrucId` musí být platné.
- Pole `Version` musí být platné.
- Pole `StrucLength` musí uvádět hodnotu, která je dostatečně velká, aby zahrnovala strukturu plus jakákoli data proměnné délky, která tvoří část struktury.
- Pole `CodedCharSetId` nesmí být nula nebo záporná hodnota, která není platná (MQCCSI_DEFAULT, MQCCSI_EMBEDDED, MQCCSI_Q_MGR a MQCCSI_UNDEFINED nejsou platné ve většině struktur záhlaví IBM MQ).
- Parametr **`BufferLength`** volání musí uvádět hodnotu, která je dostatečně velká, aby zahrnovala strukturu (struktura nesmí přesahovat konec zprávy).

Kromě obecných kontrol struktur musí být splněny tyto podmínky:

- Součet délek struktur ve zprávě PCF se musí rovnat délce určené parametrem **`BufferLength`** ve volání MQPUT nebo MQPUT1 . Zpráva PCF je zpráva, která má název formátu MQFMT_ADMIN, MQFMT_EVENT nebo MQFMT_PCF.
- Struktura IBM MQ nesmí být zkrácena, s výjimkou následujících situací, kdy jsou zkrácené struktury povoleny:
  - Zprávy, které jsou zprávami sestavy.
  - Zprávy PCF.
  - Zprávy obsahující strukturu MQDLH. (Struktury následující po prvním MQDLH lze zkrátit; struktury předcházející MQDLH nikoli.)
- Struktura IBM MQ nesmí být rozdělena na dva nebo více segmentů; struktura musí být zcela obsažena v jednom segmentu.

Vyrovnávací paměť

Pro programovací jazyk Visual Basic platí následující body:

- Pokud je velikost parametru **`Buffer`** menší než délka určená parametrem **`BufferLength`** , volání selže s kódem příčiny MQRC_BUFFER_LENGTH_ERROR.
- Parametr **`Buffer`** je deklarován jako typ `String`. Pokud data, která mají být umístěna do fronty, nejsou typu `String`, použijteMQPUTJakékoli volání namísto volání MQPUT.

  Volání MQPUTAny má stejné parametry jako volání MQPUT s tím rozdílem, že parametr **`Buffer`** je deklarován jako typ `Any`, což umožňuje umístění libovolného typu dat do fronty. To však znamená, že `Buffer` nelze zkontrolovat, aby se zajistilo, že má velikost alespoň `BufferLength` bajtů.

Vyvolání jazyka C

```
MQPUT (Hconn, Hobj, &MsgDesc, &PutMsgOpts, BufferLength, Buffer,
      &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
     CALL 'MQPUT' USING HCONN, HOBJ, MSGDESC, PUTMSGOPTS, BUFFERLENGTH,
                        BUFFER, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQPUT (Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer,
           CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQPUT,(HCONN,HOBJ,MSGDESC,PUTMSGOPTS,BUFFERLENGTH, X
               BUFFER,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Vyvolání jazyka Visual Basic

Windows

```
MQPUT Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode,
      Reason
```

Deklarujte parametry následujícím způsobem:

```
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 -Vložit jednu zprávu

Volání MQPUT1 vloží jednu zprávu do fronty, distribučního seznamu nebo do tématu.

Fronta, distribuční seznam nebo téma nemusí být otevřené.

Syntaxe

MQPUT1 (*Hconn*, *ObjDesc*, *MsgDesc*, *PutMsgOpts*, *BufferLength*, *Buffer*, *CompCode*, *Reason*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

z/OS V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

ObjDesc
Typ: MQOD-vstup/výstup

Jedná se o strukturu, která identifikuje frontu, do které je zpráva přidána, nebo téma, do kterého je zpráva publikována. Podrobnosti viz “MQOD-Popisovač objektu” na stránce 482.

Pokud je struktura fronta, uživatel musí mít oprávnění k otevření fronty pro výstup. Fronta nesmí být modelovou frontou.

MsgDesc
Typ: MQMD-vstup/výstup

Tato struktura popisuje atributy odesílané zprávy a po dokončení požadavku na vložení přijímá informace o zpětné vazbě. Podrobnosti viz "MQMD-Deskriptor zpráv" na stránce 424.

Pokud aplikace poskytuje version-1 MQMD, data zprávy mohou mít předponu se strukturou MQMDE, která určuje hodnoty pro pole, která existují v produktu version-2 MQMD, ale nikoli version-1. Nastavte pole `Format` v MQMD na MQFMT_MD_EXTENSION, abyste označili, že je MQMDE přítomen. Další informace viz část "MQMDE-Rozšíření deskriptoru zpráv" na stránce 473.

Aplikace nemusí poskytovat strukturu MQMD, pokud je v poli `MsgHandle` struktury MQGMO nebo v polích `OriginalMsgHandle` nebo `NewMsgHandle` struktury MQPMO uveden platný popisovač zprávy. Není-li v jednom z těchto polí uvedeno nic, deskriptor zprávy je převzat z deskriptoru přidruženého k popisovači zprávy.

PutMsgVolba
Typ: MQPMO-vstup/výstup

Podrobnosti viz "MQPMO-Volby vložení zprávy" na stránce 502.

BufferLength
Typ: MQLONG-vstup

Délka zprávy v souboru `Buffer`. Nula je platná a označuje, že zpráva neobsahuje žádná data aplikace. Horní limit závisí na různých faktorech; popis parametru **`BufferLength`** viz "MQPUT-Vložit zprávu" na stránce 751 .

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-vstup

Jedná se o vyrovnávací paměť obsahující data zprávy aplikace, která mají být odeslána. Zarovnejte vyrovnávací paměť na hranici odpovídající povaze dat ve zprávě. 4bajtové zarovnání je vhodné pro většinu zpráv (včetně zpráv obsahujících struktury záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8bajtové zarovnání.

Pokud `Buffer` obsahuje znaková nebo číselná data, nastavte pole `CodedCharSetId` a `Encoding` v parametru **`MsgDesc`** na hodnoty odpovídající datům; to umožní přijímači zprávy převést data (v případě potřeby) na znakovou sadu a kódování používané přijímačem.

Poznámka: Všechny ostatní parametry volání MQPUT1 musí být ve znakové sadě a kódování lokálního správce front (dané atributem správce front **`CodedCharSetId`** a hodnotou MQENC_NATIVE).

V programovacím jazyku C je parametr deklarován jako ukazatel-na-void; jako parametr lze zadat adresu libovolného typu dat.

Je-li parametr **`BufferLength`** nulový, `Buffer` se na něj neodkazuje; v tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo sestavovacím programem System/390 hodnotu null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina

Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje CompCode.

Pokud je CompCode MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr CompCode hodnotu MQCC_VAROVÁNÍ:

MQRC_MULTIPLE_PŘÍČINA
(2136, X'858 ') Bylo vráceno více kódů příčiny.

MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není dokončena.

MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není dokončena.

MQRC_PRIORITY_PŘEKRAČUJE_MAXIMUM
(2049, X'801 ') Priorita zprávy překračuje maximální podporovanou hodnotu.

MQRC_UNKNOWN_REPORT_OPTION
(2104, X'838 ') Volby sestavy v deskriptoru zprávy nebyly rozpoznány.

Má-li parametr CompCode hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ALIAS_BASE_Q_TYPE_ERROR
(2001, X'7D1') Základní fronta aliasů není platný typ.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BACKED_OUT
(2003, X'7D3') Jednotka práce byla vrácena zpět.

MQRC_BUFFER_ERROR
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') prostředek pro spojení není k dispozici.

MQRC_CF_STRUC_AUTH_FAILED
(2348, X'92C') Kontrola autorizace struktury prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_ERROR
(2349, X'92D') Struktura prostředku Coupling Facility není platná.

MQRC_CF_STRUC_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility je používána.

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347, X'92B') Záhlaví seznamu struktury prostředku Coupling Facility je používáno.

MQRC_CFGR_ERROR
(2416, X' 970 ') Struktura parametru skupiny PCF MQCFGR v datech zprávy je neplatná.

MQRC_CFH_ERROR
(2235, X'8BB') Struktura záhlaví PCF není platná.

MQRC_CFIF_ERROR
(2414, X'96E') Struktura parametrů celočíselného filtru PCF v datech zprávy je neplatná.

MQRC_CFIL_ERROR
(2236, X'8BC') Struktura parametru seznamu celých čísel PCF nebo struktura parametrů seznamu celých čísel PCIF*64 není platná.

MQRC_CFIN_ERROR
(2237, X'8BD') Struktura celočíselných parametrů PCF nebo struktura celočíselných parametrů PCIF*64 není platná.

MQRC_CFSF_ERROR
(2415, X'96F') Struktura parametru filtru řetězce PCF v datech zprávy je neplatná.

MQRC_CFSL_ERROR
(2238, X'8BE') Struktura parametru seznamu řetězců PCF není platná.

MQRC_CFST_ERROR
(2239, X'8BF') Struktura parametru řetězce PCF není platná.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

MQRC_CLUSTER_EXIT_ERROR
(2266, X'8DA') Selhala uživatelská procedura pracovní zátěže klastru.

MQRC_CLUSTER_RESOLUTION_ERROR
(2189, X'88D') Nezdařilo se rozpoznání názvu klastru.

MQRC_CLUSTER_RESOURCE_ERROR
(2269, X'8DD') Chyba prostředku klastru.

MQRC_COD_NOT_VALID_FOR_XCF_Q
(2106, X'83A') Volba sestavy COD není platná pro frontu XCF.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Není autorizováno pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Uklidení připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_CONTENT_ERROR
2554 (X'09FA') Obsah zprávy nelze analyzovat, aby se zjistilo, zda lze zprávu doručit odběrateli s rozšířeným selektorem zpráv.

MQRC_CONTEXT_HANDLE_ERROR
(2097, X'831 ') popisovač fronty, na který se odkazuje, neuloží kontext.

MQRC_CONTEXT_NOT_AVAILABLE
(2098, X'832 ') Kontext není k dispozici pro popisovač fronty, na který se odkazuje.

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Db2 subsystém není k dispozici.

MQRC_DEF_XMIT_Q_TYPE_ERROR
(2198, X'896 ') Výchozí přenosová fronta není lokální.

MQRC_DEF_XMIT_Q_USAGE_ERROR
(2199, X'897 ') Výchozí chyba použití přenosové fronty.

MQRC_DH_ERROR
(2135, X'857 ') Struktura záhlaví distribuce není platná.

MQRC_DLH_ERROR
(2141, X'85D') Struktura záhlaví nedoručených položek není platná.

MQRC_EPH_ERROR
(2420, X' 974 ') Vestavěná struktura PCF není platná.

MQRC_EXPIRY_ERROR
(2013, X'7DD') Doba vypršení platnosti není platná.

MQRC_FEEDBACK_CHYBA
(2014, X'7DE') Kód zpětné vazby není platný.

MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální konflikt pracovních jednotek.

MQRC_GROUP_ID_ERROR
(2258, X'8D2') Identifikátor skupiny není platný.

MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Rukojeť používaná pro globální jednotku práce.

MQRC_HANDLE_NOT_AVAILABLE
(2017, X'7E1') Nejsou k dispozici žádné další popisovače.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HEADER_CHYBA
(2142, X'85E') IBM MQ struktura záhlaví není platná.

MQRC_IIH_CHYBA
(2148, X'864 ') Struktura záhlaví informací IMS není platná.

MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s lokální jednotkou práce.

MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zpráv není platný.

MQRC_MDE_ERROR
(2248, X'8C8') Rozšíření deskriptoru zprávy není platné.

MQRC_MISSING_REPLY_TO_Q
(2027, X'7EB') Chybí fronta pro odpověď.

MQRC_MISSING_WIH
(2332, X'91C') Data zprávy nezačínají na MQWIH.

MQRC_MSG_FLAGS_ERROR
(2249, X'8C9') Příznaky zprávy nejsou platné.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Pořadové číslo zprávy není platné.

MQRC_MSG_TOO_BIG_FOR_Q
(2030, X'7EE') Délka zprávy je větší než maximum pro frontu.

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031, X'7EF') Délka zprávy pro správce front je větší než maximum.

MQRC_MSG_TYPE_ERROR
(2029, X'7ED') Typ zprávy v deskriptoru zprávy není platný.

MQRC_MULTIPLE_PŘÍČINA
(2136, X'858 ') Bylo vráceno více kódů příčiny.

MQRC_NO_DESTINATIONS_AVAILABLE
(2270, X'8DE') Nejsou k dispozici žádné cílové fronty.

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_OBJECT_IN_USE
(2042, X'7FA') Objekt je již otevřen s konfliktními volbami.

MQRC_OBJECT_LEVEL_NEKOMPATIBILNÍ
(2360, X' 938 ') Úroveň objektu není kompatibilní.

MQRC_OBJECT_NAME_ERROR
(2152, X'868 ') Název objektu není platný.

MQRC_OBJECT_NOT_UNIQUE
(2343, X' 927 ') Objekt není jedinečný.

MQRC_OBJECT_Q_MGR_NAME_ERROR
(2153, X'869 ') Název správce front objektů není platný.

MQRC_OBJECT_RECORDS_ERROR
(2155, X'86B') Záznamy objektů nejsou platné.

MQRC_OBJECT_TYPE_ERROR
(2043, X'7FB') Typ objektu není platný.

MQRC_OD_ERROR
(2044, X'7FC') Struktura deskriptoru objektu není platná.

MQRC_OFFSET_ERROR
(2251, X'8CB') Offset segmentu zprávy není platný.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_ORIGINAL_LENGTH_ERROR
(2252, X'8CC') Původní délka není platná.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

MQRC_PAGESET_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_PCF_ERROR
(2149, X'865 ') Struktury PCF nejsou platné.

MQRC_PERSISTENCE_ERROR
(2047, X'7FF') Perzistence není platná.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

MQRC_PMO_ERROR
(2173, X'87D') Struktura voleb vkládání zpráv je neplatná.

MQRC_PMO_RECORD_FLAGS_ERROR
(2158, X'86E') Příznaky záznamu vložení zprávy jsou neplatné.

MQRC_PRIORITY_ERROR
(2050, X'802 ') Priorita zprávy není platná.

MQRC_PUBLICATION_FAILURE
(2502, X'9C6') Publikace nebyla doručena žádnému z odběratelů.

MQRC_PUT_INHIBITED
(2051, X'803 ') Volání Put bylo pro frontu zablokováno.

MQRC_PUT_MSG_RECORDS_ERROR
(2159, X'86F') Záznamy vkládající zprávy jsou neplatné.

MQRC_Q_DELETED
(2052, X'804 ') Fronta byla odstraněna.

MQRC_Q_FULL
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uveden do klidového stavu.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_Q_SPACE_NOT_AVAILABLE
(2056, X'808 ') Není k dispozici místo na disku pro frontu.

MQRC_Q_TYPE_ERROR
(2057, X'809 ') Typ fronty není platný.

MQRC_RECS_PRESENT_ERROR
(2154, X'86A') Počet přítomných záznamů není platný.

MQRC_REMOTE_Q_NAME_ERROR
(2184, X'888 ') Název vzdálené fronty není platný.

MQRC_REPORT_OPTIONS_ERROR
(2061, X'80D') Volby sestavy v deskriptoru zprávy jsou neplatné.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Záznamy odpovědí nejsou platné.

MQRC_RFH_ERROR
(2334, X'91E') Struktura MQRFH nebo MQRFH2 není platná.

MQRC_RMH_ERROR
(2220, X'8AC') Struktura záhlaví referenční zprávy není platná.

Chyba MQRC_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

MQRC_SEGMENT_DÉLKA_NULA
(2253, X'8CD') Délka dat v segmentu zprávy je nula.

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') Existuje možný odběratel pro publikování, ale správce front nemůže zkontrolovat, zda má odeslat publikování odběrateli.

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Volání bylo odmítnuto uživatelskou procedurou pracovní zátěže klastru.

MQRC_STORAGE_CLASS_ERROR
(2105, X'839 ') Chyba paměťové třídy.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí úložné médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

MQRC_SYNCPOINT_LIMIT_DOSAŽENO
(2024, X'7E8') V rámci aktuální pracovní jednotky nelze zpracovat žádné další zprávy.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora synchronizační funkce není k dispozici.

MQRC_TM_ERROR
(2265, X'8D9') Struktura zprávy spouštěče není platná.

MQRC_TMC_ERROR
(2191, X'88F') Struktura zprávy spouštěče znaků je neplatná.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082, X'822 ') Neznámý alias základní fronty.

MQRC_UNKNOWN_DEF_XMIT_Q
(2197, X'895 ') Neznámá výchozí přenosová fronta.

MQRC_UNKNOWN_OBJECT_NAME
(2085, X'825 ') Neznámý název objektu.

MQRC_UNKNOWN_OBJECT_Q_MGR
(2086, X'826 ') Neznámý správce front objektů.

MQRC_UNKNOWN_REMOTE_Q_MGR
(2087, X'827 ') Neznámý vzdálený správce front.

MQRC_UNKNOWN_XMIT_Q
(2196, X'894 ') Neznámá přenosová fronta.

MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Registrace v globální pracovní jednotce se nezdařila.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotky práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

MQRC_WIH_ERROR
(2333, X'91D') Struktura MQWIH není platná.

MQRC_WRONG_CF_LEVEL
(2366, X'93E') Struktura prostředku Coupling Facility má chybnou úroveň.

MQRC_WRONG_MD_VERSION
(2257, X'8D1') Dodána nesprávná verze MQMD.

MQRC_XMIT_Q_TYPE_ERROR
(2091, X'82B') Přenosová fronta není lokální.

MQRC_XMIT_Q_USAGE_ERROR
(2092, X'82C') Přenosová fronta s chybným použitím.

MQRC_XQH_ERROR
(2260, X'8D4') Struktura záhlaví přenosové fronty není platná.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. K vložení zpráv do fronty lze použít volání MQPUT i MQPUT1 . Volání, které má být použito, závisí na okolnostech:

   - Použijte volání MQPUT k umístění více zpráv do *stejné* fronty.

     Nejprve je zadáno volání MQOPEN uvádějící volbu MQOO_OUTPUT následované jedním nebo více požadavky MQPUT na přidání zpráv do fronty; nakonec je fronta uzavřena voláním MQCLOSE. To poskytuje lepší výkon než opakované použití volání MQPUT1 .
   - Pomocí volání MQPUT1 lze do fronty vložit pouze *jednu* zprávu.

Toto volání zapouzdřuje volání MQOPEN, MQPUT a MQCLOSE do jednoho volání a minimalizuje počet volání, která je třeba zadat.

2. Pokud aplikace vloží posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, pokud jsou splněny určité podmínky. Ve většině prostředí však volání MQPUT1 tyto podmínky nesplňuje, a proto nezachovává pořadí zpráv. V těchto prostředích musí být místo toho použito volání MQPUT. Podrobnosti viz Poznámky k použití MQPUT .
3. Volání MQPUT1 lze použít k vložení zpráv do distribučních seznamů. Obecné informace o tomto tématu naleznete v poznámkách k použití pro volání MQOPEN a MQPUT.

   Distribuční seznamy jsou podporovány v následujících prostředích:

   -  AIX
   -  IBM i
   -  Linux
   -  Windows

   a pro klienty IBM MQ připojené k těmto systémům.

   Při použití volání MQPUT1 platí následující rozdíly:

   a. Pokud aplikace poskytuje záznamy odpovědí MQRR, musí být poskytnuty pomocí struktury MQOD; nelze je poskytnout pomocí struktury MQPMO.

   b. Kód příčiny MQRC_OPEN_FAILED není nikdy vrácen produktem MQPUT1 v záznamech odezvy; pokud se nezdaří otevření fronty, záznam odezvy pro tuto frontu obsahuje kód příčiny, který je výsledkem operace otevření.

      Pokud operace otevření pro frontu uspěje s kódem dokončení MQCC_WARNING, kód dokončení a kód příčiny v záznamu odezvy pro tuto frontu jsou nahrazeny kódy dokončení a příčiny, které jsou výsledkem operace vložení.

      Stejně jako u volání MQOPEN a MQPUT nastavuje správce front záznamy odpovědí (jsou-li poskytnuty) pouze v případě, že výsledek volání není stejný pro všechny fronty v distribučním seznamu; to je označeno dokončením volání s kódem příčiny MQRC_MULTIPLE_REASON.
4. Je-li k vložení zprávy do fronty klastru použito volání MQPUT1 , volání se chová, jako by ve volání MQOPEN byla zadána hodnota MQOOO_BIND_NOT_FIXED.
5. Pokud je zpráva vložena s jednou nebo více strukturami záhlaví IBM MQ na začátek dat zprávy aplikace, provede správce front určité kontroly struktur záhlaví, aby ověřil, zda jsou platné. Další informace naleznete v poznámkách k použití pro volání MQPUT.
6. Pokud se vyskytne více než jedna varovná situace (viz parametr **CompCode**), vrácený kód příčiny je první z následujícího seznamu, který platí:

   a. MQRC_MULTIPLE_PŘÍČINA

   b. MQRC_INCOMPLETE_MSG

   c. MQRC_INCOMPLETE_GROUP

   d. MQRC_PRIORITY_PŘEKRAČUJE_MAXIMUM nebo MQRC_UNKNOWN_REPORT_OPTION
7. Pro programovací jazyk Visual Basic platí následující body:

   - Pokud je velikost parametru **Buffer** menší než délka určená parametrem **BufferLength** , volání selže s kódem příčiny MQRC_BUFFER_LENGTH_ERROR.
   - Parametr **Buffer** je deklarován jako typ `String`. Pokud data, která mají být umístěna do fronty, nejsou typu `String`, použijteMQPUT1Any namísto volání MQPUT1.

     Volání MQPUT1Any má stejné parametry jako volání MQPUT1 s tím rozdílem, že parametr **Buffer** je deklarován jako typ `Any`, což umožňuje umístění libovolného typu dat do fronty. To však znamená, že `Buffer` nelze zkontrolovat, aby se zajistilo, že má velikost alespoň `BufferLength` bajtů.

8. Když je vydáno volání MQPUT1 s MQPMO_SYNCPOINT, výchozí chování se změní, aby byla operace vložení dokončena asynchronně. To může způsobit změnu chování některých aplikací, které spoléhají na vracená pole ve strukturách MQOD a MQMD, ale které nyní obsahují nedefinované hodnoty. Aplikace může zadat MQPMO_SYNC_RESPONSE, aby se ujistila, že operace vložení je prováděna synchronně a že jsou dokončeny všechny příslušné hodnoty polí.

Vyvolání jazyka C

```
MQPUT1 (Hconn, &ObjDesc, &MsgDesc, &PutMsgOpts,
       BufferLength, Buffer, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
     CALL 'MQPUT1' USING HCONN, OBJDESC, MSGDESC, PUTMSGOPTS,
                         BUFFERLENGTH, BUFFER, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQPUT1 (Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,
            CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
        CALL MQPUT1,(HCONN,OBJDESC,MSGDESC,PUTMSGOPTS,BUFFERLENGTH,   X
              BUFFER,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Vyvolání jazyka Visual Basic

Windows

```
MQPUT1 Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,
       CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Nastavení atributů objektu

Pomocí volání MQSET můžete změnit atributy objektu reprezentovaného manipulací. Objekt musí být frontou.

Syntaxe

MQSET (*Hconn*, *Hobj*, *SelectorCount*, *Selektory*, *IntAttrCount*, *IntAttrs*, *CharAttrLength*, *CharAttrs*, *Compcode*, *Příčina*)

Parametry

Hconn (připojení)

Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota Hconn byla vrácena předchozím voláním MQCONN nebo MQCONNX.

z/OS V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN

Výchozí manipulátor připojení.

HOBJ
Typ: MQHOBJ-vstup

Tento popisovač představuje objekt fronty s atributy, které mají být nastaveny. Manipulátor byl vrácen předchozím voláním MQOPEN, které určilo volbu MQOO_SET.

SelectorCount
Typ: MQLONG-vstup

Jedná se o počet selektorů, které jsou dodány v poli `Selectors` . Jedná se o počet atributů, které mají být nastaveny. Nula je platná hodnota. Maximální povolený počet je 256.

Selektory.
Typ: MQLONGxSelectorPočet-vstup

Toto je pole selektorů atributů **`SelectorCount`** ; každý selektor identifikuje atribut (celé číslo nebo znak) s hodnotou, která má být nastavena.

Každý selektor musí být platný pro typ fronty, který `Hobj` představuje. Povoleny jsou pouze určité hodnoty MQIA_ * a MQCA_ *, jak je uvedeno níže.

Selektory lze zadat v libovolném pořadí. Hodnoty atributů, které odpovídají selektorům celočíselných atributů (selektory MQIA_ *), musí být zadány v souboru `IntAttrs` ve stejném pořadí, v jakém se tyto selektory vyskytují v souboru `Selectors`. Hodnoty atributů, které odpovídají selektorům znakových atributů (selektory MQCA_ *), musí být uvedeny v souboru `CharAttrs` ve stejném pořadí, v jakém se tyto selektory vyskytují. Selektory MQIA_ * mohou být prokládány se selektory MQCA_ *; důležité je pouze relativní pořadí v rámci jednotlivých typů.

Stejný selektor můžete zadat více než jednou. Pokud tak učiníte, poslední hodnota zadaná pro konkrétní selektor bude ta, která se projeví.

Poznámka:

1. Selektory celočíselných a znakových atributů jsou přiděleny ve dvou různých rozsazích; selektory MQIA_ * jsou umístěny v rozsahu MQIA_FIRST až MQIA_LAST a selektory MQCA_ * v rozsahu MQCA_FIRST až MQCA_LAST.

   Pro každý rozsah konstanty MQIA_LAST_USED a MQCA_LAST_USED definují nejvyšší hodnotu, kterou správce front přijímá.
2. Pokud se nejprve vyskytnou všechny selektory MQIA_ *, lze k adresování odpovídajících prvků v polích `Selectors` a `IntAttrs` použít stejná čísla prvků.
3. Je-li parametr **`SelectorCount`** nulový, na `Selectors` se neodkazuje; v tomto případě může být adresa parametru předaná programy napsané v jazyce C nebo System/390 assembleru null.

Atributy, které lze nastavit, jsou uvedeny v následující tabulce. Pomocí tohoto volání nelze nastavit žádné další atributy. Pro selektory atributů MQCA_ * je konstanta, která definuje délku řetězce vyžadovaného v souboru `CharAttrs` v bajtech, dodána v závorkách.

Tabulka 554. Selektory atributů MQSET pro fronty

Selektor	Popis	Poznámka
MQCA_TRIGGER_DATA	Data spouštěče (MQ_TRIGGER_DATA_LENGTH).	
MQIA_DIST_LISTS	Podpora distribučního seznamu.	1
MQIA_INHIBIT_GET	Zda jsou povoleny operace získání.	
MQIA_INHIBIT_PUT	Zda jsou povoleny operace vložení.	
MQIA_TRIGGER_CONTROL	Ovládací prvek spouštěče.	
MQIA_TRIGGER_HLOUBKA	Hloubka spouątěče.	
MQIA_TRIGGER_MSG_PRIORITY	Priorita zprávy prahové hodnoty pro spouątěče.	

Tabulka 554. Selektory atributů MQSET pro fronty (pokračování)		
Selektor	**Popis**	**Poznámka**
MQIA_TRIGGER_TYPE	Typ spouątěče.	

Poznámka:

1. Podporováno pouze na následujících platformách:
   -  AIX
   -  IBM i
   -  Linux
   -  Windows

   a pro systém IBM MQ MQI clients připojený k těmto systémům.

IntAttrCount
Typ: MQLONG-vstup

Jedná se o počet prvků v poli `IntAttrs` a musí se jednat alespoň o počet selektorů MQIA_ * v parametru **`Selectors`** . Nula je platná hodnota, pokud žádná není.

IntAttrs
Typ: MQLONGxIntAttrCount -vstup

Toto je pole celočíselných hodnot atributu `IntAttrCount` . Tyto hodnoty atributů musí být ve stejném pořadí jako selektory MQIA_ * v poli `Selectors` .

Pokud je parametr **`IntAttrCount`** nebo **`SelectorCount`** nulový, na `IntAttrs` se neodkazuje; v tomto případě může být adresa parametru předaná programy napsané v jazyku C nebo sestavovacím programem System/390 null.

CharAttrDélka
Typ: MQLONG-vstup

Jedná se o délku parametru **`CharAttrs`** v bajtech, která musí být alespoň součtem délek znakových atributů uvedených v poli `Selectors` . Nula je platná hodnota, pokud v souboru `Selectors`nejsou žádné selektory MQCA_ *.

CharAttrs
Typ: MQCHAR x CharAttrDélka-vstup

Jedná se o vyrovnávací paměť obsahující hodnoty znakového atributu, zřetězenou dohromady. Délka vyrovnávací paměti je dána parametrem **`CharAttrLength`** .

Atributy znaků musí být zadány ve stejném pořadí jako selektory MQCA_ * v poli `Selectors` . Délka každého znakového atributu je pevná (viz Selektory). Pokud hodnota, která má být nastavena pro atribut, obsahuje méně neprázdných znaků, než je definovaná délka atributu, vyplní hodnotu v souboru `CharAttrs` napravo mezerami, aby se hodnota atributu shodovala s definovanou délkou atributu.

Pokud je parametr **`CharAttrLength`** nebo **`SelectorCount`** nulový, na `CharAttrs` se neodkazuje; v tomto případě může být adresa parametru předaná programy napsané v jazyku C nebo sestavovacím programem System/390 null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje CompCode.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr CompCode hodnotu MQCC_FAILED, postupujte takto:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Zařízení pro spojku není k dispozici.

MQRC_CF_STRUC_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility se nezdařila.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility je používána.

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347, X'92B') Záhlaví seznamu struktury prostředku Coupling Facility je používáno.

MQRC_CHAR_ATTR_LENGTH_ERROR
(2006, X'7D6') Délka znakových atributů není platná.

MQRC_CHAR_ATTRS_ERROR
(2007, X'7D7') Řetězec znakových atributů není platný.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Není autorizováno pro připojení.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Db2 subsystém není k dispozici.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_HOBJ_ERROR
(2019, X'7E3') popisovač objektu není platný.

MQRC_INHIBIT_VALUE_ERROR
(2020, X'7E4') Hodnota pro atribut fronty inhibit-get nebo inhibit-put není platná.

MQRC_INT_ATTR_COUNT_ERROR
(2021, X'7E5') Počet celočíselných atributů není platný.

MQRC_INT_ATTRS_ARRAY_ERROR
(2023, X'7E7') Pole atributů Integer je neplatné.

MQRC_NOT_OPEN_FOR_SET
(2040, X'7F8') Fronta není otevřena pro nastavení.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu se od doby otevření změnila.

MQRC_OBJECT_POŠKOZENÍ
(2101, X'835 ') Objekt poškozen.

MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

MQRC_Q_DELETED
(2052, X'804 ') Fronta byla odstraněna.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front je neplatný nebo neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_ZASTAVENÍ
(2162, X'872 ') Probíhá ukončování činnosti správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_SELECTOR_COUNT_ERROR
(2065, X'811 ') Počet selektorů není platný.

MQRC_SELECTOR_ERROR
(2067, X'813 ') Selektor atributů není platný.

MQRC_SELECTOR_LIMIT_PŘEKROČENO
(2066, X'812 ') Počet selektorů je příliš velký.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

MQRC_TRIGGER_CONTROL_ERROR
(2075, X'81B') Hodnota pro atribut řízení spouštěče není platná.

MQRC_TRIGGER_DEPTH_ERROR
(2076, X'81C') Hodnota pro atribut hloubky spouštěče není platná.

MQRC_TRIGGER_MSG_PRIORITY_ERR
(2077, X'81D') Hodnota pro atribut trigger-message-priority není platná.

MQRC_TRIGGER_TYPE_ERROR
(2078, X'81E') Hodnota pro atribut typu spouštěče není platná.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Pomocí tohoto volání může aplikace určit pole celočíselných atributů, kolekci řetězců znakových atributů nebo obojí. Pokud se nevyskytují žádné chyby, uvedené atributy jsou nastaveny současně.

Dojde-li k chybě (například pokud selektor není platný nebo dojde-li k pokusu o nastavení atributu na neplatnou hodnotu), volání selže a nejsou nastaveny žádné atributy.

2. Hodnoty atributů lze určit pomocí volání MQINQ; podrobnosti viz “MQINQ-Atributy dotazovaného objektu” na stránce 706 .

   Poznámka: Ne všechny atributy s hodnotami, které lze získat pomocí volání MQINQ, mohou mít své hodnoty změněny pomocí volání MQSET. S tímto voláním nelze například nastavit žádné atributy objektu procesu nebo správce front.
3. Změny atributů jsou zachovány v rámci restartů správce front (kromě změn dočasných dynamických front, které nepřežívají restartování správce front).
4. Atributy modelové fronty nelze změnit pomocí volání MQSET. Pokud však otevřete modelovou frontu pomocí volání MQOPEN s volbou MQOO_SET, můžete pomocí volání MQSET nastavit atributy dynamické lokální fronty vytvořené voláním MQOPEN.
5. Pokud je nastavovaný objekt frontou klastru, musí existovat lokální instance fronty klastru, aby bylo otevření úspěšné.

Další informace o atributech objektu viz:

- “Atributy pro fronty” na stránce 836
- “Atributy pro seznamy názvů” na stránce 869
- “Atributy pro definice procesu” na stránce 871
- “Atributy pro správce front” na stránce 799

Vyvolání jazyka C

```
MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQSET' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,
                   INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,
                   CHARATTRS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
           IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQSET,(HCONN,HOBJ,SELECTORCOUNT,SELECTORS,INTATTRCOUNT,  X
             INTATTRS,CHARATTRLENGTH,CHARATTRS,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Vyvolání jazyka Visual Basic

```
MQSET Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, CompCode, Reason
```

Deklarujte parametry následujícím způsobem:

```
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-Nastavení vlastnosti zprávy

Pomocí volání MQSETMP můžete nastavit nebo upravit vlastnost manipulátoru zprávy.

Syntaxe

MQSETMP (*Hconn*, *Hmsg*, *SetPropOpts*, *Název*, *PropDesc*, *Typ*, *ValueLength*, *Hodnota*, *Compcode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **Hmsg** . Pokud byl manipulátor zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být vytvořeno platné připojení k podprocesu nastavenému na vlastnost manipulátoru zprávy, jinak volání selže s kódem příčiny MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-vstup

Jedná se o popisovač zprávy, který má být upraven. Hodnota byla vrácena předchozím voláním MQCRTMH.

SetPropVolby
Typ: MQSMPO-vstup

Řízení způsobu nastavení vlastností zprávy.

Tato struktura umožňuje aplikacím určit volby, které řídí způsob nastavení vlastností zprávy. Struktura je vstupní parametr pro volání MQSETMP. Další informace viz MQSMPO .

Název
Typ: MQCHARV-vstup

Jedná se o název vlastnosti, která má být nastavena.

Další informace o použití názvů vlastností viz Názvy vlastností a Omezení názvů vlastností .

PropDesc
Typ: MQPD-vstup/výstup

Tato struktura se používá k definování atributů vlastnosti, včetně:

- co se stane, když vlastnost není podporována
- jaký kontext zprávy, do kterého vlastnost patří,
- do jakých zpráv je vlastnost kopírována v rámci toku

Další informace o této struktuře viz MQPD .

Typ
Typ: MQLONG-vstup

Datový typ nastavované vlastnosti. Může se jednat o jednu z následujících možností:

MQTYPE_BOOLEAN
Logická hodnota. *ValueLength* musí být 4.

MQTYPE_BYTE_STRING
Bajtový řetězec. *ValueLength* musí být nula nebo větší.

MQTYPE_INT8
8bitové celé číslo se znaménkem. *ValueLength* musí být 1.

MQTYPE_INT16
16bitové celé číslo se znaménkem. *ValueLength* musí být 2.

MQTYPE_INT32
32bitové celé číslo se znaménkem. *ValueLength* musí být 4.

MQTYPE_INT64
64bitové celé číslo se znaménkem. *ValueLength* musí být 8.

MQTYPE_FLOAT32
32bitové číslo s pohyblivou řádovou čárkou. *ValueLength* musí být 4.

Poznámka: Tento typ není podporován aplikacemi používajícími IBM COBOL for z/OS.

MQTYPE_FLOAT64
64bitové číslo s pohyblivou řádovou čárkou. *ValueLength* musí být 8.

Poznámka: Tento typ není podporován aplikacemi používajícími IBM COBOL for z/OS.

MQTYPE_STRING
Znakový řetězec. *ValueLength* musí být nula nebo větší nebo speciální hodnota MQVL_NULL_TERMINATED.

MQTYPE_NULL
Vlastnost existuje, ale má hodnotu null. *ValueLength* musí být nula.

ValueLength
Typ: MQLONG-vstup

Délka hodnoty vlastnosti v parametru *Value* v bajtech. Nula je platná pouze pro hodnoty null nebo pro řetězce nebo bajtové řetězce. Nula označuje, že vlastnost existuje, ale že hodnota neobsahuje žádné znaky nebo bajty.

Hodnota musí být větší nebo rovna nule nebo následující speciální hodnota, pokud má parametr *Type* nastavenou hodnotu MQTYPE_STRING:

MQVL_NULL_UKONČENO
Hodnota je oddělena první hodnotou null zjištěnou v řetězci. Hodnota null není zahrnuta jako součást řetězce. Tato hodnota je neplatná, pokud není také nastavena hodnota MQTYPE_STRING.

Poznámka: Znak null použitý k ukončení řetězce, je-li nastavena hodnota MQVL_NULL_TERMINATE, má hodnotu null ze znakové sady hodnoty.

Hodnota
Typ: MQBYTExValueDélka-vstup

Hodnota vlastnosti, která má být nastavena. Vyrovnávací paměť musí být zarovnána na hranici odpovídající povaze dat v hodnotě.

V programovacím jazyku C je parametr deklarován jako ukazatel-na-void; jako parametr lze zadat adresu libovolného typu dat.

Je-li *ValueLength* nula, *Hodnota* se neodkazuje. V tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo v sestavovacím modulu System/390 hodnotu null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_WARNING:

MQRC_RFH_FORMAT_ERROR
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nelze analyzovat.

Je-li *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

MQRC_BUFFER_ERROR
(2004, X'07D4') Parametr hodnoty není platný.

MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Parametr délky hodnoty není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_HMSG_ERROR
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Obsluha zprávy je již používána.

MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_PD_ERROR
(2482, X'09B2') Struktura deskriptoru vlastnosti není platná.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Neplatný název vlastnosti.

MQRC_PROPERTY_TYPE_ERROR
(2473, X'09A9') Neplatný datový typ vlastnosti.

MQRC_PROP_NUMBER_FORMAT_ERROR
(2472, X'09A8') Došlo k chybě formátu čísla v datech hodnoty.

MQRC_SMPO_CHYBA
(2463, X'099F') Nastavení struktury vlastností zprávy není platné.

MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQSETMP (Hconn, Hmsg, &SetPropOpts, &Name, &PropDesc, Type,
ValueLength, &Value, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
     CALL 'MQSETMP' USING HCONN, HMSG, SETMSGOPTS, NAME, PROPDESC, TYPE,
                          VALUELENGTH, VALUE, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQSETMP (Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength,
              Value, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
        CALL MQSETMP,(HCONN,HMSG,SETMSGHOPTS,NAME,PROPDESC,TYPE,VALUELENGTH,
              VALUE,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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-Načtení informací o stavu

Pomocí volání MQSTAT načtěte informace o stavu. Typ vrácených informací o stavu je určen hodnotou typu uvedenou ve volání.

Syntaxe

MQSTAT (*Hconn*, *Typ*, *Stat*, *Compcode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

Typ
Typ: MQLONG-vstup

Typ požadovaných informací o stavu. Platné hodnoty > jsou:

MQSTAT_TYPE_ASYNC_ERROR
Vrátit informace o předchozích asynchronních operacích vložení.

MQSTAT_TYPE_RECONNECTION

Vrátit informace o opětovném připojení. Pokud se připojení znovu připojuje nebo se mu nepodařilo znovu připojit, informace popisují selhání, které způsobilo, že se připojení znovu začalo připojovat.

Tato hodnota je platná pouze pro připojení klienta. Pro ostatní typy připojení volání selže s kódem příčiny **MQRC_ENVIRONMENT_ERROR**

MQSTAT_TYPE_RECONNECTION_ERROR

Vrátit informace o předchozím selhání souvisejícím s opětovným připojením. Pokud se připojení nezdařilo znovu připojit, informace popisují selhání, které způsobilo selhání opětovného připojení.

Tato hodnota je platná pouze pro připojení klienta. U ostatních typů připojení volání selže s kódem příčiny **MQRC_ENVIRONMENT_ERROR**.

Statistiky
Typ: MQSTS-vstup/výstup

Struktura informací o stavu. Podrobnosti viz “MQSTS-Struktura vykazování stavu” na stránce 593.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_API_EXIT_ERROR
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_CONNECTION_ZASTAVENÍ
(2203, X'89B') Probíhá ukončování připojení.

MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_Q_MGR_ZASTAVENÍ
(2162,X'872'-Probíhá zastavování správce front.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_STAT_TYPE_ERROR
(2430, X'97E' Chyba typu MQSTAT

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_STS_ERROR
(2426, X'97A') Chyba se strukturou MQSTS

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

1. Volání MQSTAT s určením typu MQSTAT_TYPE_ASYNC_ERROR vrátí informace o předchozích asynchronních operacích MQPUT a MQPUT1 . Struktura MQSTS předaná zpět při návratu z volání MQSTAT obsahuje první zaznamenané informace o asynchronním varování nebo chybě pro toto připojení. Pokud

za prvními následují další chyby nebo varování, obvykle tyto hodnoty nezmění. Pokud však dojde k chybě s kódem dokončení MQCC_WARNING, vrátí se místo toho následné selhání s kódem dokončení MQCC_FAILED .

2. Pokud se od vytvoření připojení nebo od posledního volání funkce MQSTAT nevyskytly žádné chyby, bude ve struktuře MQSTS vrácena hodnota CompCode MQCC_OK a příčina MQRC_NONE .
3. Počet asynchronních volání, která byla zpracována v rámci manipulátoru připojení, je vrácen prostřednictvím tří polí čítače: PutSuccessCount, PutWarningCount a PutFailureCount. Tyto čítače jsou správcem front zvyšovány pokaždé, když je asynchronní operace úspěšně zpracována, mají varování nebo selhávají (všimněte si, že pro účely evidence se počet vložení do distribučního seznamu počítá jednou pro cílovou frontu, nikoli jednou pro distribuční seznam). Počitadlo se nezvyšuje nad maximální kladnou hodnotu AMQ_LONG_MAX.
4. Úspěšné volání MQSTAT má za následek resetování všech předchozích chybových informací nebo počtů.
5. Chování MQSTAT závisí na hodnotě parametru **MQSTAT Type** , který zadáte.
6. **MQSTAT_TYPE_ASYNC_ERROR**
    a. Volání MQSTAT s určením typu MQSTAT_TYPE_ASYNC_ERROR vrátí informace o předchozích asynchronních operacích MQPUT a MQPUT1 . Struktura MQSTS předaná zpět při návratu z volání MQSTAT obsahuje první zaznamenané informace o asynchronním varování nebo chybě pro toto připojení. Pokud za prvními následují další chyby nebo varování, obvykle tyto hodnoty nezmění. Pokud však dojde k chybě s kódem dokončení MQCC_WARNING, vrátí se místo toho následné selhání s kódem dokončení MQCC_FAILED .
    b. Pokud se od vytvoření připojení nebo od posledního volání funkce MQSTAT nevyskytly žádné chyby, bude ve struktuře MQSTS vrácena hodnota CompCode MQCC_OK a příčina MQRC_NONE .
    c. Počet asynchronních volání, která byla zpracována v rámci manipulátoru připojení, je vrácen prostřednictvím tří polí čítače: PutSuccessCount, PutWarningCount a PutFailureCount. Tyto čítače jsou správcem front zvyšovány pokaždé, když je asynchronní operace úspěšně zpracována, mají varování nebo selhávají (všimněte si, že pro účely evidence se počet vložení do distribučního seznamu počítá jednou pro cílovou frontu, nikoli jednou pro distribuční seznam). Počitadlo se nezvyšuje nad maximální kladnou hodnotu AMQ_LONG_MAX.
    d. Úspěšné volání MQSTAT má za následek resetování všech předchozích chybových informací nebo počtů.

MQSTAT_TYPE_RECONNECTION

Předpokládejme, že voláte MQSTAT s volbou Type nastavenou na MQSTAT_TYPE_RECONNECTION uvnitř obslužné rutiny událostí během opětovného připojení. Zvažte tyto příklady.

Klient se pokouší o opětovné připojení nebo se mu nepodařilo znovu připojit.
CompCode ve struktuře MQSTS je MQCC_FAILED a Reason může být buď MQRC_CONNECTION_BROKEN , nebo MQRC_Q_MGR_QUIESCING. ObjectType je MQOT_Q_MGR, ObjectName je název správce front a ObjectQMgrName je prázdné.

Klient úspěšně dokončil opětovné připojení nebo nebyl nikdy odpojen.
CompCode ve struktuře MQSTS je MQCC_OK a Reason je MQRC_NONE

Následná volání MQSTAT vrátí stejné výsledky.

MQSTAT_TYPE_RECONNECTION_ERROR

Předpokládejme, že voláte volání MQSTAT s volbou Type nastavenou na hodnotu MQSTAT_TYPE_RECONNECTION_ERROR v reakci na přijetí MQRC_RECONNECT_FAILED volání MQI. Zvažte tyto příklady.

Při opětovném otevření fronty během opětovného připojení k jinému správci front došlo k selhání autorizace.
CompCode ve struktuře MQSTS je MQCC_FAILED a Reason je příčinou selhání opětovného připojení, například MQRC_NOT_AUTHORIZED. ObjectType je typ objektu, který způsobil problém, například MQOT_QUEUE, ObjectName je název fronty a ObjectQMgrName název správce front, který je vlastníkem fronty.

Během opětovného připojení došlo k chybě připojení soketu.
CompCode ve struktuře MQSTS je MQCC_FAILED a Reason je příčinou selhání opětovného připojení, například MQRC_HOST_NOT_AVAILABLE. ObjectType je MQOT_Q_MGR, ObjectName je název správce front a ObjectQMgrName je prázdné.

Následná volání MQSTAT vrátí stejné výsledky.

Vyvolání jazyka C

```
MQSTAT (Hconn, StatType, &Stat, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN Hconn;        /* Connection Handle */
MQLONG StatType;    /* Status type */
MQSTS Stat;           /* Status information structure */
MQLONG CompCode;    /* Completion code */
MQLONG Reason;        /* Reason code qualifying CompCode */
```

Vyvolání COBOL

```
CALL 'MQSTAT' USING HCONN, STATTYPE, STAT, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**    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.
```

Vyvolání PL/I

```
call MQSTAT (Hconn, StatType, Stat, Compcode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 Vyvolání modulu sestavení

```
CALL MQSTAT,(HCONN,STATTYPE,STAT,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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-Registrace odběru

Prostřednictvím volání MQSUB můžete registrovat odběr aplikací pro konkrétní téma.

Syntaxe

MQSUB (*Hconn*, *SubDesc*, *Hobj*, *Hsub*, *Compcode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

SubDesc
Typ: MQSD-vstup/výstup

Jedná se o strukturu, která identifikuje používaný objekt, který je registrován aplikací. Další informace viz “MQSD-deskriptor odběru” na stránce 569.

HOBJ
Typ: MQHOBJ-vstup/výstup

Tento popisovač představuje přístup, který byl vytvořen pro získání zpráv odeslaných tomuto odběru. Tyto zprávy mohou být buď uloženy ve specifické frontě, nebo může správce front spravovat své úložiště bez použití specifické fronty.

Chcete-li použít specifickou frontu, musíte ji při vytvoření odběru přidružit k odběru. Můžete to provést dvěma způsoby:

- Použitím příkazu DEFINE SUB MQSC a zadáním tohoto příkazu s názvem objektu fronty.
- Poskytnutím tohoto popisovače při volání MQSUB pomocí příkazu MQSO_CREATE

  Je-li tento manipulátor zadán jako vstupní parametr volání, musí se jednat o platný manipulátor objektu vrácený z předchozího volání MQOPEN fronty s použitím alespoň jedné z následujících voleb:

  – MQOO_INPUT_ *
  – MQOO_BROWSE
  – MQOO_OUTPUT (pokud se jedná o vzdálenou frontu)

  Pokud tomu tak není, volání selže s MQRC_HOBJ_ERROR. Nemůže se jednat o popisovač objektu pro alias fronty, která se interpretuje jako objekt tématu. Pokud ano, volání selže s MQRC_HOBJ_ERROR.

Má-li správce front spravovat úložiště zpráv odesílaných do tohoto odběru, měla by být nastavena při vytváření odběru pomocí volby MQSO_MANAGED. Správce front poté vrátí tento manipulátor jako výstupní parametr volání. Vrácený manipulátor je označován jako spravovaný manipulátor. Pokud je zadána hodnota MQHO_NONE, ale není zadána hodnota MQSO_MANAGED, volání se nezdaří s hodnotou MQRC_HOBJ_ERROR.

Pokud vám správce front vrátí spravovaný manipulátor, můžete jej použít pro volání MQGET nebo MQCB s volbami procházení nebo bez nich, pro volání MQINQ nebo pro volání MQCLOSE. Nemůžete

jej použít na MQPUT, MQSUB, MQSET; pokus o jeho použití selže s MQRC_NOT_OPEN_FOR_OUTPUT, MQRC_HOBJ_ERROR nebo MQRC_NOT_OPEN_FOR_SET.

Pokud se tento odběr obnovuje pomocí volby MQSO_RESUME ve struktuře MQSD, lze manipulátor vrátit aplikaci v tomto parametru nastavením parametru MQSO_MANAGED na MQHO_NONE. Můžete to provést bez ohledu na to, zda odběr používá spravovaný popisovač, a může být užitečné poskytnout odběry vytvořené pomocí příkazu DEFINE SUB s popisovačem pro frontu odběrů definovanou v tomto příkazu. V případě obnovení administrativně vytvořeného odběru se fronta otevře s MQOO_INPUT_AS_AS_Q_DEF a MQOO_BROWSE. Potřebujete-li zadat další volby, musí aplikace explicitně otevřít frontu odběrů a poskytnout manipulátor objektu pro volání. Dojde-li k problému při otevírání fronty, volání se nezdaří s hodnotou MQRC_INVALID_DESTINATION. Je-li zadán parametr *Hobj* , musí být ekvivalentem parametru *Hobj* v původním volání MQSUB. To znamená, že pokud je poskytován manipulátor objektu vrácený z volání MQOPEN, musí být manipulátor ve stejné frontě, jako byl dříve použit. Pokud se nejedná o stejnou frontu, volání selže s MQRC_HOBJ_ERROR.

Pokud je tento odběr měněn pomocí volby MQSO_ALTER ve struktuře MQSD, lze zadat jiný parametr *Hobj* . Veškerá publikování, která byla doručena do fronty a byla dříve identifikována prostřednictvím tohoto parametru, zůstávají v této frontě a je zodpovědností aplikace tyto zprávy načíst, pokud parametr **Hobj** nyní představuje jinou frontu.

Tabulka 555. Použití hobj s různými možnostmi odběru

Volby	***Hobj***	**Popis**
MQSO_CREATE + MQSO_MANAGED	Ignorováno na vstupu	Vytvoří odběr s úložištěm zpráv spravovaných správcem front.
MQSO_CREATE	Platný popisovač objektu	Vytvoří odběr poskytující specifickou frontu jako cíl pro zprávy.
MQSO_RESUME	MQHO_NONE	Obnoví dříve vytvořený odběr bez ohledu na to, zda byl spravován či nikoli, a správce front vrátí manipulátor objektu pro použití aplikací.
MQSO_RESUME	Platný, odpovídající, popisovač objektu	Obnoví dříve vytvořený odběr, který používá specifickou frontu jako cíl pro zprávy a používá popisovač objektu se specifickými otevřenými volbami.
MQSO_ALTER + MQSO_MANAGED	MQHO_NONE	Změní existující odběr, který dříve používal specifickou frontu, takže se nyní jedná o spravovaný odběr. Cílovou třídu (spravovanou nebo nespravovanou) nelze změnit.
MQSO_ALTER	Platný popisovač objektu	Změní existující odběr bez ohledu na to, zda byl spravován či nikoli, tak, aby nyní používal specifickou frontu. Není-li použita volba MQSO_MANAGED, lze poskytnutou frontu změnit, ale třídu místa určení (spravovanou či nikoli) nelze změnit.

Bez ohledu na to, zda byl zadán nebo vrácen, musí být při následných voláních MQGET nebo MQCB, která chtějí přijímat publikační zprávy odeslané v rámci tohoto odběru, zadán parametr *Hobj* .

Popisovač *Hobj* již není platný, když je na něm vydáno volání MQCLOSE nebo když se ukončí jednotka zpracování, která definuje rozsah popisovače (dokud se aplikace neodpojí). Rozsah vráceného manipulátoru objektu je stejný jako rozsah manipulátoru připojení určeného ve volání. Informace o rozsahu popisovače viz Hconn (MQHCONN)-výstup . Objekt MQCLOSE manipulátoru *Hobj* neovlivňuje manipulátor *Hsub* .

HSub
Typ: MQHOBJ-výstup

Tento popisovač představuje odběr, který byl proveden. Může být použit pro další dvě operace:

- Lze jej použít při následném volání MQSUBRQ k vyžádání odeslání publikací, když byla při vytváření odběru použita volba MQSO_PUBLICATIONS_ON_REQUEST.
- Lze jej použít při následném volání MQCLOSE k odebrání provedeného odběru. Manipulátor *Hsub* přestane být platný při zadání volání MQCLOSE nebo při ukončení jednotky zpracování, která definuje rozsah manipulátoru. Rozsah vráceného manipulátoru objektu je stejný jako rozsah manipulátoru připojení určeného ve volání. Objekt MQCLOSE manipulátoru *Hsub* neovlivňuje manipulátor *Hobj* .

Tento manipulátor nelze předat volání MQGET nebo MQCB. Musíte použít parametr **Hobj** . Tento popisovač nelze použít pro žádné jiné volání IBM MQ než MQCLOSE nebo MQSUBRQ. Předání tohoto popisovače ostatním výsledkům volání IBM MQ v MQRC_HOBJ_ERROR.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení

MQCC_VAROVÁNÍ
Varování (částečné dokončení)

MQCC_FAILED
Volání selhalo

Příčina
Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK, bude kód příčiny následující:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, je kód příčiny jeden z následujících:

MQRC_CLUSTER_RESOLUTION_ERROR
(2189, X'88D') Nezdařilo se rozpoznání názvu klastru.

MQRC_DURABILITY_NOT_ALLOWED
2436 (X'0984 ') Volání MQSUB s použitím volby MQSO_TRVALÝ se nezdařilo.

MQRC_FUNCTION_NOT_SUPPORTED
2298 (X'08FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

MQRC_HOBJ_ERROR
2019 (X'07E3') popisovač objektu Hobj není platný.

MQRC_IDENTITY_MISMATCH
2434 (X'0982 ') Název odběru odpovídá existujícímu odběru.

MQRC_NOT_AUTHORIZED
2035 (X'07F3') Uživatel nemá oprávnění k provedení operace.

MQRC_NO_SUBSCRIPTION
2428 (X'097C') Zadaný název odběru neexistuje.

MQRC_OBJECT_STRING_ERROR
2441 (X'0989 ') Pole Objectstring není platné.

MQRC_OPTIONS_ERROR
2046 (X'07FE') Parametr nebo pole voleb obsahuje neplatné volby nebo kombinaci neplatných voleb.

MQRC_Q_MGR_QUIESCING
2161 (X'0871 ')-uvedení správce front do klidového stavu.

MQRC_RECONNECT_Q_MGR_REQD
2555 (X'09FB' X) Volba MQCNO_RECONNECT_Q_MGR je povinná.

MQRC_RETAINED_MSG_Q_ERROR
2525 (X'09DD') Zadržená publikování, která existují pro odebíraný řetězec tématu, nelze načíst.

MQRC_RETAINED_NOT_DELIVERED
2526 (X'09DE') Zachovaná publikování, která existují pro odebíraný řetězec tématu, nelze doručit do cílové fronty odběru a nelze je doručit do fronty nedoručených zpráv.

MQRC_SD_ERROR
2424 (X'0978 ') Deskriptor odběru (MQSD) není platný.

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') Řetězec výběru neodpovídá syntaxi selektoru IBM MQ a nebyl k dispozici žádný poskytovatel rozšířeného výběru zpráv.

MQRC_SELECTION_STRING_ERROR
2519 (X'09D7') Řetězec výběru musí být zadán podle popisu v dokumentaci struktury MQCHARV.

MQRC_SELECTOR_SYNTAX_ERROR
2459 (X'099B') Bylo vydáno volání MQOPEN, MQPUT1nebo MQSUB, ale byl zadán řetězec výběru, který obsahoval chybu syntaxe.

MQRC_SUB_USER_DATA_ERROR
2431 (X'097F') SubUserData nejsou platná.

CHYBA MQRC_SUB_NAME_ERROR
Pole 2440 (X'0988 ') SubName není platné.

MQRC_SUB_ALREADY_EXISTS
2432 (X'0980 ') Odběr již existuje.

MQRC_SUB_USER_DATA_ERROR
2431 (X'097F') SubUserData nejsou platná.

MQRC_TOPIC_STRING_ERROR
2425 (X'0979 ') Řetězec tématu je neplatný.

MQRC_UNKNOWN_OBJECT_NAME
2085 (X'0825 ') Objekt identifikovaný v poli MQSD ObjectName nebyl nalezen.

MQRC_SUB_JOIN_NOT_ALTERABLE
29440 (X'7300 ') Režim sdílení předplatného je nekompatibilní s existujícím předplatným. Tato chyba může být vrácena při pokusu o obnovení sdíleného odběru JMS 2.0 v jiné aplikaci než JMS.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

- Odběr je proveden pro téma s názvem buď s použitím krátkého názvu předdefinovaného objektu tématu, úplného názvu řetězce tématu, nebo je vytvořen zřetězením dvou částí. Viz popis položek *ObjectName* a *ObjectString* v souboru “MQSD-deskriptor odběru” na stránce 569.
- Správce front provádí kontroly zabezpečení při zadání volání MQSUB, aby ověřil, zda má identifikátor uživatele, pod kterým je aplikace spuštěna, před povolením přístupu odpovídající úroveň oprávnění. Příslušný objekt tématu je umístěn v hierarchii témat a je provedena kontrola oprávnění pro tento objekt tématu, aby se zajistilo, že je nastaveno oprávnění k odběru. Pokud není použita volba MQSO_MANAGED, provede se kontrola oprávnění v cílové frontě, aby se zajistilo, že je nastaveno

oprávnění pro výstup. Je-li použita volba MQSO_MANAGED, ve spravované frontě se neprovádí žádná kontrola oprávnění pro výstup nebo přístup s dotazem.

- Pokud nezadáte Hobj jako vstup, volání MQSUB přidělí dva manipulátory, manipulátor objektu (Hobj) a manipulátor odběru (Hsub).
- Hobj vrácený při volání MQSUB při použití volby MQSO_MANAGED lze zjistit za účelem zjištění atributů, jako je například prahová hodnota vrácení a název fronty nadměrného vrácení. Můžete se také dotázat na název spravované fronty, ale nesmíte se pokusit tuto frontu přímo otevřít.
- Odběry lze seskupit tak, aby bylo možné skupině odběrů doručit pouze jedno publikování, a to i v případě, že více než jedna ze skupin odpovídá publikování. Odběry jsou seskupeny pomocí volby MQSO_GROUP_SUB a aby bylo možné seskupit odběry, musí být
  - použití stejné pojmenované fronty (která nepoužívá volbu MQSO_MANAGED) ve stejném správci front-reprezentované parametrem Hobj ve volání MQSUB
  - sdílet stejné ID SubCorrel
  - musí být stejné SubLevel

  Tyto atributy definují sadu odběrů považovaných za členy skupiny a jsou také atributy, které nelze změnit, pokud je odběr seskupen. Změna SubLevel má za následek MQRC_SUBLEVEL_NOT_ALTERABLE a změna všech ostatních (které lze změnit, není-li odběr seskupen) má za následek MQRC_GROUPING_NOT_ALTERABLE.
- Úspěšné dokončení volání MQSUB neznamená, že byla akce dokončena. Chcete-li zkontrolovat, zda bylo toto volání dokončeno, prohlédněte si krok DEFINE SUB v části Kontrola, zda byly dokončeny asynchronní příkazy pro distribuované sítě.
- Pole v MQSD jsou vyplněna při návratu z volání MQSUB, které používá volbu MQSO_RESUME. Vrácený MQSD lze předat přímo do volání MQSUB, které používá volbu MQSO_ALTER s jakýmikoli změnami, které je třeba provést v odběru použitém na MQSD. Některá pole mají zvláštní aspekty, jak je uvedeno v tabulce.

Tabulka 556. Speciální aspekty pro pole v modulu MQSD

Název pole v MQSD	**Speciální aspekty.**
Volby přístupu nebo vytvoření	Některé z voleb lze při návratu z volání MQSUB resetovat. Pokud poté znovu použijete MQSD ve volání MQSUB, volba, kterou požadujete, musí být explicitně nastavena.
Volby trvanlivosti, Volby místa určení, Volby registrace a možnosti zástupného znaku	Tyto volby jsou nastaveny podle potřeby
Volby publikování	Tyto volby jsou nastaveny podle potřeby, s výjimkou volby MQSO_NEW_PUBLICATIONS_ONLY, která je použitelná pouze pro MQSO_CREATE.
Další volby	Tyto volby se při návratu z volání MQSUB nezměnily. Řídí, jak je volání rozhraní API vydáváno a není uloženo s odběrem. Musí být nastaveny podle potřeby pro každé následné volání MQSUB, které znovu použije MQSD.
ObjectName	Toto pole pouze pro vstup je při návratu z volání MQSUB nezměněno.
ObjectString	Toto pole pouze pro vstup je při návratu z volání MQSUB nezměněno. Použitý úplný název tématu je vrácen v poli *ResObjectString* , pokud je poskytnuta vyrovnávací paměť.

Tabulka 556. Speciální aspekty pro pole v modulu MQSD (pokračování)	
Název pole v MQSD	**Speciální aspekty.**
AlternateUserID a AlternateSecurityID	Tato vstupní pole jsou při návratu z volání MQSUB beze změny. Řídí, jak je volání rozhraní API vydáváno a není uloženo s odběrem. Musí být nastaveny podle potřeby pro každé následné volání MQSUB, které znovu použije MQSD.
SubExpiry	Při návratu z volání MQSUB pomocí volby MQSO_RESUME je toto pole nastaveno na původní vypršení platnosti odběru a nikoli na zbývající dobu vypršení platnosti. Pokud poté znovu použijete MQSD ve volání MQSUB s použitím volby MQSO_ALTER, resetujte vypršení platnosti odběru a začněte znovu odpočítávat.
SubName	Toto pole je vstupní pole pro volání MQSUB a ve výstupu není změněno.
SubUserData a SelectionString	Tato pole s proměnnou délkou jsou vrácena při výstupu volání MQSUB s použitím volby MQSO_RESUME, pokud je zadána vyrovnávací paměť, a také s kladnou délkou vyrovnávací paměti v souboru *VSBufSize*. Není-li poskytnuta žádná vyrovnávací paměť, je vrácena pouze délka v poli *VSLength* MQCHARV. Pokud je poskytnutá vyrovnávací paměť menší než prostor požadovaný pro vrácení pole, vrátí se v poskytnuté vyrovnávací paměti pouze *VSBufSize* bajtů. Pokud poté znovu použijete MQSD ve volání MQSUB s použitím volby MQSO_ALTER a není poskytnuta vyrovnávací paměť, ale je zadána nenulová hodnota *VSLength* , pokud tato délka odpovídá existující délce pole, nedojde k žádné změně pole.
SubCorrelID a token PubAccounting	Pokud nepoužíváte MQSO_SET_CORREL_ID, je *SubCorrelId* generován správcem front. Pokud nepoužíváte MQSO_SET_IDENTITY_CONTEXT, je *PubAccountingToken* generován správcem front. Tato pole jsou vrácena v modulu MQSD z volání MQSUB s použitím volby MQSO_RESUME. Pokud jsou generovány správcem front, je vygenerovaná hodnota vrácena při volání MQSUB pomocí volby MQSO_CREATE nebo MQSO_ALTER.
PubPriority, SubLevel & PubApplIdentityData	Tato pole jsou vrácena v modulu MQSD.
Řetězec ResObject	Toto pole pouze pro výstup je vráceno v MQSD, pokud je poskytnuta vyrovnávací paměť.

Vyvolání jazyka C

```
MQSUB (Hconn, &SubDesc, &Hobj, &Hsub, &CompCode, &Reason)
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQSUB' USING HCONN, SUBDESC, HOBJ, HSUB, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   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.
```

Vyvolání PL/I

```
call MQSUB (Hconn, SubDesc, Hobj, Hsub, CompCode, Reason)
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQSUB,(HCONN,SUBDESC,HOBJ,HSUB,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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-Požadavek na odběr

Použijte volání MQSUBRQ k vytvoření požadavku na zachované publikování, když byl odběratel registrován s MQSO_PUBLICATIONS_ON_REQUEST.

Syntaxe

MQSUBRQ (*Hconn*, *Hsub*, *Action*, *SubRqOpts*, *Compcode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *Hconn*zadat následující hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

HSub
Typ: MQHOBJ-vstup

Tento popisovač představuje odběr, pro který má být požadována aktualizace. Hodnota *Hsub* byla vrácena z předchozího volání MQSUB.

Akce
Typ: MQLONG-vstup

Tento parametr řídí konkrétní akci, která je požadována pro odběr. Musí být zadána následující hodnota:

MQSR_ACTION_PUBLIKACE

Tato akce požaduje, aby se pro uvedené téma odeslalo publikování aktualizace.
Lze jej použít pouze v případě, že odběratel při provádění odběru zadal volbu MQSO_PUBLICATIONS_ON_REQUEST pro volání MQSUB. Pokud má správce front zachované publikování pro dané téma, bude odesláno odběrateli. Pokud ne, volání selže. Pokud je aplikaci odesláno publikování, které bylo zachováno, je to označeno vlastností zprávy MQIsRetained tohoto publikování.

Vzhledem k tomu, že téma v existujícím odběru reprezentovaném parametrem Hsub může obsahovat zástupné znaky, může odběratel obdržet více zachovaných publikování.

SubRqOpty
Typ: MQSRO-vstup/výstup

Tyto volby řídí akci MQSUBRQ, podrobnosti viz “MQSRO-Volby požadavku na odběr” na stránce 591 .

Nejsou-li vyžadovány žádné volby, programy napsané v jazyku C nebo sestavovacím modulu S/390 mohou místo určení adresy struktury MQSRO zadat adresu parametru null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; jedná se o jeden z následujících:

MQCC_OK
Úspěšné dokončení

MQCC_VAROVÁNÍ
Varování (částečné dokončení)

MQCC_FAILED
Volání selhalo

Příčina
Typ: MQLONG-výstup

Kód příčiny, který kvalifikuje *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_FUNCTION_NOT_SUPPORTED
2298 (X'08FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

MQRC_NO_RETAINED_MSG
2437 (X'0985 ') Pro toto téma nejsou aktuálně uložena žádná zachovaná publikování.

MQRC_OPTIONS_ERROR
2046 (X'07FE') Parametr nebo pole voleb obsahuje neplatné volby nebo kombinaci neplatných voleb.

MQRC_Q_MGR_QUIESCING
2161 (X'0871 ')-uvedení správce front do klidového stavu.

MQRC_SRO_ERROR
2438 (X'0986 ') Při volání MQSUBRQ nejsou volby požadavku na odběr MQSRO platné.

MQRC_RETAINED_MSG_Q_ERROR
2525 (X'09DD') Zadržená publikování, která existují pro odebíraný řetězec tématu, nelze načíst.

MQRC_RETAINED_NOT_DELIVERED
2526 (X'09DE') Zachovaná publikování, která existují pro odebíraný řetězec tématu, nelze doručit do cílové fronty odběru a nelze je doručit do fronty nedoručených zpráv.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Poznámky k použití

Pro použití kódu akce MQSR_ACTION_PUBLICATION platí následující poznámky k použití:

1. Pokud je toto příkazové slovo úspěšně dokončeno, zachovaná publikování odpovídající zadanému odběru byla odeslána k odběru a lze je přijmout pomocí příkazu MQGET nebo MQCB s použitím příkazu Hobj vráceného pro původní příkazové slovo MQSUB, které vytvořilo odběr.
2. Pokud téma odebírané původním příkazovým slovem MQSUB, které vytvořilo odběr, obsahovalo zástupný znak, lze odeslat více než jedno zachované publikování. Počet publikování odeslaných jako výsledek tohoto volání je zaznamenán v poli NumPubs ve struktuře optů SubRq.
3. Pokud je toto příkazové slovo dokončeno s kódem příčiny MQRC_NO_RETAINED_MSG, nebyla pro uvedené téma aktuálně zachována žádná publikování. #
4. Pokud je toto příkazové slovo dokončeno s kódem příčiny MQRC_RETAINED_MSG_Q_ERROR nebo MQRC_RETAINED_NOT_DELIVERED, jsou pro uvedené téma aktuálně zachována publikování, ale došlo k chybě, která znamenala, že nebylo možné je doručit.
5. Aplikace musí mít aktuální odběr tématu, aby mohla toto volání provést. Pokud byl odběr proveden v předchozí instanci aplikace a není k dispozici platný manipulátor pro odběr, musí aplikace nejprve volat MQSUB s volbou MQSO_RESUME, aby získala manipulátor pro použití v tomto volání.
6. Publikování jsou odesílána do místa určení, které je registrováno pro použití s aktuálním odběrem této aplikace. Pokud musí být publikování odeslána někde jinde, je třeba nejprve změnit odběr pomocí volání MQSUB s volbou MQSO_ALTER.

Vyvolání jazyka C

```
MQSUB (Hconn, Hsub, Action, &SubRqOpts, &CompCode, &Reason)
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání COBOL

```
CALL 'MQSUBRQ' USING HCONN, HSUB, ACTION, SUBRQOPTS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
** 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.
```

Vyvolání PL/I

```
call MQSUBRQ (Hconn, Hsub, Action, SubRqOpts, CompCode, Reason)
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Vyvolání High Level Assembler

```
CALL MQSUBRQ,(HCONN, HSUB, ACTION, SUBRQOPTS,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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
```

Atributy objektů

V této kolekci témat jsou uvedeny pouze ty objekty IBM MQ , které mohou být předmětem volání funkce MQINQ, a jsou uvedeny podrobnosti o atributech, které lze dotazovat, a o selektorech, které mají být použity.

Atributy pro správce front

Některé atributy správce front jsou opraveny pro konkrétní implementace; jiné lze změnit pomocí příkazu MQSC ALTER QMGR.

Atributy lze také zobrazit pomocí příkazu DISPLAY QMGR. Většinu atributů správce front lze získat otevřením speciálního objektu MQOT_Q_MGR a použitím volání MQINQ s vráceným popisovačem.

Následující tabulka shrnuje atributy, které jsou specifické pro správce front. Atributy jsou popsány v abecedním pořadí.

Poznámka: Názvy atributů zobrazené v této sekci jsou popisné názvy použité s voláním MQINQ; názvy jsou stejné jako pro příkazy PCF. Při použití příkazů MQSC k definování, změně nebo zobrazení atributů jsou použity alternativní krátké názvy. Další informace naleznete v tématu Příkazy MQSC .

Tabulka 557. Atributy pro správce front

Atribut	Popis
AccountingConnOverride	Potlačit nastavení evidence.
AccountingInterval	Jak často zapisovat mezilehlé účetní záznamy.
ActivityConnOverride	Potlačit nastavení aktivity.
ActivityTrace	Řídí kolekci trasování aktivity aplikace IBM MQ MQI.
AdoptNewMCACheck	Prvky zkontrolovány, aby se určilo, zda se má adoptovat nový MCA.
AdoptNewMCAType	Určuje, zda se má automaticky restartovat osiřelá instance MCA určitého typu kanálu.
AlterationDate	Datum, kdy byla definice naposledy změněna
AlterationTime	Čas, kdy byla definice naposledy změněna
AuthorityEvent	Řídí, zda jsou generovány události autorizace (bez autorizace)
BridgeEvent	Řídicí atribut pro události mostu.
ChannelAutoDef	Určuje, zda je povolena automatická definice kanálu.
ChannelAutoDefEvent	Řídí, zda jsou generovány události automatické definice kanálu
ChannelAutoDefExit	Název uživatelské procedury pro automatickou definici kanálu
ChannelEvent	Řídicí atribut pro události kanálu.
ChannelInitiatorControl	Atribut řízení pro inicializátor kanálu
ChannelMonitoring	Online monitorovací data pro kanály
ChannelStatistics	Řídí shromažďování statistických dat pro kanály.
ChinitAdapters	Počet dílčích úloh adaptéru pro zpracování volání IBM MQ .
ChinitDispatchers	Počet dispečerů, které mají být použity pro inicializátor kanálu.
	Vyhrazeno pro použití v systému IBM .
ChinitTraceAutoStart	Určuje, zda se má trasování inicializátoru kanálu spouštět automaticky.
ChinitTraceTableSize	Velikost prostoru dat trasování inicializátoru kanálu.
ClusterSenderMonitoringDefault	Výchozí nastavení dat monitorování online pro odesílací kanály klastru
ClusterSenderOdesílatel klastru	Řídí shromažďování statistických monitorovacích informací pro odesílací kanály klastru.
ClusterWorkloadData	Uživatelská data pro uživatelskou proceduru pracovní zátěže klastru
ClusterWorkloadExit	Název uživatelské procedury pro správu pracovní zátěže klastru
ClusterWorkloadLength	Maximální délka dat zprávy předaných uživatelské proceduře pracovní zátěže klastru
CLWLMRUChannels	Počet naposledy použitých kanálů pro vyrovnávání pracovní zátěže klastru
CLWLUseQ	Pracovní zátěž klastru používá vzdálenou frontu.
CodedCharSetId	Identifikátor znakové sady
CommandEvent	Řídicí atribut pro události příkazu.
CommandInputAtribut QName	Název fronty vstupu příkazů
CommandLevel	Úroveň příkazů
CommandServerŘídicí atribut	Řídicí atribut pro příkazový server.
Atribut Událost konfigurace	Řídicí atribut pro události konfigurace.
DeadLetterQName	Název fronty nedoručených zpráv

Tabulka 557. Atributy pro správce front (pokračování)	
Atribut	**Popis**
DefClusterXmitQueue	Výchozí typ přenosové fronty klastru
DefXmitQName	Výchozí název přenosové fronty
DistLists	Podpora seznamu distribuce
DNSGroup	Název skupiny pro modul listener TCP při použití podpory služby Dynamic Domain Name Services správce pracovní zátěže.
DNSWLM	Zda se modul listener protokolu TCP registruje ve správci pracovní zátěže pro služby dynamického názvu domény.
ExpiryInterval	Interval mezi skenováním zpráv s vypršenou platností
IGQPutAuthority	Oprávnění k zařazování do fronty v rámci skupiny
IGQUserId	Identifikátor uživatele řazení do fronty v rámci skupiny
InhibitEvent	Řídí, zda jsou generovány události blokování (Inhibit Get a Inhibit Put)
V 9.3.0 V 9.3.0 InitialKey 1	Počáteční klíč pro systém ochrany heslem.
IPAddressVersion	Verze adresy Internet Protocol
IntraGroupqueuing	Podpora řazení do front v rámci skupiny
ListenerTimer	Časový interval mezi pokusy o restart modulu listener po selhání APPC nebo TCP/IP.
LocalEvent	Řídí, zda jsou generovány lokální chybové události
LoggerEvent	Řídí, zda jsou generovány události modulu protokolování
LUGroupName	Generické jméno LU pro modul listener 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front.
LUName	Název LU, která má být použita pro odchozí přenosy LU 6.2 .
LU62ARMSuffix	Přípona SYS1.PARMLIB člen APPCPMxx, který nominuje LUADD pro tento inicializátor kanálu.
LU62Channels	Maximální počet aktuálních kanálů nebo připojených klientů, kteří používají LU 6.2.
MaxActiveChannels	Maximální počet kanálů, které mohou být kdykoli aktivní.
MaxChannels	Maximální počet aktuálních kanálů.
MaxHandles	Maximální počet popisovačů
MaxMsgLength	Maximální délka zprávy v bajtech
MaxPriority	Maximální priorita
MaxPropertiesLength	Maximální délka dat vlastností v bajtech
MaxUncommittedMsgs	Maximální počet nepotvrzených zpráv v rámci pracovní jednotky
MQIAccounting	Řídí shromažďování účetních informací pro data MQI.
MQIStatistics	Řídí shromažďování statistických monitorovacích informací pro správce front.
MsgMarkBrowseInterval	Interval, po jehož uplynutí může správce front odebrat značku z procházených zpráv.
OutboundPortMin	S volbou *OutboundPortMin*definuje rozsah čísel portů, která se mají použít při vázání odchozích kanálů.
OutboundPortMin	S volbou *OutboundPortMax*definuje rozsah čísel portů, která se mají použít při vázání odchozích kanálů.
PerformanceEvent	Řídí, zda jsou generovány události související s výkonem
Platforma	Platforma, na které je spuštěn správce front
PubSubNPInputMsg	Zda zrušit (nebo uchovat) nedoručenou vstupní zprávu
PubSubOdezva NPResponse	Řídí chování nedoručených
PubSubMaxMsgRetryCount	Počet opakování při zpracování (pod synchronizačním bodem) zprávy o nezdařených příkazech
PubSubSyncPoint	Zda mají být pod bodem synchronizace zpracovány pouze trvalé zprávy (nebo všechny).
PubSubMode	Zda je rozhraní publikování/odběru ve frontě spuštěno
QMgrDesc	Popis správce front
QMgrIdentifier	Jedinečný interně generovaný identifikátor správce front
QMgrName	Název správce front

Tabulka 557. Atributy pro správce front (pokračování)	
Atribut	**Popis**
QSGName	Název skupiny sdílení front
QueueAccounting	Řídí shromažďování informací o monitorování účtů pro fronty.
QueueMonitoring	Data monitorování online pro fronty
QueueStatistics	Řídí shromažďování statistických dat pro fronty.
ReceiveTimeout	Jak dlouho kanál TCP/IP čeká na data před návratem do neaktivního stavu.
ReceiveTimeoutMin	Kvalifikátor pro *ReceiveTimeout*.
ReceiveTimeoutType	Minimální doba, po kterou kanál TCP/IP čeká na data, než se vrátí do neaktivního stavu.
RemoteEvent	Řídí, zda jsou generovány vzdálené chybové události
RepositoryName	Název klastru, pro který tento správce front poskytuje služby úložiště
RepositoryNamelist	Název objektu seznamu názvů obsahujícího názvy klastrů, pro které tento správce front poskytuje služby úložiště
ScyCase	Případ bezpečnostních profilů
SharedQMgrmodulu SharedQMgr	Název sdíleného správce front
“SPLCAP” na stránce 833	IBM MQ Rozšířená ochrana zabezpečení zpráv pro správce front je zapnuta nebo vypnuta.
SSLCRLNamelist 1	Název objektu seznamu názvů obsahujícího názvy objektů ověřovacích informací.
SSLCryptoHardware 1	Konfigurační řetězec kryptografického hardwaru.
SSLEvent	Atribut řízení pro události TLS.
SSLFIPSRequired	Pro šifrování používejte pouze algoritmy s certifikací FIPS.
SSLKeyRepository 1	Umístění úložiště klíčů TLS.
V 9.3.0 V 9.3.0 SSLKeyRepositoryHeslo 1	Heslo pro úložiště klíčů TLS.
SSLKeyResetPočet	Počet resetů klíče TLS.
SSLTasks 1	Počet dílčích úloh serveru pro zpracování volání TLS.
StatisticsInterval	Jak často se zapisují data monitorování statistiky.
StartStopEvent	Řídí, zda jsou generovány události spuštění a zastavení
SyncPoint	Dostupnost synchronizačního bodu
TCPChannels	Maximální počet aktuálních kanálů nebo připojených klientů, kteří používají protokol TCP/IP.
TCPKeepAlive	Zda se má použít TCP KEEPALIVE pro kontrolu jiného konce připojení.
TCPName	Název systému TCP/IP, který používáte.
TCPStackType	Jak může inicializátor kanálu používat adresy TCP/IP.
TraceRouteAtribut záznamu	Řídí záznam informací o trasovací trase.
TriggerInterval	Interval zpráv spouštěče
verze	Verze
XrCapability	Určuje, zda jsou podporovány příkazy telemetrie.

Notes:

1. Tento atribut nelze zjišťovat pomocí volání MQINQ a není popsán v této sekci. Podrobnosti o tomto atributu naleznete v části Změnit správce front .

Související úlohy

Určení, že za běhu jsou v klientu MQI použity pouze specifikace CipherSpecs s certifikací FIPS.

Související odkazy

Standard FIPS (Federal Information Processing Standards) pro AIX, Linux, and Windows

AccountingConnPotlačení (MQLONG)

To umožňuje aplikacím přepsat nastavení hodnot ACCTMQI a ACCTQDATA v atributu Qmgr.

Hodnota je jedna z následujících:

MQMON_DISABLED
Aplikace nemohou přepsat nastavení atributů ACCTMQI a ACCTQ Qmgr pomocí pole Volby ve struktuře MQCNO ve volání MQCONNX. Toto je výchozí hodnota.

MQMON_ENABLED
Aplikace mohou přepsat atributy ACCTQ a ACCTMQI Qmgr pomocí pole Volby ve struktuře MQCNO.

Změny této hodnoty se projeví pouze u připojení ke správci front po změně atributu.

Tento atribut je podporován pouze na následujících platformách:

- 
IBM i
- 
AIX and Linux
- 
Windows

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACCOUNTING_CONN_OVERRIDE s voláním MQINQ.

AccountingInterval (MQLONG)

Určuje, jak dlouho před zápisem přechodných evidenčních záznamů (v sekundách).

Hodnota je celé číslo v rozsahu 0 až 604800 s výchozí hodnotou 1800 (30 minut). Chcete-li vypnout přechodné záznamy, zadejte hodnotu 0.

Tento atribut je podporován pouze na následujících platformách:

- 
IBM i
- 
AIX and Linux
- 
Linux
- 
Windows

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACCOUNTING_INTERVAL s voláním MQINQ.

ActivityConnPřepsat (MQLONG)

To umožňuje aplikacím potlačit nastavení hodnoty ACTVTRC v atributu správce front.

Hodnota je jedna z následujících:

MQMON_DISABLED
Aplikace nemohou přepsat nastavení atributu správce front ACTVTRC pomocí pole Volby ve struktuře MQCNO ve volání MQCONNX. Toto je výchozí hodnota.

MQMON_ENABLED
Aplikace mohou přepsat atribut správce front ACTVTRC pomocí pole Volby ve struktuře MQCNO.

Změny této hodnoty se projeví pouze u připojení ke správci front po změně atributu.

Tento atribut je podporován pouze na systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACTIVITY_CONN_OVERRIDE s voláním `MQINQ` .

ActivityTrace (MQLONG)

Tato volba řídí kolekci trasování aktivity aplikace IBM MQ MQI.

Hodnota je jedna z následujících:

MQMON_ON
Shromážděte trasování aktivity aplikace IBM MQ MQI.

MQMON_OFF
Neshromažďujte trasování aktivity aplikace IBM MQ MQI. Toto je výchozí hodnota.

Nastavíte-li atribut správce front ACTVCONO na hodnotu ENABLED, může být tato hodnota přepsána pro jednotlivá připojení pomocí pole Volby ve struktuře MQCNO.

Změny této hodnoty se projeví pouze u připojení ke správci front po změně atributu.

Tento atribut je podporován pouze na systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACTIVITY_TRACE s voláním MQINQ .

AdoptNewMCACheck (MQLONG)

Definuje prvky pro kontrolu, zda má být převzat agent MCA při zjištění nového příchozího kanálu, který má stejný název jako již aktivní agent MCA.

Hodnota je jedna z následujících:

MQADOPT_CHECK_Q_MGR_NAME
Zkontrolujte název správce front.

MQADOPT_CHECK_NET_ADDR
Zkontrolujte síťovou adresu.

MQADOPT_CHECK_ALL
Zkontrolujte název správce front a síťovou adresu. Pokud je to možné, proveďte tuto kontrolu, abyste ochránili své kanály před tím, aby byly vypnuty, neúmyslně nebo úmyslně. Toto je výchozí hodnota.

MQADOPT_CHECK_NONE
Nezaškrtávejte žádné prvky.

Změny tohoto atributu se projeví při příštím pokusu kanálu o převzetí kanálu.

z/OS Tento atribut je podporován pouze na systému z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ADOPTNEWMCA_CHECK s voláním MQINQ.

AdoptNewMCAType (MQLONG)

Tato volba určuje, zda má být automaticky restartována osiřelá instance agenta MCA konkrétního typu kanálu při zjištění nového příchozího požadavku kanálu odpovídajícího atributu AdoptNewMCACheck.

Jedná se o jednu z následujících hodnot:

MQADOPT_TYPE_NO
Adoptování osiřelých instancí kanálu není vyžadováno. Toto je výchozí hodnota.

MQADOPT_TYPE_ALL
Adoptovat všechny typy kanálů.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ADOPTNEWMCA_TYPE s voláním MQINQ.

AlterationDate (MQCHAR12)

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby se délka 12 bajtů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_DATE s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_DATE_LENGTH.

AlterationTime (MQCHAR8)

Toto je čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_TIME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_TIME_LENGTH.

AuthorityEvent (MQLONG)

Tato volba určuje, zda jsou generovány události autorizace (bez oprávnění). Jedná se o jednu z následujících hodnot:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_AUTHORITY_EVENT s voláním MQINQ.

BridgeEvent (MQLONG)

Tato volba určuje, zda jsou generovány události mostu IMS .

Hodnota je jedna z následujících:

MQEVR_ENABLED
Generujte události mostu IMS následujícím způsobem:

MQRC_BRIDGE_STARTED
MQRC_BRIDGE_ZASTAVENO

MQEVR_DISABLED
Negenerujte události mostu IMS ; toto je výchozí hodnota.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_BRIDGE_EVENT s voláním MQINQ.

ChannelAutoDef (MQLONG)

Tento atribut řídí automatickou definici kanálů typu MQCHT_RECEIVER a MQCHT_SVRCONN. Automatická definice kanálů MQCHT_CLUSSDR je vždy povolena. Hodnota je jedna z následujících:

MQCHAD_DISABLED
Automatická definice kanálu je zakázána.

MQCHAD_ENABLED
Automatická definice kanálu je povolena.

Multi Tento atribut je podporován pouze na systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHANNEL_AUTO_DEF s voláním MQINQ.

ChannelAutoDefEvent (MQLONG)

Tato volba určuje, zda mají být generovány události automatické definice kanálu. Týká se kanálů typu MQCHT_RECEIVER, MQCHT_SVRCONN a MQCHT_CLUSSDR. Hodnota je jedna z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Multi Tento atribut je podporován pouze na systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHANNEL_AUTO_DEF_EVENT s voláním MQINQ.

ChannelAutoDefExit (MQCHARn)

Jedná se o název uživatelské procedury pro automatickou definici kanálu. Pokud je tento název neprázdný a parametr *ChannelAutoDef* má hodnotu MQCHAD_ENABLED, je uživatelská procedura volána pokaždé, když se správce front chystá vytvořit definici kanálu. Toto platí pro kanály typu MQCHT_RECEIVER, MQCHT_SVRCONN a MQCHT_CLUSSDR. Ukončení pak může provést jednu z následujících možností:

- Vytvořte definici kanálu beze změny.
- Upravte atributy vytvořené definice kanálu.
- Zcela potlačit vytvoření kanálu.

Poznámka: Délka i hodnota tohoto atributu jsou specifické pro prostředí. Podrobnosti o hodnotě tohoto atributu v různých prostředích naleznete v úvodu struktury MQCD v části "MQCD-Definice kanálu" na stránce 1471 .

z/OS V systému z/OSse tento atribut používá pouze pro odesílací a přijímací kanály klastru.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CHANNEL_AUTO_DEF_EXIT s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_EXIT_NAME_LENGTH.

ChannelEvent (MQLONG)

Tato volba určuje, zda mají být generovány události kanálu.

Jedná se o jednu z následujících hodnot:

VÝJIMKA-MQEVR_EXCEPTION
Generovat pouze následující události kanálu:

- MQRC_CHANNEL_ACTIVATED
- MQRC_CHANNEL_CONV_ERROR
- MQRC_CHANNEL_NOT_ACTIVATED
- MQRC_CHANNEL_STOPPED s následujícím ReasonQualifiers:

  MQRQ_CHANNEL_STOPPED_ERROR
  MQRQ_CHANNEL_STOPPED_RETRY
  MQRQ_CHANNEL_STOPPED_DISABLED

  MQRC_CHANNEL_STOPPED_BY_USER

MQEVR_ENABLED
Generovat všechny události kanálu. To znamená, že kromě událostí generovaných pomocí EXCEPTION, vygenerujte následující události kanálu:

- MQRC_CHANNEL_STARTED
- MQRC_CHANNEL_STOPPED s následujícím ReasonQualifier:

  MQRQ_CHANNEL_STOPPED_OK

MQEVR_DISABLED
Negenerovat události kanálu; toto je výchozí hodnota.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHANNEL_EVENT s voláním MQINQ.

Řízení ChannelInitiator(MQLONG)

Tato volba určuje, zda má být inicializátor kanálu spuštěn při spuštění správce front.

Jedná se o jednu z následujících hodnot:

MQSVC_CONTROL_MANUAL
Inicializátor kanálu nemá být spuštěn automaticky.

MQSVC_CONTROL_Q_MGR
Inicializátor kanálu se spustí automaticky při spuštění správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_CONTROL s voláním MQINQ.

ChannelMonitoring (MQLONG)

Tento atribut uvádí data monitorování online pro kanály.

Hodnota je jedna z následujících:

MQMON_NONE
Zakažte shromažďování dat pro monitorování kanálu pro všechny kanály bez ohledu na nastavení atributu kanálu MONCHL. Toto je výchozí hodnota.

MQMON_OFF
Vypněte shromažďování dat monitorování pro kanály, které uvádějí QMGR v atributu kanálu MONCHL.

MQMON_LOW
Zapněte shromažďování dat monitorování s nízkým poměrem shromažďování dat pro kanály uvádějící QMGR v atributu kanálu MONCHL.

MQMON_MEDIUM
Zapněte shromažďování dat monitorování se středním poměrem shromažďování dat pro kanály uvádějící QMGR v atributu kanálu MONCHL.

MQMON_HIGH-vysoká
Zapněte shromažďování dat monitorování s vysokým poměrem shromažďování dat pro kanály uvádějící QMGR v atributu kanálu MONCHL.

z/OS Na systémech z/OS povolení tohoto parametru jednoduše zapne shromažďování statistických dat bez ohledu na vybranou hodnotu. Zadáním `LOW`, `MEDIUM` nebo `HIGH` nezpůsobíte ve výsledcích žádný změnu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MONITORING_CHANNEL s voláním MQINQ.

ChannelStatistics (MQLONG)

Tato volba řídí shromažďování statistických dat pro kanály.

Hodnota je jedna z následujících:

MQMON_NONE
Zakažte shromažďování dat pro statistiku kanálu pro všechny kanály bez ohledu na nastavení atributu kanálu STATCHL. Toto je výchozí hodnota.

MQMON_OFF
Vypněte shromažďování statistických dat pro kanály, které uvádějí QMGR v atributu kanálu STATCHL.

MQMON_LOW
Zapněte shromažďování statistických dat s nízkým poměrem shromažďování dat pro kanály, které uvádějí QMGR v atributu kanálu STATCHL.

MQMON_MEDIUM
Zapněte shromažďování statistických dat se středním poměrem shromažďování dat pro kanály určující QMGR v atributu kanálu STATCHL.

MQMON_HIGH-vysoká
Zapněte shromažďování statistických dat s vysokým poměrem shromažďování dat pro kanály, které určují QMGR v atributu kanálu STATCHL.

Pro většinu systémů se doporučuje používat MEDIUM. Avšak pro kanál, který zpracovává vysoký objem zpráv každou sekundu, můžete snížit úroveň vzorkování výběrem hodnoty LOW. Pro kanál, který

zpracovává pouze několik zpráv a pro který jsou důležité nejaktuálnější informace, můžete také vybrat hodnotu HIGH.

z/OS Na systémech z/OS povolení tohoto parametru jednoduše zapne shromažďování statistických dat bez ohledu na vybranou hodnotu. Zadáním `LOW`, `MEDIUM` nebo `HIGH` nezpůsobíte ve výsledcích žádný změnu.Tento parametr musí být povolen, aby bylo možné shromažďovat účtovací záznamy kanálu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_STATISTICS_CHANNEL s voláním MQINQ.

ChinitAdapters (MQLONG)

Jedná se o počet podúloh adaptéru, které se mají použít ke zpracování volání IBM MQ . Hodnota musí být 0-9999, s výchozí hodnotou 8.

Poměr adaptérů k dispečerům (atribut ChinitDispatchers) by měl být přibližně 8 až 5. Máte-li však pouze několik kanálů, nemusíte hodnotu tohoto parametru snižovat z výchozí hodnoty. Můžete použít následující hodnoty: pro testovací systém, 8 (výchozí); pro produkční systém, 20. V ideálním případě byste měli mít 20 adaptérů, které poskytují větší paralelizmus volání IBM MQ . To je důležité pro trvalé zprávy. Pro přechodné zprávy může být lepší méně adaptérů.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_ADAPTERS s voláním MQINQ.

ChinitDispatchers (MQLONG)

Jedná se o počet dispečerů, které mají být použity pro inicializátor kanálu. Hodnota musí být 0-9999, s výchozí hodnotou 5.

Jako vodítko povolte jeden dispečer pro 50 aktuálních kanálů. Máte-li však pouze několik kanálů, nemusíte hodnotu tohoto atributu snižovat z výchozí hodnoty. Používáte-li protokol TCP/IP, je největší počet dispečerů, kteří jsou používáni pro kanály TCP/IP, 100, a to i v případě, že zde zadáte větší hodnotu. Můžete použít následující nastavení: testovací systémy, 5 (výchozí); produkční systémy, 20 (k obsluze až 1000 aktivních kanálů potřebujete 20 dispečerů).

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_DISPATCHERS s voláním MQINQ.

ChinitTraceAutoStart (MQLONG)

Tato volba určuje, zda se má trasování inicializátoru kanálu spouštět automaticky.

Hodnota je jedna z následujících:

MQTRAXSTR_YES
Spustit trasování inicializátoru kanálu automaticky. Toto je výchozí hodnota.

MQTRAXSTR_NO
Nespouštět trasování inicializátoru kanálu automaticky.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_TRACE_AUTO_START s voláním MQINQ.

ChinitTraceTableSize (MQLONG)

Jedná se o velikost prostoru dat trasování inicializátoru kanálu (v MB).

Hodnota musí být v rozsahu 0 až 2048 s výchozí hodnotou 2.

Poznámka: Kdykoli použijete velké datové prostory z/OS , ujistěte se, že máte v systému dostatek pomocné paměti pro podporu jakékoli související aktivity stránkování z/OS . Pravděpodobně bude potřeba také zvýšit velikost datových sad SYS1.DUMP.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_TRACE_TABLE_SIZE s voláním MQINQ.

ClusterSenderMonitoringDefault (MQLONG)

Tato volba určuje hodnotu, která má být nahrazena atributem ChannelMonitoring automaticky definovaných odesílacích kanálů klastru.

Hodnota je jedna z následujících:

MQMON_Q_MGR
Kolekce online dat monitorování je zděděna z nastavení atributu **ChannelMonitoring** správce front. Toto je výchozí hodnota.

MQMON_OFF
Monitorování kanálu je zakázáno.

MQMON_LOW
Pokud *ChannelMonitoring* není MQMON_NONE, monitorování je povoleno s nízkou rychlostí shromažďování dat s minimálním dopadem na výkon systému. Shromážděná data pravděpodobně nebudou nejaktuálnější.

MQMON_MEDIUM
Pokud *ChannelMonitoring* není MQMON_NONE, monitorování je povoleno se střední rychlostí shromažďování dat s omezeným dopadem na výkon systému.

MQMON_HIGH-vysoká
Pokud *ChannelMonitoring* není MQMON_NONE, monitorování je povoleno s vysokou rychlostí shromažďování dat s pravděpodobným dopadem na výkon systému. Shromážděná data jsou nejaktuálnější dostupná.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MONITORING_AUTO_CLUSSDR s voláním MQINQ.

Statistika ClusterSender(MQLONG)

Vzhledem k tomu, že odesílací kanály klastru lze automaticky definovat z definice CLUSRCVR v úložišti, nelze změnit nastavení atributu STATCHL pro tyto automaticky definované odesílací kanály klastru pomocí příkazu ALTER channel. Pro tyto kanály je rozhodnutí, zda shromažďovat data monitorování online, založeno na nastavení tohoto atributu správce front.

Hodnota je jedna z následujících:

MQMON_Q_MGR
Shromažďování statistických dat pro automaticky definované odesílací kanály klastru je založeno na hodnotě atributu STATCHL správce front. Toto je výchozí hodnota.

MQMON_OFF
Vypněte shromažďování statistických dat pro automaticky definované odesílací kanály klastru.

MQMON_LOW
Povolte shromažďování statistických dat pro automaticky definované odesílací kanály klastru s nízkým poměrem shromažďování dat.

MQMON_MEDIUM
Povolte shromažďování statistických dat pro automaticky definované odesílací kanály klastru se středním poměrem shromažďování dat.

MQMON_HIGH-vysoká
Povolte shromažďování statistických dat pro automaticky definované odesílací kanály klastru s vysokým poměrem shromažďování dat.

Pro většinu systémů doporučujeme MEDIUM. V případě automaticky definovaného odesílacího kanálu klastru, který zpracovává vysoký objem zpráv každou sekundu, však můžete snížit úroveň vzorkování výběrem volby LOW. Pro kanál, který zpracovává pouze několik zpráv a pro který jsou důležité nejaktuálnější informace, můžete také vybrat hodnotu HIGH.

z/OS Na systémech z/OS povolení tohoto parametru jednoduše zapne shromažďování statistických dat bez ohledu na vybranou hodnotu. Zadáním `LOW`, `MEDIUM` nebo `HIGH` nezpůsobíte ve výsledcích žádný změnu.Tento parametr musí být povolen, aby bylo možné shromažďovat účtovací záznamy kanálu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_STATISTICS_AUTO_CLUSSDR s voláním MQINQ.

ClusterWorkloadData (MQCHAR32)

Jedná se o uživatelem definovaný 32bajtový znakový řetězec, který je při volání předán uživatelské proceduře pracovní zátěže klastru. Pokud neexistují žádná data, která by bylo možné předat uživatelské proceduře, řetězec je prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CLUSTER_WORKLOAD_DATA s voláním MQINQ.

ClusterWorkloadKonec (MQCHARn)

Jedná se o název uživatelské procedury pro správu pracovní zátěže klastru. Není-li tento název prázdný, je uživatelská procedura volána pokaždé, když je zpráva vložena do fronty klastru nebo přesunuta z jedné odesílací fronty klastru do jiné. Uživatelská procedura pak může buď přijmout instanci fronty vybranou správcem front jako místo určení pro zprávu, nebo vybrat jinou instanci fronty.

Poznámka: Délka i hodnota tohoto atributu jsou specifické pro prostředí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CLUSTER_WORKLOAD_EXIT s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_EXIT_NAME_LENGTH.

ClusterWorkloadDélka (MQLONG)

Jedná se o maximální délku dat zprávy, která jsou předána uživatelské proceduře pracovní zátěže klastru. Skutečná délka dat předaných do ukončení je minimální z následujících hodnot:

- Délka zprávy.
- Atribut **`MaxMsgLength`** správce front.
- Atribut **`ClusterWorkloadLength`** .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLUSTER_WORKLOAD_LENGTH s voláním MQINQ.

CLWLMRUChannels (MQLONG)

Tato volba určuje maximální počet naposledy použitých kanálů klastru, které mají být brány v úvahu pro použití algoritmem výběru pracovní zátěže klastru.

Jedná se o hodnotu v rozsahu 1 až 999999999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLWL_MRU_CHANNELS s voláním MQINQ.

CLWLUseQ (MQLONG)

Tato volba určuje, zda mají být pro pracovní zátěž klastru použity vzdálené fronty.

Hodnota je jedna z následujících:

MQCLWL_USEQ_ANY
Použijte lokální i vzdálené fronty.

MQCLWL_USEQ_LOCAL
Nepoužívat vzdálené fronty. Toto je výchozí hodnota.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLWL_USEQ s voláním MQINQ.

CodedCharSetId (MQLONG)

Tato volba definuje znakovou sadu používanou správcem front pro všechna pole řetězce znaků definovaná v rozhraní MQI, například názvy objektů a datum a čas vytvoření fronty. Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech objektů. Nevztahuje se na data aplikace přenášené ve zprávě. Hodnota závisí na prostředí:

- V systému z/OSje hodnota nastavena z parametrů systému při spuštění správce front; výchozí hodnota je 500.
- V systému Windowsje hodnotou primární CODEPAGE uživatele, který vytváří správce front.
- V systému IBM ije hodnota nastavena v prostředí při prvním vytvoření správce front.
- V systému AIX and Linuxje výchozí hodnota CODESET pro národní prostředí uživatele, který vytváří správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CODED_CHAR_SET_ID s voláním MQINQ.

CommandEvent (MQLONG)

Tato volba určuje, zda mají být generovány události příkazu, a to následujícím způsobem:

MQEVR_DISABLED
Negenerovat události příkazu. Toto nastavení je výchozí.

MQEVR_ENABLED
Generovat události příkazu.

MQEVR_NO_DISPLAY
Události příkazu jsou generovány pro všechny úspěšné příkazy jiné než MQINQ.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_COMMAND_EVENT s voláním MQINQ.

CommandInputQName (MQCHAR48)

Jedná se o název vstupní fronty příkazů definované v lokálním správci front. Jedná se o frontu, do které mohou uživatelé odesílat příkazy, pokud k tomu mají oprávnění. Název fronty závisí na prostředí:

- V systému z/OSje název fronty SYSTEM.COMMAND.INPUT; příkazy MQSC a PCF lze odeslat. Podrobné informace o příkazech MQSC a definicích programovatelných formátů příkazů naleznete v tématu Příkazy MQSC .
- Ve všech ostatních prostředích je název fronty SYSTEM.ADMIN.COMMAND.QUEUEa lze do ní odesílat pouze příkazy PCF. Příkaz MQSC však může být odeslán do této fronty, pokud je příkaz MQSC uzavřen v příkazu PCF typu MQCMD_ESCAPE. Informace o příkazu Escape viz Escape .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_COMMAND_INPUT_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

CommandLevel (MQLONG)

Poznámka: Podpora operačního systému HP-UX pro všechny komponenty systému IBM MQ , včetně serveru a klientů, byla v produktu IBM MQ 9.1odebrána.

To označuje úroveň řídicích příkazů systému podporovaných správcem front. Může se jednat o jednu z následujících hodnot:

MQCMDL_LEVEL_800
Úroveň 800 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 8.0
- IBM MQ for IBM i 8.0
- IBM MQ for Linux 8.0

- IBM MQ for Windows 8.0
- IBM MQ for z/OS 8.0

MQCMDL_LEVEL_801
Úroveň 801 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 8.0.0 Fix Pack 2
- IBM MQ for HP-UX 8.0.0 Fix Pack 2
- IBM MQ for IBM i 8.0.0 Fix Pack 2
- IBM MQ for Linux 8.0.0 Fix Pack 2

MQCMDL_LEVEL_802
Úroveň 802 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 8.0.0 Fix Pack 3
- IBM MQ for IBM i 8.0.0 Fix Pack 3
- IBM MQ for Linux 8.0.0 Fix Pack 3
- IBM MQ for Windows 8.0.0 Fix Pack 3

MQCMDL_LEVEL_900
Úroveň 900 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.0
- IBM MQ for IBM i 9.0
- IBM MQ for Linux 9.0
- IBM MQ for Windows 9.0
- IBM MQ for z/OS 9.0

MQCMDL_LEVEL_901
Úroveň 901 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for Linux 9.0.1
- IBM MQ for Windows 9.0.1
- IBM MQ for z/OS 9.0.1

MQCMDL_LEVEL_902
Úroveň 902 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for Linux 9.0.2
- IBM MQ for Windows 9.0.2
- IBM MQ for z/OS 9.0.2

MQCMDL_LEVEL_903
Úroveň 903 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for Linux 9.0.3
- IBM MQ for Windows 9.0.3
- IBM MQ for z/OS 9.0.3

MQCMDL_LEVEL_904
Úroveň 904 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.0.4
- IBM MQ for Linux 9.0.4
- IBM MQ for Windows 9.0.4
- IBM MQ for z/OS 9.0.4

MQCMDL_LEVEL_905
Úroveň 905 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.0.5
- IBM MQ for Linux 9.0.5
- IBM MQ for Windows 9.0.5
- IBM MQ for z/OS 9.0.5

MQCMDL_LEVEL_910
Úroveň 910 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1
- IBM MQ for IBM i 9.1
- IBM MQ for Linux 9.1
- IBM MQ for Windows 9.1
- IBM MQ for z/OS 9.1

MQCMDL_LEVEL_911
Úroveň 911 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1.1
- IBM MQ for Linux 9.1.1
- IBM MQ for Windows 9.1.1
- IBM MQ for z/OS 9.1.1

MQCMDL_LEVEL_912
Úroveň 912 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1.2
- IBM MQ for Linux 9.1.2
- IBM MQ for Windows 9.1.2
- IBM MQ for z/OS 9.1.2

MQCMDL_LEVEL_913
Úroveň 913 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1.3
- IBM MQ for Linux 9.1.3
- IBM MQ for Windows 9.1.3
- IBM MQ for z/OS 9.1.3

MQCMDL_LEVEL_914
Úroveň 914 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1.4
- IBM MQ for Linux 9.1.4
- IBM MQ for Windows 9.1.4
- IBM MQ for z/OS 9.1.4

MQCMDL_LEVEL_915
Úroveň 915 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1.5
- IBM MQ for Linux 9.1.5
- IBM MQ for Windows 9.1.5
- IBM MQ for z/OS 9.1.5

MQCMDL_LEVEL_910
Úroveň 910 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1
- IBM MQ for IBM i 9.1
- IBM MQ for Linux 9.1
- IBM MQ for Windows 9.1
- IBM MQ for z/OS 9.1

MQCMDL_LEVEL_920
Úroveň 920 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2
- IBM MQ for IBM i 9.2
- IBM MQ for Linux 9.2
- IBM MQ for Windows 9.2
- IBM MQ for z/OS 9.2

MQCMDL_LEVEL_921
Úroveň 921 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.1
- IBM MQ for Linux 9.2.1
- IBM MQ for Windows 9.2.1
- IBM MQ for z/OS 9.2.1

MQCMDL_LEVEL_922
Úroveň 922 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.2
- IBM MQ for Linux 9.2.2
- IBM MQ for Windows 9.2.2

- IBM MQ for z/OS 9.2.2

MQCMDL_LEVEL_923
Úroveň 923 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.3
- IBM MQ for Linux 9.2.3
- IBM MQ for Windows 9.2.3
- IBM MQ for z/OS 9.2.3

MQCMDL_LEVEL_924
Úroveň 924 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.4
- IBM MQ for Linux 9.2.4
- IBM MQ for Windows 9.2.4
- IBM MQ for z/OS 9.2.4

MQCMDL_LEVEL_925
Úroveň 925 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.5
- IBM MQ for Linux 9.2.5
- IBM MQ for Windows 9.2.5
- IBM MQ for z/OS 9.2.5

MQCMDL_LEVEL_930
Úroveň 930 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.3
- IBM MQ for IBM i 9.3
- IBM MQ for Linux 9.3
- IBM MQ for Windows 9.3
- IBM MQ for z/OS 9.3

MQCMDL_LEVEL_931
Úroveň 931 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.3.1
- IBM MQ for Linux 9.3.1
- IBM MQ for Windows 9.3.1
- IBM MQ for z/OS 9.3.1

MQCMDL_LEVEL_932
Úroveň 932 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.3.2
- IBM MQ for Linux 9.3.2
- IBM MQ for Windows 9.3.2

- IBM MQ for z/OS 9.3.2

Sada řídicích příkazů systému, která odpovídá konkrétní hodnotě atributu **CommandLevel** , se liší v závislosti na hodnotě atributu **Platform** ; obojí musí být použito k rozhodnutí, které řídicí příkazy systému jsou podporovány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_COMMAND_LEVEL s voláním funkce MQINQ .

Řízení CommandServer(MQLONG)

Určuje, zda má být příkazový server spuštěn při spuštění správce front.

Hodnota může být některá z následujících:

MQSVC_CONTROL_MANUAL
Příkazový server se nemá spouštět automaticky.

MQSVC_CONTROL_Q_MGR
Příkazový server se spustí automaticky při spuštění správce front.

Tento atribut není v systému z/OSpodporován.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CMD_SERVER_CONTROL s voláním MQINQ.

ConfigurationEvent (MQLONG)

Řídí, zda jsou generovány události konfigurace.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CONFIGURATION_EVENT s voláním MQINQ.

Hodnota může být některá z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Multi *Velikost CurrentQFile(MQLONG)*

Aktuální velikost souboru fronty v megabajtech zaokrouhlená nahoru na nejbližší megabajt.

Tabulka 558. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Hodnota tohoto atributu stavu fronty je bez ohledu na velikost fronty, která je momentálně ve frontě, zaokrouhlená nahoru na nejbližší megabajt. Pro novou frontu s výchozími atributy je hodnota **CurrentQFileSize** 1.

Maximální hodnota tohoto atributu je 99,999,9999 MB a pro tento atribut neexistuje žádná výchozí hodnota.

Multi *CurrentMaxQFileSize (MQLONG)*

Aktuální maximální velikost, na kterou může soubor fronty růst, zaokrouhlená nahoru na nejbližší megabajt, vzhledem k aktuální velikosti bloku používané ve frontě.

Tabulka 559. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Použití tohoto pole je dvojí:

- Pokud nastavíte **MaxQFileSize** na výchozí hodnotu pro aktuální velikost bloku, **CurrentMaxQFileSize** zobrazí skutečnou hodnotu, se kterou se výchozí hodnota rovná.
- Pokud **CurrentMaxQFileSize** neodpovídá **MaxQFileSize**, víte, že fronta musí být vyprázdněna, aby bylo možné převzít větší granularitu.

Poznámka: Další informace o změně velikosti souborů fronty a velikosti bloku a granularity naleznete v tématu Úprava souborů fronty IBM MQ .

Maximální hodnota tohoto atributu je 99,999,9999 MB a neexistuje žádná výchozí hodnota. Hodnota je bez ohledu na aktuálně nastavenou maximální hodnotu; pro novou frontu s výchozími atributy je hodnota **CurrentMaxQFileSize** 2,088,960 MB.

Název fronty DeadLetter(MQCHAR48)

Jedná se o název fronty definované v lokálním správci front jako fronta nedoručených zpráv. Zprávy jsou odesílány do této fronty, pokud je nelze směrovat na správné místo určení.

Zprávy jsou například vloženy do této fronty, když:

- Zpráva dorazí do správce front určeného pro frontu, která dosud není v daném správci front definována.
- Zpráva dorazí do správce front, ale fronta, pro kterou je určena, ji nemůže přijmout, protože je možné, že:
  - Fronta je plná
  - Požadavky na vložení jsou zablokovány
  - Odesílající uzel nemá oprávnění vkládat zprávy do fronty.

Aplikace mohou také vkládat zprávy do fronty nedoručených zpráv.

Se zprávami sestavy se zachází stejně jako s běžnými zprávami; pokud zprávu sestavy nelze doručit do cílové fronty (obvykle do fronty určené polem *ReplyToQ* v deskriptoru původní zprávy), zpráva sestavy se umístí do fronty nedoručených zpráv (nedoručených zpráv).

Poznámka: Zprávy, které překročily svůj čas vypršení platnosti (viz pole MQMD-Expiry). **nejsou** přeneseny do této fronty, když jsou vyřazeny. Zpráva sestavy vypršení platnosti (MQRO_EXPIRATION) je však i nadále generována a odeslána do fronty *ReplyToQ* , pokud to vyžaduje odesílající aplikace.

Zprávy nejsou vloženy do fronty nedoručených zpráv (nedoručených zpráv), pokud aplikace, která vydala požadavek na vložení, byla na problém upozorněna synchronně pomocí kódu příčiny vráceného voláním MQPUT nebo MQPUT1 (například zpráva vložená do lokální fronty, pro kterou jsou požadavky na vložení blokována).

Zprávy ve frontě nedoručených zpráv mají někdy svá data zpráv aplikace s předponou se strukturou MQDLH. Tato struktura obsahuje další informace, které označují, proč byla zpráva umístěna do fronty nedoručených zpráv (nedoručených zpráv). Další podrobnosti o této struktuře viz “MQDLH-Záhlaví nedoručených zpráv” na stránce 354 .

Tato fronta musí být lokální frontou s atributem **Usage** nastaveným na hodnotu MQUS_NORMAL.

Pokud správce front nepodporuje frontu nedoručených zpráv (nedoručených zpráv) nebo pokud nebyla definována, je název prázdný. Všichni správci front IBM MQ podporují frontu nedoručených zpráv (nedoručených zpráv), ale standardně není definována.

Není-li fronta nedoručených zpráv (nedoručených zpráv) definována, plná nebo nepoužitelná z jiného důvodu, je zpráva, která by do ní byla přenesena agentem kanálu zpráv, uchována v přenosové frontě.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_DEAD_LETTER_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

DefClusterXmitQueue(MQLONG)

Atribut `DefClusterXmitQueue` řídí, která přenosová fronta je standardně vybrána odesílacími kanály klastru pro získání zpráv, pro odeslání zpráv přijímacím kanálům klastru.

Hodnoty **DefClusterXmitQueueType** jsou MQCLXQ_SCTQ nebo MQCLXQ_CHANNEL.

MQCLXQ_SCTQ

Všechny odesílací kanály klastru odesílají zprávy z produktu SYSTEM.CLUSTER.TRANSMIT.QUEUE. correlID zpráv uvedený v přenosové frontě identifikuje, pro který odesílací kanál klastru je zpráva určena.

Parametr SCTQ je nastaven při definování správce front.

MQCLXQ_CHANNEL

Každý odesílací kanál klastru posílá zprávy z různých přenosových front. Každá přenosová fronta je vytvořena jako trvalá dynamická fronta z modelové fronty SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE.

Pokud je atribut správce front DefClusterXmitQueueTypenastaven na hodnotu CHANNEL, Výchozí konfigurace se změnila na odesílací kanály klastru přidružené k jednotlivým přenosovým frontám klastru. Přenosové fronty jsou trvalé dynamické fronty vytvořené z modelové fronty SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE. Každá přenosová fronta je přidružená k jednomu odesílacímu kanálu klastru. Protože přenosovou frontu klastru obsluhuje jeden odesílací kanál klastru, obsahuje přenosová fronta zprávy pouze pro jednoho správce front v jednom klastru. Klastry můžete nakonfigurovat tak, aby každý správce front z klastru obsahoval pouze jednu frontu klastru. V takovém případě se zprávy ze správce front budou do každé fronty klastru přenášet odděleně od zpráv do jiných front.

Chcete-li zadat dotaz na hodnotu, zavolejte funkci MQINQnebo odešlete příkaz PCF správce front Inquire (MQCMD_INQUIRE_Q_MGR), nastavte selektor MQIA_DEF_CLUSTER_XMIT_Q_TYPE . Chcete-li změnit hodnotu, odešlete příkaz Změnit správce front (MQCMD_CHANGE_Q_MGR) PCF a nastavte selektor MQIA_DEF_CLUSTER_XMIT_Q_TYPE .

Související odkazy

Změnit správce front

Zjistit správce front

“MQINQ-Atributy dotazovaného objektu” na stránce 706

Volání MQINQ vrátí pole celých čísel a sadu znakových řetězců obsahujících atributy objektu.

DefXmitNázev fronty (MQCHAR48)

Jedná se o název přenosové fronty, která se používá pro přenos zpráv do vzdálených správců front, pokud neexistuje jiná indikace, kterou přenosovou frontu použít.

Pokud neexistuje žádná výchozí přenosová fronta, název je zcela prázdný. Počáteční hodnota tohoto atributu je prázdná.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_DEF_XMIT_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

DistLists (MQLONG)

Označuje, zda lokální správce front podporuje distribuční seznamy ve voláních MQPUT a MQPUT1 . Jedná se o jednu z následujících hodnot:

MQDL_PODPOROVÁNO
Distribuční seznamy jsou podporovány.

MQDL_NOT_SUPPORTED
Distribuční seznamy nejsou podporovány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DIST_LISTS s voláním MQINQ.

DNSGroup (MQCHAR18)

Tento parametr není již používán. Viz Co se změnilo v produktu IBM MQ 8.0.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_DNS_GROUP s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_DNS_GROUP_NAME_LENGTH.

DNSWLM (MQLONG)

Tento parametr není již používán. Viz Co se změnilo v produktu IBM MQ 8.0.

Hodnota je jedna z následujících:

MQDNSWLM_YES
Tato hodnota se může zobrazit u správce front migrovaného ze starší verze. Hodnota je ignorována.

MQDNSWLM_NO
Jedná se o jedinou hodnotu podporovanou správcem front.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DNS_WLM s voláním MQINQ.

ExpiryInterval (MQLONG)

To označuje frekvenci, s jakou správce front prochází fronty, které hledají zprávy s vypršenou platností. Jedná se buď o časový interval v sekundách v rozsahu 1 až 99 999 999, nebo o následující speciální hodnotu:

MQEXPI_OFF
Správce front nevyhledává ve frontách zprávy s vypršenou platností.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_EXPIRY_INTERVAL s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

IGQPutAuthority (MQLONG)

Tento atribut se používá pouze v případě, že je lokální správce front členem skupiny sdílení front. Označuje typ kontroly oprávnění, která se provádí, když lokální agent front v rámci skupiny (agent IGQ) odebere zprávu ze sdílené přenosové fronty a umístí zprávu do lokální fronty. Hodnota je jedna z následujících:

VÝCHOZÍ-MQIGQPA_DEFAULT
Identifikátor uživatele kontrolovaný pro autorizaci je hodnota pole *UserIdentifier* v *samostatném* deskriptoru MQMD, který je přidružen ke zprávě, když je zpráva ve sdílené přenosové frontě. Jedná se o identifikátor uživatele programu, který umístil zprávu do sdílené přenosové fronty, a je obvykle stejný jako identifikátor uživatele, pod kterým je spuštěn vzdálený správce front.

Pokud profil RESLEVEL označuje, že se má zkontrolovat více než jeden identifikátor uživatele, zkontroluje se také identifikátor uživatele lokálního agenta IGQ (*IGQUserId*).

MQIGQPA_CONTEXT
Identifikátor uživatele kontrolovaný pro autorizaci je hodnota pole *UserIdentifier* v *samostatném* deskriptoru MQMD, který je přidružen ke zprávě, když je zpráva ve sdílené přenosové frontě. Jedná se o identifikátor uživatele programu, který umístil zprávu do sdílené přenosové fronty, a je obvykle stejný jako identifikátor uživatele, pod kterým je spuštěn vzdálený správce front.

Pokud profil RESLEVEL označuje, že se má zkontrolovat více než jeden identifikátor uživatele, zkontroluje se také identifikátor uživatele lokálního agenta IGQ (*IGQUserId*) a hodnota pole *UserIdentifier* ve *vloženém* MQMD. Druhý identifikátor uživatele je obvykle identifikátor uživatele aplikace, která je původcem zprávy.

MQIGQPA_ONLY_IGQ
Identifikátor uživatele kontrolovaný pro autorizaci je identifikátor uživatele lokálního agenta IGQ (*IGQUserId*).

Pokud profil RESLEVEL označuje, že se má zkontrolovat více než jeden identifikátor uživatele, tento identifikátor uživatele se použije pro všechny kontroly.

MQIGQPA_ALTERNATE_OR_IGQ
Identifikátor uživatele kontrolovaný pro autorizaci je identifikátor uživatele lokálního agenta IGQ (*IGQUserId*).

Pokud profil RESLEVEL označuje, že se má zkontrolovat více než jeden identifikátor uživatele, zkontroluje se také hodnota pole *UserIdentifier* ve *vloženém* deskriptoru MQMD. Tento identifikátor uživatele je obvykle identifikátor uživatele aplikace, která zprávu vyvolala.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_IGQ_PUT_AUTHORITY s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

IGQUserId (MQLONG)

Tento atribut lze použít pouze v případě, že je lokální správce front členem skupiny sdílení front. Určuje identifikátor uživatele, který je přidružen k lokálnímu agentu front v rámci skupiny (agent IGQ). Tento identifikátor je jedním z identifikátorů uživatelů, které lze zkontrolovat pro autorizaci, když agent IGQ vkládá zprávy do lokálních front. Skutečné kontrolované identifikátory uživatelů závisí na nastavení atributu **IGQPutAuthority** a na volbách externího zabezpečení.

Je-li parametr *IGQUserId* prázdný, k agentu IGQ není přidružen žádný identifikátor uživatele a odpovídající kontrola autorizace se neprovede (ačkoli ostatní identifikátory uživatelů mohou být stále kontrolovány pro autorizaci).

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_IGQ_USER_ID s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_USER_ID_LENGTH.

z/OS Tento atribut je podporován pouze na systému z/OS.

InhibitEvent (MQLONG)

Tato volba určuje, zda mají být generovány události blokování (Inhibit Get a Inhibit Put). Hodnota je jedna z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor událostí MQIA_INHIBITORY s voláním MQINQ.

V systému z/OSnemůžete použít volání MQINQ k určení hodnoty tohoto atributu.

IntraGroupqueuing (MQLONG)

Tento atribut se používá pouze v případě, že je lokální správce front členem skupiny sdílení front. Označuje, zda je pro skupinu sdílení front povoleno řazení do front v rámci skupiny. Hodnota je jedna z následujících:

MQIGQ_DISABLED
Všechny zprávy určené pro ostatní správce front ve skupině sdílení front jsou přenášeny pomocí konvenčních kanálů.

MQIGQ_ENABLED
Zprávy určené pro ostatní správce front ve skupině sdílení front jsou přenášeny pomocí sdílené přenosové fronty, pokud je splněna následující podmínka:

- Délka dat zprávy plus záhlaví přenosu nepřesahuje 63 kB (64 512 bajtů).

  Doporučuje se, aby bylo pro záhlaví přenosu přiděleno o něco více místa, než je velikost MQXQH; pro tento účel je poskytnuta konstanta MQ_MSG_HEADER_LENGTH.

Není-li tato podmínka splněna, je zpráva přenášena pomocí konvenčních kanálů.

Poznámka: Je-li povoleno řazení do front v rámci skupiny, pořadí zpráv přenášených prostřednictvím sdílené přenosové fronty není zachováno vzhledem k těm, které jsou přenášeny konvenčními kanály.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_INTRA_GROUP_queeing s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

IPAddressVersion (MQLONG)

Uvádí, která verze adresy IP, buď IPv4 , nebo IPv6, se použije.

Tento atribut je relevantní pouze pro systémy, na kterých běží produkt IPv4 i IPv6 , a ovlivňuje pouze kanály definované jako mající *TransportType* MQXPY_TCP, když je jedna z následujících podmínek pravdivá:

- *ConnectionName* kanálu je název hostitele, který se interpretuje jako adresa IPv4 i IPv6 a jeho parametr **LocalAddress** není uveden.
- *ConnectionName* a *LocalAddress* kanálu jsou názvy hostitelů, které se interpretují na adresy IPv4 i IPv6 .

Hodnota může být některá z následujících:

MQIPADDR_IPV4
IPv4 bude použita.

MQIPADDR_IPV6
IPv6 bude použita.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_IP_ADDRESS_VERSION s voláním MQINQ.

ListenerTimer (MQLONG)

Jedná se o časový interval (v sekundách) mezi pokusy produktu IBM MQ restartovat modul listener, pokud došlo k selhání APPC nebo TCP/IP. Hodnota musí být mezi 5 a 9999, s výchozí hodnotou 60.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_LISTENER_TIMER s voláním MQINQ.

LocalEvent (MQLONG)

Tato volba určuje, zda mají být generovány lokální chybové události. Hodnota je jedna z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_LOCAL_EVENT s voláním MQINQ.

V systému z/OSnemůžete použít volání MQINQ k určení hodnoty tohoto atributu.

LoggerEvent (MQLONG)

Tato volba určuje, zda mají být generovány události protokolu zotavení. Hodnota je jedna z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_LOGGER_EVENT s voláním MQINQ.

Multi Tento atribut je podporován pouze na systému Multiplatforms.

LUGroupName (MQCHAR8)

Toto je generické jméno LU pro modul listener 6.2 , který obsluhuje příchozí přenosy pro skupinu sdílení front. Ponecháte-li tento název prázdný, nemůžete tento modul listener použít.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_LU_GROUP_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_LU_NAME_LENGTH.

Název LUName (MQCHAR8)

Jedná se o název jednotky LU, která má být použita pro odchozí přenosy LU 6.2 . Nastavte tuto hodnotu na stejnou logickou jednotku, kterou modul listener používá pro příchozí přenosy. Ponecháte-li toto jméno prázdné, použije se předvolená logická jednotka APPC/MVS; jedná se o proměnnou, takže vždy nastavte hodnotu LUName, pokud používáte LU6.2.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_LU_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_LU_NAME_LENGTH.

LU62ARMSuffix (MQCHAR2)

Jedná se o příponu SYS1.PARMLIB člen APPCPMxx, který nominuje LUADD pro tento inicializátor kanálu. Příkaz z/OS SET APPC=xx je vydán, když ARM restartuje inicializátor kanálu. Ponecháte-li tento název prázdný, nebude vydán žádný příkaz SET APPC=xx.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_LU62_ARM_SUFFIX s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_ARM_SUFFIX_LENGTH.

LU62Channels (MQLONG)

Jedná se o maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni a kteří používají přenosový protokol LU 6.2 .

Hodnota musí být v rozsahu 0 až 9999 s výchozí hodnotou 200. Nastavíte-li tuto hodnotu na nulu, nebude použit přenosový protokol LU 6.2 .

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_LU62_CHANNELS s voláním MQINQ.

MaxActiveKanály (MQLONG)

Tento atribut představuje maximální počet kanálů, které mohou být *aktivní* kdykoli.

Výchozí hodnota je extrahována z atributu `MaxChannels`.

Pro parametr z/OSmusí být hodnota v rozsahu 1 až 9 999.

Pro všechny ostatní platformy je výchozí hodnota 999 999 999 999, což znamená, že počet aktivních kanálů je neomezený, nebo může být nastaven na skutečné číslo pro zavedení limitu.

MQ Appliance Hodnotu **MaxActiveChannels** byste neměli měnit na IBM MQ Appliance. Chcete-li omezit maximální počet kanálů klienta, použijte atributy MAXINST a MAXINSTC pro jednotlivé kanály v definicích kanálu SVRCONN k definování limitů pro každý kanál SVRCONN, viz téma Konfigurace správce front na zařízení IBM MQ Appliance v dokumentaci IBM MQ Appliance .

Parametr **MaxActiveChannels** je atributem správce front pouze v systému z/OS . Na ostatních platformách je atribut **MaxActiveChannels** atributem v souboru `qm.ini` . Informace o nastavení

atributu **MaxActiveChannels** na jiných platformách naleznete v části Sekce konfiguračního souboru pro distribuované fronty .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACTIVE_CHANNELS s voláním **MQINQ** .

Související pojmy
Stavy kanálů

MaxChannels (MQLONG)

Tento atribut představuje maximální počet kanálů, které mohou být *aktuální* (včetně kanálů připojení serveru s připojenými klienty).

Pro z/OSmusí být hodnota v rozsahu 1 až 9 999, s výchozí hodnotou 200.

Pro systém IBM MQ Applianceje výchozí hodnota 999 999 999 a neměla by se měnit. Chcete-li omezit maximální počet kanálů klienta, použijte atributy MAXINST a MAXINSTC pro jednotlivé kanály v definicích kanálu SVRCONN k definování limitů pro každý kanál SVRCONN, viz téma Konfigurace správce front na zařízení IBM MQ Appliance v dokumentaci IBM MQ Appliance .

Systém, který je zaneprázdněn obsluhováním připojení ze sítě, může vyžadovat vyšší číslo, než je výchozí nastavení. Určete hodnotu, která je správná pro vaše prostředí, v ideálním případě sledováním chování vašeho systému během testování.

Pro všechny ostatní platformy je výchozí hodnota 100. Můžete nastavit **MaxChannels** na jinou hodnotu, chcete-li v případě potřeby omezit maximální počet aktuálních kanálů.

Parametr **MaxChannels** je atributem správce front pouze v systému z/OS . Na ostatních platformách je atribut **MaxChannels** atributem v souboru `qm.ini` . Informace o nastavení atributu **MaxChannels** na jiných platformách naleznete v části Sekce konfiguračního souboru pro distribuované fronty .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_CHANNELS s voláním **MQINQ** .

Související pojmy
Stavy kanálů

MaxHandles (MQLONG)

Jedná se o maximální počet otevřených popisovačů, které může kterákoli úloha souběžně používat. Každé úspěšné volání MQOPEN pro jednu frontu (nebo pro objekt, který není frontou) používá jeden popisovač. Tento popisovač bude po zavření objektu k dispozici pro opětovné použití. Při otevření distribučního seznamu je však každé frontě v distribučním seznamu přidělen samostatný manipulátor, takže volání MQOPEN používá stejný počet manipulátorů jako fronty v distribučním seznamu. To je třeba vzít v úvahu při rozhodování o vhodné hodnotě pro *MaxHandles*.

Volání MQPUT1 provádí v rámci svého zpracování volání MQOPEN; v důsledku toho produkt MQPUT1 používá tolik manipulátorů, kolik by měl MQOPEN, ale manipulátory se používají pouze po dobu trvání samotného volání MQPUT1 .

V systému z/OS *úloha* znamená úlohu CICS , úlohu MVS nebo závislou oblast IMS .

Hodnota je v rozsahu 1 až 999 999 999 999. Výchozí hodnota je určena prostředím:

- V systému z/OSje výchozí hodnota 100.
- Ve všech ostatních prostředích je výchozí hodnota 256.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_MANIPULUJE s voláním MQINQ.

MaxMsgDélka (MQLONG)

Jedná se o délku nejdelší *fyzické* zprávy, kterou může správce front zpracovat. Protože však lze atribut správce front **MaxMsgLength** nastavit nezávisle na atributu fronty **MaxMsgLength** , nejdelší fyzická zpráva, kterou lze umístit do fronty, je nižší z těchto dvou hodnot.

Pokud správce front podporuje segmentaci, může aplikace vložit logickou zprávu, která je delší než menší ze dvou atributů **MaxMsgLength** , ale pouze v případě, že aplikace v MQMD určuje příznak

MQMF_SEGMENTATION_ALLOWED. Je-li uveden tento příznak, horní limit pro délku logické zprávy je 999 999 999 bajtů, ale obvykle omezení prostředků vynucené operačním systémem nebo prostředím, ve kterém je aplikace spuštěna, má za následek dolní limit.

Dolní limit pro atribut **MaxMsgLength** je 32 kB (32 768 bajtů). Horní limit je 100 MB (104 857 600 bajtů).

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_MSG_LENGTH s voláním MQINQ.

MaxPriority (MQLONG)

Jedná se o maximální prioritu zpráv podporovanou správcem front. Priority jsou v rozsahu od nuly (nejnižší) do *MaxPriority* (nejvyšší).

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_PRIORITY s voláním MQINQ.

MaxPropertiesDélka (MQLONG)

Používá se k řízení velikosti vlastností, které mohou proudit se zprávou. To zahrnuje jak název vlastnosti v bajtech, tak i velikost hodnoty vlastnosti v bajtech.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_PROPERTIES_LENGTH s voláním MQINQ.

Multi *Velikost MaxQFile(MQLONG)*

Maximální velikost (v megabajtech), na kterou může soubor fronty růst.

Tabulka 560. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Je možné, aby soubor fronty překročil maximální velikost, pokud je konfigurován na hodnotu nižší než aktuální velikost souboru fronty. Pokud k tomu dojde, soubor fronty již nepřijímá nové zprávy, ale umožňuje spotřebovat existující zprávy. Po zrušení velikosti souboru fronty pod konfigurovanou hodnotu je povoleno vkládat do fronty nové zprávy.

Poznámka: Tento obrázek se může lišit od hodnoty atributu konfigurovaného ve frontě, protože interně může být nutné, aby správce front použil větší velikost bloku k dosažení zvolené velikosti. Další informace o změně velikosti souborů fronty a velikosti bloku a granularity naleznete v tématu Úprava souborů fronty IBM MQ .

Když se granularita potřebuje změnit, protože byl tento atribut zvýšen, do protokolů AMQERR se zapíše varovná zpráva AMQ7493W `Granularita změněna` . To vám dává indikaci, že musíte naplánovat vyprázdnění fronty, aby produkt IBM MQ převzal novou granularitu.

Maximální hodnota tohoto atributu je 267 386 880 MB a výchozí hodnota a migrovaná hodnota je 2 088 960 MB, což je aktuální maximum pro frontu s granularitou rovnající se 512.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_Q_FILE_SIZE s voláním MQINQ.

MaxUncommittedPočet zpráv (MQLONG)

Jedná se o maximální počet nepotvrzených zpráv, které mohou existovat v rámci pracovní jednotky. Počet nepotvrzených zpráv je součtem následujících od spuštění aktuální pracovní jednotky:

- Zprávy vkládané aplikací s volbou MQPMO_SYNCPOINT
- Zprávy načtené aplikací s volbou MQGMO_SYNCPOINT
- Zprávy spouštěče a zprávy sestavy COA generované správcem front pro zprávy vložené s volbou MQPMO_SYNCPOINT
- Zprávy sestavy COD generované správcem front pro zprávy načtené pomocí volby MQGMO_SYNCPOINT

Následující zprávy se nepočítají jako nepotvrzené:

- Zprávy vkládané nebo načtené aplikací mimo pracovní jednotku

- Zprávy spouštěče nebo zprávy sestavy COA/COD generované správcem front v důsledku zpráv vložených nebo načtených mimo pracovní jednotku
- Zprávy sestavy vypršení platnosti generované správcem front (i v případě, že volání způsobující zprávu sestavy vypršení platnosti uvádí MQGMO_SYNCPOINT)
- Zprávy událostí generované správcem front (i když volání způsobující zprávu události uvádí MQPMO_SYNCPOINT nebo MQGMO_SYNCPOINT)

Poznámka:

1. Zprávy sestavy výjimek jsou generovány agentem MCA (Message Channel Agent) nebo aplikací a jsou zpracovány stejným způsobem jako běžné zprávy vkládané nebo načtené aplikací.
2. Je-li zpráva nebo segment vložen s volbou MQPMO_SYNCPOINT, počet nepotvrzených zpráv se zvýší o jednu bez ohledu na to, kolik fyzických zpráv je skutečně výsledkem vložení. (Pokud musí správce front zprávu nebo segment dále dělit, může dojít k více než jedné fyzické zprávě.)
3. Když je distribuční seznam vložen s volbou MQPMO_SYNCPOINT, počet nepotvrzených zpráv se zvýší o jeden *pro každou vygenerovanou fyzickou zprávu*. To může být tak malý jako jeden, nebo stejně velký jako počet destinací v distribučním seznamu.

Dolní limit pro tento atribut je 1; horní limit je 999 999 999. Výchozí hodnota je 10000.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_UNCOMMITTED_MSGS s voláním MQINQ.

MQIAccounting (MQLONG)

Tato volba řídí shromažďování účetních informací pro data MQI.

Hodnota je jedna z následujících:

MQMON_ON
Shromažďovat data evidence rozhraní API.

MQMON_OFF
Neshromažďujte data evidence rozhraní API. Toto je výchozí hodnota.

Nastavíte-li atribut správce front ACCTCONO na hodnotu ENABLED, může být tato hodnota přepsána pro jednotlivá připojení pomocí pole Volby ve struktuře MQCNO. Změny této hodnoty se projeví pouze u připojení ke správci front, ke kterým dojde po změně atributu.

Tento atribut je podporován pouze na systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACCOUNTING_MQI s voláním MQINQ.

MQIStatistics (MQLONG)

Tato volba řídí shromažďování statistických monitorovacích informací pro správce front.

Hodnota je jedna z následujících:

MQMON_ON
Shromážděte statistiku rozhraní MQI.

MQMON_OFF
Neshromažďovat statistiku rozhraní MQI. Toto je výchozí hodnota.

Tento atribut je podporován pouze na systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_STATISTICS_MQI s voláním MQINQ.

MsgMarkBrowseInterval (MQLONG)

Časový interval v milisekundách, po jehož uplynutí může správce front automaticky odebrat značku z procházení zpráv.

Jedná se o časový interval (v milisekundách), po kterém může správce front automaticky odebrat značku z procházení zpráv.

Tento atribut popisuje časový interval, po který mají zprávy, které byly označeny jako procházené voláním MQGET s použitím volby získání zprávy MQGMO_MARK_BROWSE_CO_OP, zůstat označeny jako procházené.

Správce front může automaticky zrušit označení procházených zpráv, které byly označeny jako procházené pro spolupracující sadu popisovačů, pokud byly označeny pro více než tento přibližný interval.

To nemá vliv na stav zprávy označené jako procházet, která byla získána voláním MQGET s použitím volby získání zprávy MQGMO_MARK_BROWSE_HANDLE.

Maximální hodnota je 999 999 999 a výchozí hodnota je 5000. Speciální hodnota -1 pro *MsgMarkBrowseInterval* představuje neomezený časový interval.

Upozornění: Tato hodnota by neměla být nižší než výchozí hodnota 5000.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MSG_MARK_BROWSE_INTERVAL s voláním MQINQ.

OutboundPortMax (MQLONG)

Jedná se o nejvyšší číslo portu v rozsahu definovaném hodnotou OutboundPortMin a OutboundPortMax pro čísla portů, která mají být použita pro vazbu odchozích kanálů.

Hodnota je celé číslo v rozsahu 0 až 65535 a musí být větší nebo rovna minimální hodnotě OutboundPort. Výchozí hodnota je 0.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_OUTBOUND_PORT_MAX s voláním MQINQ.

OutboundPortMin (MQLONG)

Jedná se o nejnižší číslo portu v rozsahu definovaném hodnotou OutboundPortMin a OutboundPortMax pro čísla portů, která mají být použita pro vazbu odchozích kanálů.

Hodnota je celé číslo v rozsahu 0 až 65535 a musí být menší nebo rovna maximální hodnotě OutboundPort. Výchozí hodnota je 0.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_OUTBOUND_PORT_MIN s voláním MQINQ.

PerformanceEvent (MQLONG)

Tato volba určuje, zda jsou generovány události související s výkonem. Jedná se o jednu z následujících hodnot:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PERFORMANCE_EVENT s voláním MQINQ.

Platforma (MQLONG)

To označuje operační systém, na kterém je spuštěn správce front:

MQPL_AIX
AIX (stejná hodnota jako MQPL_UNIX).

MQPL_APPLIANCE
IBM MQ Appliance

MQPL_MVS
z/OS (stejná hodnota jako MQPL_ZOS).

MQPL_OS390
z/OS (stejná hodnota jako MQPL_ZOS).

MQPL_OS400
IBM i.

MQPL_UNIX
UNIX.

MQPL_WINDOWS_NT
Systémy Windows .

MQPL_ZOS
z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PLATFORM s voláním MQINQ.

PubSubNPInputMsg (MQLONG)
Zda se má zrušit nebo uchovat nedoručená vstupní zpráva.

Hodnota je jedna z následujících:

MQUNDELIVERED_DISCARD
Netrvalé vstupní zprávy lze odložit, pokud je nelze zpracovat.

Toto je výchozí hodnota.

MQUNDELIVERED_KEEP
Netrvalé vstupní zprávy nebudou odloženy, pokud je nelze zpracovat. V této situaci bude rozhraní pro publikování/odběr ve frontě pokračovat v opakování procesu v příslušných intervalech a nebude pokračovat ve zpracování následných zpráv.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PUBSUB_NP_MSG s voláním MQINQ.

PubSubNPResponse (MQLONG)
Řídí chování nedoručených zpráv odpovědi.

Hodnota je jedna z následujících:

MQUNDELIVERED_NORMAL
Dočasné odpovědi, které nelze umístit do fronty odpovědí, jsou umístěny do fronty nedoručených zpráv. Pokud nemohou být umístěny do fronty nedoručených zpráv, jsou vyřazeny.

MQUNDELIVERED_SAFE
Netrvalé odpovědi, které nelze umístit do fronty odpovědí, jsou umístěny do fronty zablokovaných zpráv (DLQ). Pokud odpověď nelze nastavit a nelze ji umístit do fronty DLQ, rozhraní publikování/odběru ve frontě odvolá aktuální operaci a poté pokus zopakuje ve vhodných intervalech a nebude pokračovat ve zpracování následných zpráv.

MQUNDELIVERED_DISCARD
Dočasné odpovědi nejsou umístěny do fronty odpovědí, jsou vyřazeny.

Jedná se o výchozí hodnotu pro nové správce front.

MQUNDELIVERED_KEEP
Dočasné odpovědi nejsou umístěny do fronty nedoručených zpráv nebo vyřazeny. Místo toho rozhraní pro publikování/odběr ve frontě vrátí zpět aktuální operaci a poté ji v příslušných intervalech zopakuje.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PUBSUB_NP_RESP s voláním MQINQ.

Výchozí hodnota pro migrované správce front.
Pokud byl správce front migrován z produktu IBM MQ V6.0, počáteční hodnota tohoto atributu závisí na hodnotách *DiscardNonPersistentResponse* a *DLQNonPersistentResponse* před migrací, jak ukazuje následující tabulka.

		DLQNonPersistentOdezva		
		Ano	**Ne**	**Nenastaveno**
DiscardNonPersistentResponse	**Ano**	MQUNDELIVERED_NORMAL	MQUNDELIVERED_DISCARD	MQUNDELIVERED_NORMAL
	Ne	MQUNDELIVERED_SAFE	MQUNDELIVERED_KEEP	MQUNDELIVERED_SAFE
	Nenastaveno	Je-li volba SyncPointtrvalá = Ne, MQUNDELIVERED_SAFE, jinak MQUNDELIVERED_NORMAL	Je-li hodnota SyncPointTrvalý = Ne, MQUNDELIVERED_KEEP, jinak MQUNDELIVERED_DISCARD	Je-li volba SyncPointtrvalá = Ne, MQUNDELIVERED_SAFE, jinak MQUNDELIVERED_NORMAL

PubSubMaxMsgRetryCount (MQLONG)

Počet opakování při zpracování nezdařené zprávy příkazu pod synchronizačním bodem.

Hodnota je jedna z následujících:

0-999 999 999 999
Výchozí hodnota je 5.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PUBSUB_MAXMSG_RETRY_COUNT s voláním MQINQ.

PubSubSyncPoint (MQLONG)

Zda jsou pod synchronizačním bodem zpracovány pouze trvalé zprávy nebo všechny zprávy.

Hodnota je jedna z následujících:

MQSYNCPOINT_IFPER
To způsobí, že rozhraní publikování/odběru zařazené ve frontě bude přijímat dočasné zprávy mimo synchronizační bod. Pokud démon obdrží publikaci mimo bod synchronizace, démon pošle publikaci odběratelům, známým mimo bod synchronizace.

Toto je výchozí hodnota.

MQSYNCPOINT_YES
To způsobí, že rozhraní publikování/odběru zařazené do fronty přijme všechny zprávy pod synchronizačním bodem.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PUBSUB_SYNC_PT s voláním MQINQ.

Režim PubSub(MQLONG)

Zda je spuštěn stroj publikování/odběru a rozhraní publikování/odběru ve frontě, což umožňuje aplikacím publikovat/odebírat pomocí rozhraní API a front, které jsou monitorovány rozhraním publikování/odběru ve frontě.

Hodnota je jedna z následujících:

MQPSM_COMPAT
Stroj pro publikování/odběr je spuštěn. Proto je možné publikovat/přihlásit se k odběru pomocí rozhraní API. Rozhraní publikování/odběru ve frontě není spuštěno, proto není žádná zpráva vložená do front, které jsou monitorovány rozhraním publikování/odběru ve frontě, zpracovávána. Toto nastavení se používá pro kompatibilitu s produktem WebSphere Message Broker V6 nebo s dřívějšími verzemi, které používají tohoto správce front, protože musí číst stejné fronty, ze kterých běžně čte rozhraní publikování/odběru zařazené ve frontě.

MQPSM_DISABLED
Stroj pro publikování/odběr a rozhraní pro publikování/odběr ve frontě nejsou spuštěny. Proto není možné publikovat/odebírat pomocí rozhraní API. Žádné zprávy publikování/odběru, které jsou vkládány do front, které jsou monitorovány rozhraním pro publikování/odběr ve frontě, se nepoužívají.

MQPSM_ENABLED
Stroj publikování/odběru a rozhraní publikování/odběru ve frontě jsou spuštěny. Proto je možné publikovat/odebírat pomocí rozhraní API a front, které jsou monitorovány rozhraním publikování/odběru zařazeným do fronty. Jedná se o počáteční výchozí hodnotu správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PUBSUB_MODE s voláním MQINQ.

QMgrDesc (MQCHAR64)

Toto pole použijte pro komentář popisující správce front. Obsah pole nemá pro správce front žádný význam, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může toto pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou obsaženy ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

- V systému z/OSje výchozí hodnota název produktu a číslo verze.
- Ve všech ostatních prostředích je výchozí hodnota prázdná.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_Q_MGR_DESC s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_MGR_DESC_LENGTH.

QMgrIdentifier (MQCHAR48)

Jedná se o interně generovaný jedinečný název pro správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_Q_MGR_IDENTIFIER s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_MGR_IDENTIFIER_LENGTH.

Tento atribut je podporován v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows
- z/OS

a IBM MQ klientů připojených k těmto systémům.

QMgrName (MQCHAR48)

Jedná se o název lokálního správce front, tj. název správce front, ke kterému je aplikace připojena.

Prvních 12 znaků názvu se používá k vytvoření jedinečného identifikátoru zprávy (viz pole MQMD- MsgId). Správci front, kteří mohou komunikovat, proto musí mít názvy, které se liší v prvních 12 znacích, aby byly identifikátory zpráv v síti správců front jedinečné.

V systému z/OSje název stejný jako název subsystému, který je omezen na 4 neprázdné znaky.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_Q_MGR_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

Název oblasti QSGName (MQCHAR4)

Jedná se o název skupiny sdílení front, do které patří lokální správce front. Pokud lokální správce front nepatří do skupiny sdílení front, je název prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_QSG_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_QSG_NAME_LENGTH.

z/OS Tento atribut je podporován pouze na systému z/OS.

QueueAccounting (MQLONG)

Tato volba řídí shromažďování informací o monitorování účtů pro fronty.

Hodnota je jedna z následujících:

MQMON_NONE
Neshromažďujte data evidence pro fronty, bez ohledu na nastavení atributu evidence front ACCTQ. Toto je výchozí hodnota.

MQMON_OFF
Neshromažďujte data evidence pro fronty, které uvádějí QMGR v atributu fronty ACCTQ.

MQMON_ON
Shromážděte data evidence pro fronty, které uvádějí QMGR v atributu fronty ACCTQ.

Změny této hodnoty se projeví pouze u připojení ke správci front, ke kterým dojde po změně atributu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACCOUNTING_Q s voláním MQINQ.

QueueMonitoring (MQLONG)
Tato volba určuje výchozí nastavení pro monitorování front online.

Pokud je atribut fronty **QueueMonitoring** nastaven na MQMON_Q_MGR, tento atribut určuje hodnotu, kterou kanál přebírá. Hodnota může být následující:

MQMON_OFF
Shromažďování dat monitorování online je vypnuto. Jedná se o počáteční výchozí hodnotu správce front.

MQMON_NONE
Online shromažďování dat monitorování je vypnuto pro fronty bez ohledu na nastavení jejich atributu **QueueMonitoring** .

MQMON_LOW
Online shromažďování dat monitorování je zapnuto s nízkým poměrem shromažďování dat.

MQMON_MEDIUM
Online shromažďování dat monitorování je zapnuto se středním poměrem shromažďování dat.

MQMON_HIGH-vysoká
Online shromažďování dat monitorování je zapnuto s vysokým poměrem shromažďování dat.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MONITORING_Q s voláním MQINQ.

QueueStatistics (MQLONG)

Tato volba řídí shromažďování statistických dat pro fronty.

Jedná se o jednu z následujících hodnot:

MQMON_NONE
Neshromažďujte statistiku front pro fronty bez ohledu na nastavení atributu fronty **QueueStatistics** . Toto je výchozí hodnota.

MQMON_OFF
Neshromažďujte statistická data pro fronty, které určují správce front v atributu fronty **QueueStatistics** .

MQMON_ON
Shromážděte statistická data pro fronty, které určují správce front v atributu fronty **QueueStatistics** .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_STATISTICS_Q s voláním MQINQ.

ReceiveTimeout (MQLONG)
Tato volba určuje, jak dlouho kanál TCP/IP čeká na příjem dat, včetně prezenčních signálů, od svého partnera před návratem do neaktivního stavu. Vztahuje se pouze na kanály zpráv a nikoli na kanály MQI.

Přesný význam parametru ReceiveTimeout je změněn hodnotou určenou v poli ReceiveTimeout. Typ ReceiveTimeoutlze nastavit na jednu z následujících hodnot:

- MQRCVTIME_EQUAL-tato hodnota je počet sekund, po které má kanál čekat. Uveďte hodnotu v rozsahu 0-999999.
- MQRCVTIME_ADD-tato hodnota představuje počet sekund, který má být přidán k vyjednanému HBINT, a určuje, jak dlouho má kanál čekat. Uveďte hodnotu v rozsahu 1-999999.
- MQRCVTIME_MULTIPLY-tato hodnota je multiplikátor, který se má použít na vyjednaný HBINT. Zadejte hodnotu 0 nebo hodnotu v rozsahu 2-99.

Výchozí hodnota je 0.

Nastavte volbu ReceiveTimeoutna hodnotu MQRCVTIME_MULTIPLY nebo MQRCVTIME_EQUAL a volbu ReceiveTimeout na hodnotu 0, chcete-li zabránit kanálu v vypršení časového limitu čekání na přijetí dat od jeho partnera.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_RECEIVE_TIMEOUT s voláním MQINQ.

ReceiveTimeoutMin (MQLONG)

Jedná se o minimální dobu (v sekundách), po kterou kanál TCP/IP čeká na přijetí dat, včetně prezenčních signálů, od svého partnera, než se vrátí do neaktivního stavu.

Vztahuje se pouze na kanály zpráv, nikoli na kanály MQI. Hodnota musí být v rozsahu 0 až 999999 s výchozí hodnotou 0.

Pokud použijete volbu ReceiveTimeoutType k určení, že doba čekání kanálu TCP/IP se má vypočítat vzhledem k vyjednané hodnotě HBINT a výsledná hodnota je menší než hodnota tohoto parametru, použije se místo toho tato hodnota.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_RECEIVE_TIMEOUT_MIN s voláním MQINQ.

Typ ReceiveTimeout(MQLONG)

Jedná se o kvalifikátor použitý pro ReceiveTimeout , který definuje, jak dlouho kanál TCP/IP čeká na přijetí dat, včetně prezenčních signálů, od svého partnera, než se vrátí do neaktivního stavu. Vztahuje se pouze na kanály zpráv, nikoli na kanály MQI.

Hodnota je jedna z následujících:

MQRCVTIME_MULTIPLY
ReceiveTimeout je multiplikátor, který se má použít na vyjednanou hodnotu HBINT k určení, jak dlouho kanál čeká. Toto je výchozí hodnota.

MQRCVTIME_ADD
ReceiveTimeout je hodnota v sekundách, která se má přidat k vyjednané hodnotě HBINT, aby se určilo, jak dlouho kanál čeká.

MQRCVTIME_EQUAL-další informace
ReceiveTimeout je hodnota v sekundách, po kterou kanál čeká.

Chcete-li zastavit vypršení časového limitu kanálu při čekání na příjem dat od partnera, nastavte volbu ReceiveTimeoutna hodnotu MQRCVTIME_MULTIPLY nebo MQRCVTIME_EQUAL a volbu ReceiveTimeout na hodnotu 0.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_RECEIVE_TIMEOUT_TYPE s voláním MQINQ.

RemoteEvent (MQLONG)

Toto řídí, zda jsou generovány vzdálené chybové události. Jedná se o jednu z následujících hodnot:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_REMOTE_EVENT s voláním MQINQ.

RepositoryName (MQCHAR48)

Jedná se o název klastru, pro který tento správce front poskytuje službu správce úložiště. Pokud správce front poskytuje tuto službu pro více než jeden klastr, parametr *RepositoryNamelist* určuje název objektu seznamu názvů, který identifikuje klastry, a parametr *RepositoryName* je prázdný. Alespoň jeden z *RepositoryName* a *RepositoryNamelist* musí být prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_REPOSITORY_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

RepositoryNamelist (MQCHAR48)

Jedná se o název objektu seznamu názvů, který obsahuje názvy klastrů, pro které tento správce front poskytuje službu správce úložiště. Pokud správce front poskytuje tuto službu pouze pro jeden klastr, obsahuje objekt seznamu názvů pouze jeden název. Alternativně lze *RepositoryName* použít k určení názvu klastru, v takovém případě je *RepositoryNamelist* prázdný. Alespoň jeden z *RepositoryName* a *RepositoryNamelist* musí být prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_REPOSITORY_NAMELIST s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_NAMELIST_NAME_LENGTH.

ScyCase(MQCHAR8)

Určuje, zda správce front podporuje názvy profilů zabezpečení s použitím velkých i malých písmen nebo pouze velkých písmen.

Hodnota je jedna z následujících:

MQSCYC_UPPER
Názvy profilů zabezpečení musí být uvedeny velkými písmeny.

MQSCYC_MIXED
Názvy profilů zabezpečení mohou být psány velkými písmeny nebo smíšenými písmeny.

Změny tohoto atributu se projeví po spuštění příkazu Refresh Security s uvedeným parametrem *SecurityType(MQSECTYPE_CLASSES)* .

z/OS Tento atribut je podporován pouze na systému z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SECURITY_CASE s voláním MQINQ.

SharedQMgrNázev (MQLONG)

Tato volba určuje, zda má být pro sdílenou frontu při volání MQOPEN použita hodnota *ObjectQmgrName* jako lokální správce front nebo zda s ní má být zacházeno jako s lokálním správcem front pro sdílenou frontu, pokud je hodnota *ObjectQmgrName* určena pro jiného správce front ve skupině sdílení front.

Hodnota může být některá z následujících:

MQSQQM_USE
Použije se *ObjectQmgrName* a otevře se příslušná přenosová fronta.

MQSQQM_IGNORE
Pokud je cílová fronta sdílená a *ObjectQmgrName* je fronta správce front ve stejné skupině sdílení front, otevření se provede lokálně.

Tento atribut je platný pouze na systému z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SHARED_Q_Q_MGR_NAME volání MQINQ.

SPLCAP
Označuje, zda jsou pro správce front k dispozici funkce zabezpečení produktu Advanced Message Security .

MQCAP_SUPPORTED-podporováno
Jedná se o výchozí hodnotu, pokud je pro instalaci, pod kterou je spuštěn správce front, nainstalována komponenta AMS .

MQCAP_NOT_SUPPORTED

SSLEvent (MQLONG)
Určuje, zda jsou generovány události TLS.

Jedná se o jednu z následujících hodnot:

MQEVR_ENABLED
Generujte události TLS takto:

MQRC_CHANNEL_SSL_ERROR

MQEVR_DISABLED
Negenerovat události TLS; toto je výchozí hodnota.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SSL_EVENT s voláním MQINQ.

SSLFIPSRequired (MQLONG)

Poznámka: V systému AIX, Linux, and Windowsposkytuje produkt IBM MQ kompatibilitu se standardem FIPS 140-2 prostřednictvím šifrovacího modulu IBM Crypto for C (ICC) . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C (ICC) certifikát a měli by si být vědomi všech doporučení poskytnutých NIST. Náhradní modul FIPS 140-3 momentálně probíhá a jeho stav lze zobrazit jeho vyhledáním v modulech NIST CMVP v seznamu procesů.

To vám umožní určit, že se mají použít pouze algoritmy certifikované FIPS, pokud se šifrování provádí v produktu IBM MQ, spíše než v šifrovacím hardwaru. Pokud je kryptografický hardware nakonfigurován, použité šifrovací moduly jsou moduly poskytované hardwarovým produktem; tyto moduly mohou nebo nemusí být certifikovány FIPS na konkrétní úroveň v závislosti na používaném hardwarovém produktu.

Hodnota je jedna z následujících hodnot:

MQSSL_FIPS_NO
Použijte libovolnou CipherSpec podporovanou na používané platformě. Tato hodnota je výchozí hodnota.

MQSSL_FIPS_YES
Používejte pouze šifrovací algoritmy certifikované FIPS v CipherSpecs povolených pro všechna připojení TLS z tohoto správce front a do tohoto správce front.

Tento parametr je platný pouze na platformách z/OS, AIX, Linux, and Windows .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SSL_FIPS_REQUIRED s voláním MQINQ.

Související úlohy
Určení, že za běhu jsou v klientu MQI použity pouze specifikace CipherSpecs s certifikací FIPS.
Související odkazy
Standard FIPS (Federal Information Processing Standards) pro AIX, Linux, and Windows

Počet operací SSLKeyReset(MQLONG)
Tato volba určuje, kdy agenti kanálu zpráv TLS (MCA), kteří zahajují komunikaci, vynulují tajný klíč použitý pro šifrování na kanálu.

Hodnota reprezentuje celkový počet nezašifrovaných bajtů, které jsou odeslány a přijaty na kanálu před novým vyjednáním tajného klíče. Počet bajtů zahrnuje řídicí informace odeslané MCA.

Hodnota je číslo v rozsahu 0 až 999 999 999 999, s výchozí hodnotou 0. Zadáte-li počet resetů tajného klíče TLS v rozsahu 1 bajt až 32 kB, budou kanály TLS používat počet resetů tajného klíče 32 kB. Tím se vyhnete nákladům na zpracování nadměrných resetů klíčů, které by se vyskytly pro malé hodnoty resetu tajného klíče TLS.

Tajný klíč je znovu vyjednán, když celkový počet nešifrovaných bajtů odeslaných a přijatých inicializujícím kanálem MCA překročí uvedenou hodnotu. Pokud jsou povoleny prezenční signály kanálu, tajný klíč je znovu vyjednán před odesláním nebo přijetím dat po synchronizačním signálu kanálu, nebo když celkový počet nešifrovaných bajtů překročí uvedenou hodnotu, podle toho, co nastane dříve.

Počet bajtů odeslaných a přijatých k opětovnému vyjednání zahrnuje řídicí informace odeslané a přijaté kanálem MCA a resetuje se vždy, když dojde k opětovnému vyjednání.

Chcete-li označit, že tajné klíče nejsou nikdy znovu vyjednány, použijte hodnotu 0.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SSL_RESET_COUNT s voláním MQINQ.

Událost StartStop(MQLONG)

Tato volba určuje, zda mají být generovány události spuštění a zastavení. Hodnota je jedna z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_START_STOP_EVENT s voláním MQINQ.

StatisticsInterval (MQLONG)

Určuje, jak často (v sekundách) se mají data monitorování statistiky zapisovat do fronty monitorování.

Hodnota je celé číslo v rozsahu 0 až 604800 s výchozí hodnotou 1800 (30 minut).

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_STATISTICS_INTERVAL s voláním MQINQ.

SyncPoint (MQLONG)

Označuje, zda lokální správce front podporuje pracovní jednotky a synchronizaci s voláními MQGET, MQPUT a MQPUT1 .

MQSP_AVAILABLE
K dispozici jsou jednotky práce a synchronizace.

MQSP_NOT_AVAILABLE
Pracovní jednotky a synchronizační funkce nejsou k dispozici.

- V systému z/OS není tato hodnota nikdy vrácena.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SYNCPOINT s voláním MQINQ.

TCPChannels (MQLONG)

Toto je maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni, kteří používají přenosový protokol TCP/IP.

Hodnota musí být v rozsahu 0 až 9999 s výchozí hodnotou 200. Zadáte-li hodnotu 0, protokol TCP/IP se nepoužije.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TCP_CHANNELS s voláním MQINQ.

TCPKeepAlive (MQLONG)

Uvádí, zda se má použít KEEPALIVE TCP ke kontrole, že druhý konec připojení je stále k dispozici. Jestliže není k dispozici, je kanál uzavřen.

Hodnota je jedna z následujících:

MQTCPKEEP_YES
Použijte volbu TCP KEEPALIVE, jak je uvedeno v datové sadě konfigurace profilu TCP. Zadáte-li atribut kanálu KeepAliveInterval (KAINT), použije se hodnota, na kterou je nastaven.

MQTCPKEEP_NO
Nepoužívejte volbu KEEPALIVE protokolu TCP. Toto je výchozí hodnota.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TCP_KEEP_ALIVE s voláním MQINQ.

TCPName (MQCHAR8)

Jedná se o název jediného nebo upřednostňovaného zásobníku TCP/IP, který bude použit, v závislosti na hodnotě TCPStackType. Tento parametr lze použít pouze v prostředích s více zásobníky CINET. Výchozí hodnota je TCPIP.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_TCP_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_TCP_NAME_LENGTH.

TCPStackType (MQLONG)

Tato volba určuje, zda může inicializátor kanálu použít pouze zásobník TCP/IP určený v parametru TCPName, nebo zda může volitelně vytvořit vazbu s libovolně vybraným zásobníkem TCP/IP. Tento parametr lze použít pouze v prostředích s více zásobníky CINET.

Hodnota je jedna z následujících:

MQTCPSTACK_SINGLE
Inicializátor kanálu může používat pouze adresní prostory TCP/IP uvedené v TCPName. Toto je výchozí hodnota.

MQTCPSTACK_MULTIPLE
Inicializátor kanálu může používat libovolný adresní prostor TCP/IP, který má k dispozici. Je-li pro kanál nebo modul listener zadána jiná hodnota TCPName, použije se výchozí hodnota TCPName.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TCP_STACK_TYPE s voláním MQINQ.

Záznam TraceRoute(MQLONG)

Tato volba řídí záznam informací o trasovací trase.

Hodnota je jedna z následujících:

MQRECORDING_DISABLED
Není povoleno připojení ke zprávám trasovací trasy.

MQRECORDING_Q
Vložit zprávy trasování-směrovat do pevné pojmenované fronty.

MQRECORDING_MSG
Vložit zprávy trasování-směrovat zprávy do fronty určené pomocí zprávy samotné. Toto je výchozí hodnota

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TRACE_ROUTE_RECORDING s voláním MQINQ.

TriggerInterval (MQLONG)

Jedná se o časový interval (v milisekundách) používaný k omezení počtu zpráv spouštěče. To je relevantní pouze v případě, že *TriggerType* je MQTT_FIRST. V tomto případě jsou zprávy spouštěče obvykle generovány pouze tehdy, když do fronty dorazí vhodná zpráva a fronta byla dříve prázdná. Za určitých okolností však lze pomocí spouštěče MQTT_FIRST vygenerovat další zprávu spouštěče, a to i v případě, že fronta nebyla prázdná. Tyto další zprávy spouštěče se negenerují častěji než každých *TriggerInterval* milisekund.

Další informace o spouštění naleznete v tématu Spouštěcí kanály.

Hodnota není menší než 0 a větší než 999 999 999 999. Výchozí hodnota je 999 999 999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TRIGGER_INTERVAL s voláním MQINQ.

TriggerInterval (MQLONG)

Jedná se o časový interval (v milisekundách) používaný k omezení počtu zpráv spouštěče. To je relevantní pouze v případě, že *TriggerType* je MQTT_FIRST. V tomto případě jsou zprávy spouštěče obvykle generovány pouze tehdy, když do fronty dorazí vhodná zpráva a fronta byla dříve prázdná. Za určitých okolností však lze pomocí spouštěče MQTT_FIRST vygenerovat další zprávu spouštěče, a to i v případě, že fronta nebyla prázdná. Tyto další zprávy spouštěče se negenerují častěji než každých *TriggerInterval* milisekund.

Další informace o spouštění naleznete v tématu Spouštěcí kanály.

Hodnota není menší než 0 a větší než 999 999 999 999. Výchozí hodnota je 999 999 999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TRIGGER_INTERVAL s voláním MQINQ.

Verze (MQCFST)

Toto je verze kódu IBM MQ jako VVRRMMFF, kde:

VV-Verze

RR-uvolnění

MM-Úroveň údržby

FF-Úroveň opravy

XrCapability(MQLONG)

Tato volba určuje, zda jsou příkazy MQ Telemetry podporovány správcem front.

Hodnota je jedna z následujících:

MQCAP_SUPPORTED-podporováno
Nainstalovaná komponenta MQ Telemetry a podporované příkazy telemetrie.

MQCAP_NOT_SUPPORTED
Komponenta MQ Telemetry není nainstalována.

Tento atribut je podporován pouze na systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_XR_CAPABILITY s voláním MQINQ .

Atributy pro fronty

Existuje pět typů definice fronty. Některé atributy fronty platí pro všechny typy front; jiné atributy fronty platí pouze pro určité typy front.

Typy front

Správce front podporuje následující typy definic front:

Lokální fronta
Zprávy můžete ukládat do lokální fronty.

z/OS V systému z/OS jej můžete nastavit jako sdílenou nebo soukromou frontu.

Fronta je programu známa jako *lokální*, pokud ji vlastní správce front, ke kterému je program připojen. Do lokální fronty můžete vkládat zprávy a získávat je z ní.

Objekt definice fronty obsahuje informace o definici fronty spolu s fyzickými zprávami vloženými do této fronty.

Lokální fronta správce front
Fronta existuje v lokálním správci front.

z/OS Fronta je známá jako soukromá fronta v systému z/OS.

z/OS **Sdílená fronta (pouzez/OS)**
Fronta existuje ve sdíleném úložišti, které je přístupné všem správcům front, kteří patří do skupiny sdílení front, jež je vlastníkem sdíleného úložiště.

Aplikace připojené k libovolnému správci front ve skupině sdílení front mohou umísťovat zprávy do front tohoto typu a odebírat zprávy z front tohoto typu. Tyto fronty jsou ve skutečnosti stejné jako lokální fronty. Hodnota atributu fronty **QType** je `MQQT_LOCAL`.

Aplikace připojené k lokálnímu správci front mohou umisťovat zprávy do front tohoto typu a odebírat je z nich. Hodnota atributu fronty **QType** je `MQQT_LOCAL`.

Fronta klastru

Zprávy můžete ukládat do fronty klastru ve správci front, kde jsou definovány. Fronta klastru je fronta, jejímž hostitelem je správce front klastru, a která je dostupná ostatním správcům front v klastru. Hodnota atributu fronty **QType** je `MQQT_CLUSTER`.

Definice fronty klastru se oznamuje ostatním správcům front v klastru. Ostatní správci front v klastru mohou vkládat zprávy do fronty klastru, aniž by potřebovali odpovídající definici vzdálené fronty. Fronta klastru může být oznámena ve více než jednom klastru pomocí seznamu názvů klastrů.

Po oznámení fronty může každý správce front v klastru do ní vkládat zprávy. Chcete-li správce front vložit zprávu, musí z úplných úložišť zjistit, kdo je hostitelem této fronty. Pak přidá do zprávy informace o směrování a vloží zprávu do přenosové fronty klastru.

Správce front může ukládat zprávy pro ostatní správce front z klastru do více přenosových front. Správce front můžete nakonfigurovat tak, aby ukládal zprávy do více přenosových front klastru, dvěma různými způsoby. Nastavíte-li atribut správce front **DEFCLXQ** na hodnotu `CHANNEL`, bude pro každý odesílací kanál klastru v produktu `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE` automaticky vytvořena odlišná přenosová fronta klastru. Pokud nastavíte volbu přenosové fronty `CLCHNAME` tak, aby se shodovala s jedním nebo více odesílacími kanály klastru, bude správce front moci ukládat zprávy pro odpovídající kanály do těchto přenosových front.

Upozornění: Používáte-li vyhrazenou hodnotu `SYSTEM.CLUSTER.TRANSMIT.QUEUES` se správcem front, který byl upgradován z verze produktu starší než IBM WebSphere MQ 7.5, ujistěte se, že má `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE` volbu SHARE/NOSHARE nastavenou na hodnotu **SHARE**.

z/OS Fronta klastru může být fronta, kterou sdílí členové skupiny sdílení front v produktu IBM MQ for z/OS.

Vzdálená fronta
Vzdálená fronta není fyzickou frontou. Jedná se o lokální definici fronty, která existuje ve vzdáleném správci front. Lokální definice vzdálené fronty obsahuje informace, které lokálnímu správci front sdělují, jak má směrovat zprávy do vzdáleného správce front.

Aplikace připojené k lokálnímu správci front mohou umisťovat zprávy do front tohoto typu. Zprávy jsou umístěny do lokální přenosové fronty používané ke směrování zpráv do vzdáleného správce front. Aplikace nemohou odebírat zprávy ze vzdálených front. Hodnota atributu fronty **QType** je `MQQT_REMOTE`.

Definici vzdálené fronty můžete také použít pro:

- Alias fronty odpovědí

  V tomto případě je název definice názvem fronty pro odpověď. Další informace naleznete v tématu Aliasy fronty pro odpověď a klastry.

- Aliasy správce front

  V tomto případě je název definice aliasem pro správce front, nikoli názvem fronty. Další informace naleznete v tématu Aliasy a klastry správce front.

Fronta aliasů
Nejedná se o fyzickou frontu; jedná se o alternativní název pro lokální frontu, sdílenou frontu, frontu klastru nebo vzdálenou frontu. Název fronty, na kterou se alias interpretuje, je součástí definice alias fronty.

Aplikace připojené k lokálnímu správci front mohou umisťovat zprávy do front tohoto typu; zprávy jsou umisťovány do fronty, do které se alias převádí. Aplikace mohou odebrat zprávy z front tohoto typu, pokud se alias interpretuje jako lokální fronta, sdílená fronta nebo fronta klastru, která má lokální instanci. Hodnota atributu fronty **QType** je MQQT_ALIAS.

Modelová fronta
Nejedná se o fyzickou frontu; jedná se o sadu atributů fronty, ze které lze vytvořit lokální frontu.

Zprávy nelze ukládat do front tohoto typu.

limity front

V produktu IBM MQ 9.2.0máte možnost konfigurovat a monitorovat fronty, které budou podporovat podstatně více než dva terabajtové výchozí limity používané v dřívějších vydáních produktu IBM MQ. Máte také možnost zmenšit velikost, na kterou může soubor fronty růst.

Chcete-li povolit konfiguraci front, můžete použít atribut **MAXFSIZE** v lokálních a modelových frontách a k monitorování front, můžete použít atributy stavu front **CURFSIZE** a **CURMAXFS**.

Další informace naleznete v tématu Úprava souborů fronty IBM MQ.

Atributy fronty

Některé atributy fronty platí pro všechny typy front; jiné atributy fronty platí pouze pro určité typy front. Typy front, pro které se atribut používá, jsou zobrazeny v tabulce Tabulka 561 na stránce 839 a následných tabulkách.

Tabulka 561 na stránce 839 shrnuje atributy, které jsou specifické pro fronty. Atributy jsou popsány v abecedním pořadí.

Poznámka: Názvy atributů zobrazené v této sekci jsou popisné názvy používané s voláními MQINQ a MQSET ; názvy jsou stejné jako pro příkazy PCF. Při použití příkazů MQSC k definování, změně nebo zobrazení atributů jsou použity alternativní krátké názvy. Podrobnosti naleznete v části Příkazy MQSC .

V následující tabulce platí následující sloupce:

- Sloupec pro lokální fronty platí také pro sdílené fronty.
- Sloupec pro modelové fronty označuje, které atributy jsou zděděny lokální frontou vytvořenou z modelové fronty.
- Sloupec pro fronty klastru označuje atributy, které lze zjistit, když je fronta klastru otevřena pouze pro dotazování nebo pro dotazování a výstup. Pokud jsou zjišťovány další atributy, volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_SELECTOR_NOT_FOR_TYPE (2068).

  Je-li fronta klastru otevřena pro dotazování plus jeden nebo více vstupů, procházení nebo nastavení, použije se místo toho sloupec pro lokální fronty.

  Je-li fronta klastru otevřena pouze pro dotaz nebo pro dotaz a výstup a je-li zadán název správce základní fronty, použije se místo toho sloupec pro lokální fronty.

Tabulka 561. Atributy pro fronty

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
AlterationDate	Datum, kdy byla definice naposledy změněna	X		X	X	
AlterationTime	Čas, kdy byla definice naposledy změněna	X		X	X	
BackoutRequeueQName	Nadměrný název fronty vrácení	X	X			
BackoutThreshold	Práh vrácení	X	X			
BaseQName	Název fronty, na kterou se alias interpretuje			X		
CFStrucName	Název struktury prostředku Coupling Facility	X	X			
CLCHNAME	Názvy kanálů odesilatele klastru	✓	✓			
ClusterName	Název klastru, ke kterému fronta patří	X		X	X	X
ClusterNamelist	Název objektu seznamu názvů obsahujícího názvy klastrů, ke kterým fronta náleží	X		X	X	
CLWLQueuePriority	Priorita fronty pracovní zátěže klastru	X		X	X	X
CLWLQueueRank	Skupina front pracovní zátěže klastru	X		X	X	X
CLWLUseQ	Použít vzdálenou frontu	X				
CreationDate	Datum, kdy byla fronta vytvořena	X				
CreationTime	Čas, kdy byla fronta vytvořena	X				
CurrentQDepth	Aktuální hloubka fronty	X				
DefaultPutResponse	Odezva výchozího umístění	✓	✓	✓	✓	
DefBind	Výchozí vazba	X		X	X	X
DefinitionType attribute	Typ definice fronty	X	X			
DefInputOpenOption	Výchozí volba otevření pro vstup	X	X			
DefPersistence	Výchozí trvalost zpráv	X	X	X	X	X
DefPriority	Výchozí priorita zpráv	✓	✓	✓	✓	✓
DefReadAhead	Výchozí dopředné čtení	X	X	X		
DistLists	Podpora seznamu distribuce	X	X			
HardenGetBackout	Zda se má udržovat přesný počet vrácení	X	X			
IndexType	Typ indexu	X	X			
InhibitGet	Zda jsou povoleny operace získání pro frontu	X	X	X		
InhibitPut	Zda jsou povoleny operace vložení pro frontu	X	X	X	X	X
InitiationQName	Název inicializační fronty	X	X			
MaxMsgLength	Maximální délka zprávy v bajtech	X	X			
MaxQDepth	Maximální hloubka fronty	X	X			
MsgDeliverySequence attribute	Pořadí doručení zpráv	X	X			

Tabulka 561. Atributy pro fronty (pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
NonPersistentMessage Class	Cíl spolehlivosti pro dočasné zprávy	X	X			
OpenInputCount	Počet otevření pro vstup	X				
OpenOutputCount	Počet otevření pro výstup	X				
PropertyControl	Řízení vlastností	✓	✓	✓		
ProcessName	Název procesu	X	X			
QDepthHighEvent attribute	Zda se generují události s vysokou hloubkou fronty	X	X			
QDepthHighLimit	Horní limit pro délku fronty	X	X			
QDepthLowEvent attribute	Zda jsou generovány události nízké hloubky fronty	X	X			
QDepthLowLimit attribute	Dolní limit pro hloubku fronty	X	X			
QDepthMaxEvent	Zda jsou generovány události zaplnění fronty	X	X			
QDesc	Popis fronty	X	X	X	X	X
QName	Název fronty	X		X	X	X
QServiceInterval	Cíl pro interval služby fronty	X	X			
QServiceIntervalEvent attribute	Zda jsou generovány události servisního intervalu vysoké nebo servisního intervalu OK	X	X			
QSGDisp attribute	Dispozice skupiny sdílení front	X		X	X	
QueueAccounting	Shromažďování dat evidence front	X	X	X	X	X
QueueMonitoring	Data monitorování online pro fronty	X	✓			
QueueStatistics	Shromažďování statistických dat fronty	X	X	X	X	X
QType	Typ fronty	X		X	X	X
RemoteQMgrName	Název vzdáleného správce front				X	
RemoteQName	Název vzdálené fronty				X	
RetentionInterval	Interval uchování	X	X			
Scope	Zda položka pro frontu existuje také v adresáři buňky	X		X	X	
Shareability	Možnost sdílení fronty	X	X			
StorageClass	Paměťová třída pro frontu	X	X			
TriggerControl	Řízení spouštěče	X	X			
TriggerData	Data spouątěče	X	X			
TriggerDepth	Hloubka spouątěče	X	X			
TriggerMsgPriority	Prahová hodnota priority zpráv pro spouštěče	X	X			
TriggerType	Typ spouątěče	X	X			
Usage attribute	Použití fronty	X	X			
XmitQName	Jméno přenosové fronty				X	

Související pojmy

Fronty klastru

Lokální fronty

Jak vybrat typ přenosové fronty klastru, který se má použít

AlterationDate (MQCHAR12)

Datum, kdy byla definice naposledy změněna.

Tabulka 562. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby byla délka 12 bajtů (například 1992-09-23¬¬, kde ¬¬ představuje dva prázdné znaky).

Hodnoty určitých atributů (například *CurrentQDepth*) se mění při práci správce front. Změny těchto atributů neovlivňují *AlterationDate*.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_DATE s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_DATE_LENGTH.

AlterationTime (MQCHAR8)

Čas, kdy byla definice naposledy změněna.

Tabulka 563. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Toto je čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS používající 24hodinové hodiny, s úvodní nulou, pokud je hodina menší než 10 (například 09.10.20).

- V systému z/OSje čas greenwichský střední čas (GMT), s tím, že systémové hodiny jsou přesně nastaveny na GMT.
- V jiných prostředích je čas místní čas.

Hodnoty určitých atributů (například *CurrentQDepth*) se mění při práci správce front. Změny těchto atributů neovlivňují *AlterationTime*.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_TIME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_TIME_LENGTH.

Název fronty BackoutRequeue(MQCHAR48)

Jedná se o nadměrný název fronty vrácení. Kromě povolení dotazování na její hodnotu neprovede správce front žádnou akci založenou na hodnotě tohoto atributu.

Tabulka 564. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Aplikace spuštěné v produktu WebSphere Application Server a ty, které používají zařízení serveru IBM MQ Application Server, používají tento atribut k určení, kam by měly zprávy, které byly vráceny zpět, jít. Pro všechny ostatní aplikace neprovede správce front žádnou akci na základě hodnoty atributu.

Produkt IBM MQ classes for JMS používá tento atribut k určení, kam se má přenést zpráva, která již byla maximálně vrácena zpět, jak je uvedeno v atributu *BackoutThreshold* .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_BACKOUT_REQ_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

BackoutThreshold (prahová hodnota back-endového systému) (MQLONG)

Jedná se o prahovou hodnotu vrácení. Kromě povolení dotazování na její hodnotu neprovede správce front žádnou akci založenou na hodnotě tohoto atributu.

Tabulka 565. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Aplikace spuštěné uvnitř produktu WebSphere Application Server a ty, které používají zařízení IBM MQ Application Server Facilities, použijí tento atribut k určení, zda má být zpráva vrácena zpět. Pro všechny ostatní aplikace neprovede správce front žádnou akci na základě hodnoty atributu.

Produkt IBM MQ classes for JMS používá tento atribut k určení, kolikrát povolit zálohování zprávy před přenosem zprávy do fronty uvedené atributem *BackoutRequeueQName* .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_BACKOUT_THRESHOLD s voláním MQINQ.

BaseQName (MQCHAR48)

Jedná se o název fronty, která je definována pro lokálního správce front.

Tabulka 566. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
		X		

(Další informace o názvech front viz pole MQOD- ObjectName.) Fronta je jednoho z následujících typů:

MQQT_LOCAL
Lokální fronta.

MQQT_REMOTE
Lokální definice vzdálené fronty.

MQQT_CLUSTER
Fronta klastru.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_BASE_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

BaseType (MQCFIN)

Typ objektu, na který se alias interpretuje.

Tabulka 567. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
		X		

Jedná se o jednu z následujících hodnot:

MQOT_Q
Základní typ objektu je fronta

MQOT_TOPIC
Základní typ objektu je téma

CFStrucName (MQCHAR12)
Jedná se o název struktury prostředku Coupling Facility, v níž jsou uloženy zprávy ve frontě. První znak názvu je v rozsahu A až Z a zbývající znaky jsou v rozsahu A až Z, 0 až 9 nebo prázdné.

Tabulka 568. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Chcete-li získat úplný název struktury v prostředku Coupling Facility, zadejte hodnotu atributu správce front **QSGName** s hodnotou atributu fronty **CFStrucName** .

Tento atribut se vztahuje pouze na sdílené fronty; je ignorován, pokud *QSGDisp* nemá hodnotu MQQSGD_SHARED.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CF_STRUC_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_CF_STRUC_NAME_LENGTH.

z/OS Tento atribut je podporován pouze na systému z/OS.

ClusterChannelNázev (MQCHAR20)
`ClusterChannel` je generický název odesílacích kanálů klastru, které používají tuto frontu jako přenosovou frontu. Atribut uvádí, které odesílací kanály klastru budou z této přenosové fronty klastru posílat zprávy do přijímacího kanálu klastru.

Tabulka 569. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Výchozí konfigurace správce front je určena pro všechny odesílací kanály klastru k odesílání zpráv z jedné přenosové fronty `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. Výchozí konfiguraci lze změnit úpravou atributu správce front, **DefClusterXmitQueueType**. Výchozí hodnota tohoto atributu je `SCTQ`. Tuto hodnotu můžete změnit na `CHANNEL`. Nastavíte-li atribut **DefClusterXmitQueueType** na hodnotu `CHANNEL`, bude každý odesílací kanál klastru standardně používat specifickou přenosovou frontu klastru, `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`*.

Atribut přenosové fronty `ClusterChannelName` můžete také nastavit na odesílací kanál klastru ručně. Zprávy, které jsou určeny pro správce front připojeného prostřednictvím odesílacího kanálu klastru, jsou uloženy do přenosové fronty, která identifikuje odesílací kanál klastru. Tyto zprávy se nebudou ukládat do výchozí přenosové fronty klastru. Pokud nastavíte atribut `ClusterChannelName` na prázdné znaky, přepne se kanál na výchozí přenosovou frontu klastru, jakmile se kanál restartuje. Výchozí fronta je buď `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`*, nebo `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, v závislosti na hodnotě atributu správce front `DefClusterXmitQueueType`.

Zadáním hvězdiček, "`*`", do pole **ClusterChannelName** můžete přidružit přenosovou frontu k sadě odesílacích kanálů klastru. Hvězdička může být na začátku, na konci nebo kdekoli ve středu řetězce názvu klastru. Pole **ClusterChannelName** je omezeno na délku 20 znaků: `MQ_CHANNEL_NAME_LENGTH`.

ClusterName (MQCHAR48)
Jedná se o název klastru, ke kterému fronta patří.

Tabulka 570. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Pokud fronta patří do více než jednoho klastru, *ClusterNamelist* uvádí název objektu seznamu názvů, který identifikuje klastry, a *ClusterName* je prázdné. Alespoň jeden z *ClusterName* a *ClusterNamelist* musí být prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CLUSTER_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_CLUSTER_NAME_LENGTH.

ClusterNamelist (MQCHAR48)

Jedná se o název objektu seznamu názvů, který obsahuje názvy klastrů, do kterých tato fronta patří.

Tabulka 571. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Pokud fronta náleží pouze jednomu klastru, obsahuje objekt seznamu názvů pouze jeden název. Alternativně lze *ClusterName* použít k určení názvu klastru, v takovém případě je *ClusterNamelist* prázdný. Alespoň jeden z *ClusterName* a *ClusterNamelist* musí být prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CLUSTER_NAMELIST s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_NAMELIST_NAME_LENGTH.

CLWLQueuePriority (MQLONG)

Jedná se o prioritu fronty pracovní zátěže klastru, což je hodnota v rozsahu 0 až 9 představující prioritu fronty.

Tabulka 572. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Další informace naleznete v tématu Fronty klastru.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLWL_Q_PRIORITY s voláním MQINQ.

CLWLQueueRank (MQLONG)

Jedná se o ohodnocení důležitosti fronty pracovní zátěže klastru, což je hodnota v rozsahu 0 až 9 reprezentující ohodnocení důležitosti fronty.

Tabulka 573. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Další informace naleznete v tématu Fronty klastru.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLWL_Q_RANK s voláním MQINQ.

CLWLUseQ (MQLONG)

To definuje chování operace MQPUT v případě, že cílová fronta obsahuje lokální instanci i alespoň jednu vzdálenou instanci klastru. Tento atribut se nepoužije v případě, že je zdrojem operace vložení kanál klastru.

Tabulka 574. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Hodnota je jedna z následujících:

MQCLWL_USEQ_ANY
Použijte vzdálené a lokální fronty.

MQCLWL_USEQ_LOCAL
Nepoužívat vzdálené fronty.

MQCLWL_USEQ_AS_Q_MGR
Zdědit definici z MQIA_CLWL_USEQ správce front.

Další informace naleznete v tématu Fronty klastru.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLWL_USEQ s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_CLWL_USEQ_LENGTH.

CreationDate (MQCHAR12)
Toto je datum vytvoření fronty.

Tabulka 575. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby byla délka 12 bajtů (například 2013-09-23¬¬, kde ¬¬ představuje 2 prázdné znaky).

- V systému IBM ise může datum vytvoření fronty lišit od data vytvoření základní entity operačního systému (souboru nebo uživatelského prostoru), která představuje frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CREATION_DATE s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_CREATION_DATE_LENGTH.

CreationTime (MQCHAR8)
Toto je čas, kdy byla fronta vytvořena.

Tabulka 576. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Formát času je HH.MM.SS používající 24hodinové hodiny, s úvodní nulou, pokud je hodina menší než 10 (například 09.10.20).

- V systému z/OSje čas greenwichský střední čas (GMT), s tím, že systémové hodiny jsou přesně nastaveny na GMT.
- V jiných prostředích je čas místní čas.
- V systému IBM ise čas vytvoření fronty může lišit od času vytvoření základní entity operačního systému (souboru nebo uživatelského prostoru), která představuje frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CREATION_TIME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_CREATION_TIME_LENGTH.

CurrentQDepth (MQLONG)
Jedná se o počet zpráv aktuálně uložených ve frontě.

Tabulka 577. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Zvyšuje se během volání MQPUT a během vrácení volání MQGET. Je dekrementován během volání MQGET bez procházení a během odvolání volání MQPUT. Výsledkem je, že počet zahrnuje zprávy, které byly

vloženy do fronty v rámci pracovní jednotky, ale dosud nebyly potvrzeny, i když nejsou vhodné pro načtení pomocí volání MQGET. Podobně vyloučí zprávy, které byly načteny v rámci pracovní jednotky pomocí volání MQGET, ale dosud nebyly potvrzeny.

Tento počet také zahrnuje zprávy, které prošly časem vypršení platnosti, ale dosud nebyly vyřazeny, ačkoli tyto zprávy nejsou vhodné k načtení. Další informace naleznete v tématu Pole MQMD-Vypršení platnosti .

Zpracování jednotky práce a segmentace zpráv mohou způsobit, že *CurrentQDepth* překročí *MaxQDepth*. To však nemá vliv na načítatelnost zpráv. *všechny* zprávy ve frontě lze načíst pomocí volání MQGET běžným způsobem.

Hodnota tohoto atributu kolísá v průběhu činnosti správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CURRENT_Q_DEPTH s voláním MQINQ.

Odezva DefaultPut(MQLONG)

Určuje typ odezvy, která má být použita pro operace vložení do fronty, pokud aplikace určuje MQPMO_RESPONSE_AS_Q_DEF.

Tabulka 578. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	

Jedná se o jednu z následujících hodnot:

MQPRT_SYNC_RESPONSE
Operace vložení je vydána synchronně a vrací odezvu.

MQPRT_ASYNC_RESPONSE
Operace vložení je vydána asynchronně a vrací podmnožinu polí MQMD.

DefBind (MQLONG)

Jedná se o výchozí vazbu, která se používá v případě, že je ve volání MQOPEN určena hodnota MQOO_BIND_AS_Q_DEF a fronta je frontou klastru.

Tabulka 579. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Hodnota je jedna z následujících:

MQBND_BIND_ON_OPEN
Vazba byla opravena voláním MQOPEN.

MQBND_BIND_NOT_FIXED
Vazba není opravena.

MQBND_BIND_ON_GROUP
Umožňuje aplikaci požadovat, aby byla skupina zpráv přidělena ke stejné cílové instanci.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_BIND s voláním MQINQ.

DefinitionType (MQLONG)

Označuje, jak byla fronta definována.

Tabulka 580. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Hodnota je jedna z následujících:

MQQDT_PŘEDDEFINOVANÝ
Fronta je trvalá fronta vytvořená administrátorem systému; odstranit ji může pouze administrátor systému.

Předdefinované fronty jsou vytvořeny pomocí příkazu DEFINE MQSC a lze je odstranit pouze pomocí příkazu DELETE MQSC. Předdefinované fronty nelze vytvořit z modelových front.

Příkazy mohou být vydány buď operátorem, nebo autorizovaným uživatelem odesílajícím zprávu příkazu do vstupní fronty příkazů (další informace viz CommandInputQName).

MQQDT_PERMANENT_DYNAMIC
Fronta je trvalá fronta, která byla vytvořena aplikací vydávající volání MQOPEN s názvem modelové fronty určené v deskriptoru objektu MQOD. Definice modelové fronty měla pro atribut **DefinitionType** hodnotu MQQDT_PERMANENT_DYNAMIC.

Tento typ fronty lze odstranit pomocí volání MQCLOSE. Další informace viz část “MQCLOSE-Zavřít objekt” na stránce 652.

Hodnota atributu **QSGDisp** pro trvalou dynamickou frontu je MQQSGD_Q_MGR.

MQQDT_TEMPORARY_DYNAMIC
Fronta je dočasná fronta, která byla vytvořena aplikací vydávající volání MQOPEN s názvem modelové fronty určené v deskriptoru objektu MQOD. Definice modelové fronty měla pro atribut **DefinitionType** hodnotu MQQDT_TEMPORARY_DYNAMIC.

Tento typ fronty je automaticky odstraněn voláním MQCLOSE, když je zavřen aplikací, která jej vytvořila.

Hodnota atributu **QSGDisp** pro dočasnou dynamickou frontu je MQQSGD_Q_MGR.

MQQDT_SHARED_DYNAMIC
Fronta je sdílená trvalá fronta, která byla vytvořena aplikací vydávající volání MQOPEN s názvem modelové fronty určené v deskriptoru objektu MQOD. Definice modelové fronty měla pro atribut **DefinitionType** hodnotu MQQDT_SHARED_DYNAMIC.

Tento typ fronty lze odstranit pomocí volání MQCLOSE. Další informace viz část “MQCLOSE-Zavřít objekt” na stránce 652.

Hodnota atributu **QSGDisp** pro sdílenou dynamickou frontu je MQQSGD_SHARED.

Tento atribut v definici modelové fronty neoznačuje, jak byla modelová fronta definována, protože modelové fronty jsou vždy předdefinované. Místo toho se hodnota tohoto atributu v modelové frontě používá k určení *DefinitionType* každé dynamické fronty vytvořené z definice modelové fronty pomocí volání MQOPEN.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEFINITION_TYPE s voláním MQINQ.

DefInputOpenOption (MQLONG)
Toto je výchozí způsob, jak otevřít frontu pro vstup.

Tabulka 581. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Použije se, pokud je volba MQOO_INPUT_AS_Q_DEF zadána ve volání MQOPEN při otevření fronty. Hodnota je jedna z následujících:

MQOO_INPUT_EXCLUSIVE
Chcete-li získat zprávy s výlučným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání selže s kódem příčiny MQRC_OBJECT_IN_USE, pokud je fronta aktuálně otevřena touto nebo jinou aplikací pro vstup libovolného typu (MQOOO_INPUT_SHARED nebo MQOO_INPUT_EXCLUSIVE).

MQOO_INPUT_SHARED
Chcete-li získat zprávy se sdíleným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání může být úspěšné, pokud je fronta aktuálně otevřena touto nebo jinou aplikací s MQOO_INPUT_SHARED, ale nezdaří se s kódem příčiny MQRC_OBJECT_IN_USE, pokud je fronta aktuálně otevřena s MQOO_INPUT_EXCLUSIVE.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_INPUT_OPEN_OPTION s voláním MQINQ.

DefPersistence (MQLONG)
Jedná se o výchozí perzistenci zpráv ve frontě. Použije se, pokud je v deskriptoru zprávy při vložení zprávy zadána hodnota MQPER_PERSISTENCE_AS_Q_DEF.

Tabulka 582. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Pokud je v cestě rozpoznání názvu fronty více než jedna definice, je výchozí perzistence převzata z hodnoty tohoto atributu v *první* definici v cestě v době volání MQPUT nebo MQPUT1 . To může být:

- Alias fronty
- Lokální fronta
- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta *DefXmitQName*).

Hodnota je jedna z následujících:

MQPER_PERSISTENT
Zpráva přežije selhání systému a správce front se restartuje. Trvalé zprávy nelze umístit na:

- Dočasné dynamické fronty
- Sdílené fronty, které se mapují na objekt CFSTRUCT na úrovni CFLEVEL (2) nebo níže, nebo kde je objekt CFSTRUCT definován jako RECOVER (NO).

Trvalé zprávy lze umístit do trvalých dynamických front a do předdefinovaných front.

MQPER_NOT_PERSISTENT
Zpráva obvykle nepřežije selhání systému nebo restartování správce front. To platí i v případě, že je během restartu správce front v pomocné paměti nalezena neporušená kopie zprávy.

V případě sdílených front dočasné zprávy *přežijí* restarty správců front ve skupině sdílení front, ale nepřežijí selhání prostředku Coupling Facility používaného pro ukládání zpráv ve sdílených frontách.

Ve stejné frontě mohou existovat trvalé i přechodné zprávy.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_PERSISTENCE s voláním MQINQ.

DefPriority (MQLONG)
Toto je výchozí priorita pro zprávy ve frontě. To platí v případě, že je v deskriptoru zprávy při vložení zprávy do fronty zadána hodnota MQPRI_PRIORITY_AS_Q_DEF.

Tabulka 583. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Pokud je v cestě k rozlišení názvu fronty více než jedna definice, je výchozí priorita pro zprávu převzata z hodnoty tohoto atributu v *první* definici v cestě v době operace vložení. To může být:

- Alias fronty
- Lokální fronta
- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta *DefXmitQName*).

Způsob umístění zprávy do fronty závisí na hodnotě atributu **MsgDeliverySequence** fronty:

- Má-li atribut **MsgDeliverySequence** hodnotu MQMDS_PRIORITY, závisí logická pozice, na které je zpráva umístěna do fronty, na hodnotě pole *Priority* v deskriptoru zprávy.
- Pokud má atribut **MsgDeliverySequence** hodnotu MQMDS_FIFO, jsou zprávy umístěny do fronty, jako by měly prioritu rovnou *DefPriority* vyřešené fronty, bez ohledu na hodnotu pole *Priority* v deskriptoru zprávy. V poli *Priority* je však zachována hodnota určená aplikací, která zprávu vložila. Další informace viz MsgDeliveryMsgDelivery .

Priority jsou v rozsahu nula (nejnižší) až *MaxPriority* (nejvyšší); viz MaxPriority.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_PRIORITY s voláním MQINQ.

DefReadpřed (MQLONG)

Určuje výchozí chování dopředného čtení pro dočasné zprávy doručené klientovi.

Tabulka 584. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X		

Parametr DefReadlze nastavit na jednu z následujících hodnot:

MQREADA_NO
Přechodné zprávy nejsou klientovi odesílány dopředu, dokud je aplikace nepožádají. Pokud klient skončí abnormálně, dojde ke ztrátě maximálně jedné netrvalé zprávy.

MQREADA_YES
Přechodné zprávy jsou klientovi odesílány dopředu před tím, než je aplikace požádá. Dočasné zprávy mohou být ztraceny, pokud klient skončí nestandardně nebo pokud klient nespotřebuje všechny zprávy, které odeslal.

MQREADA_DISABLED
Čtení před dočasnou zprávou není pro tuto frontu povoleno. Zprávy se klientovi neodesílají dopředu bez ohledu na to, zda aplikace klienta požaduje dopředné čtení.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_READ_AHEAD s voláním MQINQ.

DefPResp (MQLONG)

Atribut výchozího typu odezvy vložení (DEFPRESP) definuje hodnotu používanou aplikacemi, když byl typ PutResponsev rámci MQPMO nastaven na MQPMO_RESPONSE_AS_Q_DEF. Tento atribut je platný pro všechny typy front.

Tabulka 585. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Hodnota je jedna z následujících:

SYNC
Operace vložení je vydána synchronně a vrací odezvu.

ASYNC
Operace vložení je vydána asynchronně a vrací podmnožinu polí MQMD.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_PUT_RESPONSE_TYPE s voláním MQINQ.

DistLists (MQLONG)

Označuje, zda lze zprávy rozdělovníku umístit do fronty.

Tabulka 586. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Agent kanálu zpráv (MCA) nastaví atribut tak, aby informoval lokálního správce front, zda správce front na druhém konci kanálu podporuje distribuční seznamy. Tento druhý správce front (nazývaný správce front *partnering*) je ten, který poté, co byl odebrán z lokální přenosové fronty odesílajícím agentem MCA, obdrží zprávu.

Odesílající agent MCA nastaví atribut vždy, když vytvoří připojení k přijímajícímu adaptéru MCA v partnerském správci front. Tímto způsobem může odesílající agent MCA způsobit, že lokální správce front umístí do přenosové fronty pouze zprávy, které může správně zpracovat partnerský správce front.

Tento atribut je primárně určen pro použití s přenosovými frontami, ale popsané zpracování se provádí bez ohledu na využití definované pro frontu (viz Atribut využití).

Hodnota je jedna z následujících:

MQDL_PODPOROVÁNO
Zprávy distribučního seznamu mohou být uloženy ve frontě a přeneseny do partnerských správců front v tomto formuláři. Tím se sníží množství zpracování potřebné k odeslání zprávy do více míst určení.

MQDL_NOT_SUPPORTED
Zprávy distribučního seznamu nelze uložit do fronty, protože partnerský správce front nepodporuje distribuční seznamy. Pokud aplikace vloží zprávu distribučního seznamu a tato zpráva má být umístěna do této fronty, správce front místo toho rozdělí zprávu distribučního seznamu a umístí jednotlivé zprávy do fronty. Tím se zvýší množství zpracování potřebné k odeslání zprávy do více míst určení, ale zajistí se, že zprávy budou správně zpracovány partnerským správcem front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DIST_LISTS s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

Tento atribut není v systému z/OSpodporován.

HardenGetVrátit zpět (MQLONG)

Pro každou zprávu je zachován počet případů, kdy je zpráva načtena voláním MQGET v rámci pracovní jednotky, a tato jednotka práce byla následně odvolána.

Tabulka 587. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Tento počet je k dispozici v poli *BackoutCount* v deskriptoru zprávy po dokončení volání MQGET.

Počet vrácení zprávy přežije restartování správce front. Chcete-li se však ujistit, že je počet přesný, musí být informace *upřesněny* (zaznamenány na disku nebo jiném zařízení trvalého úložiště) pokaždé, když volání MQGET načte zprávu v rámci pracovní jednotky pro tuto frontu. Pokud se tak nestane, správce front selže a volání MQGET se odvolá, počet se může nebo nemusí zvýšit.

Upřesňující informace pro každé volání MQGET v rámci pracovní jednotky však ukládají další náklady na zpracování, takže nastavte atribut **HardenGetBackout** na hodnotu MQQA_BACKOUT_HARDENED pouze v případě, že je nezbytné, aby byl počet přesný.

V systému Multiplatformsje počet vrácení zprávy vždy zpřesněn, bez ohledu na nastavení tohoto atributu.

Možné jsou následující hodnoty:

MQQA_BACKOUT_HARDENED
K zajištění přesnosti počtu vrácení pro zprávy v této frontě se používá zpřesnění.

MQQA_BACKOUT_NOT_HARDENED
Upřesnění se nepoužívá k zajištění přesnosti počtu vrácení pro zprávy v této frontě. Počet proto může být nižší, než by měl být.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_HARDEN_GET_BACKOUT s voláním MQINQ.

IndexType (MQLONG)

Tato volba určuje typ indexu, který správce front udržuje pro zprávy ve frontě.

Tabulka 588. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Požadovaný typ indexu závisí na způsobu, jakým aplikace načítá zprávy, a na tom, zda se jedná o sdílenou frontu nebo nesdílenou frontu (viz atribut QSGDisp). Pro parametr *IndexType*jsou možné následující hodnoty:

MQIT_NONE
Správce front pro tuto frontu neudržuje žádný index. Tuto hodnotu použijte pro fronty, které jsou obvykle zpracovávány sekvenčně, tj. bez použití kritérií výběru ve volání MQGET.

MQIT_MSG_ID
Správce front udržuje index, který používá identifikátory zpráv ve frontě. Tuto hodnotu použijte ve frontách, kde aplikace obvykle načítá zprávy s použitím identifikátoru zprávy jako kritéria výběru pro volání MQGET.

MQIT_CORREL_ID
Správce front udržuje index, který používá identifikátory korelace zpráv ve frontě. Tuto hodnotu použijte pro fronty, ve kterých aplikace obvykle načítá zprávy s použitím identifikátoru korelace jako kritéria výběru pro volání MQGET.

MQIT_MSG_TOKEN

Důležité: Tento typ indexu by se měl používat pouze pro fronty používané s produktem IBM MQ Workflow for z/OS .

Správce front udržuje index, který používá tokeny zpráv ve frontě pro použití s funkcemi správce pracovní zátěže (WLM) produktu z/OS.

Tuto volbu *musíte* zadat pro fronty spravované WLM; nezadávejte ji pro žádný jiný typ fronty. Tuto hodnotu také nepoužívejte pro frontu, ve které aplikace nepoužívá funkce správce pracovní zátěže z/OS , ale načítá zprávy s použitím tokenu zprávy jako kritéria výběru pro volání MQGET.

ID_skupiny_MQIT_ID
Správce front udržuje index, který používá identifikátory skupin zpráv ve frontě. Tato hodnota musí být použita pro fronty, ve kterých aplikace načítá zprávy pomocí volby MQGMO_LOGICAL_ORDER ve volání MQGET.

Fronta s tímto typem indexu nemůže být přenosovou frontou. Sdílená fronta s tímto typem indexu musí být definována pro mapování na objekt CFSTRUCT na úrovni CFLEVEL (3) nebo vyšší.

Poznámka:

1. Fyzické pořadí zpráv ve frontě s typem indexu MQIT_GROUP_ID není definováno, protože fronta je optimalizována pro efektivní načítání zpráv pomocí volby MQGMO_LOGICAL_ORDER ve volání MQGET. To znamená, že fyzické pořadí zpráv obvykle není pořadí, ve kterém zprávy dorazily do fronty.
2. Má-li fronta MQIT_GROUP_ID hodnotu *MsgDeliverySequence* MQMDS_PRIORITY, použije správce front priority zpráv 0 a 1 k optimalizaci načítání zpráv v logickém pořadí. V důsledku toho první zpráva ve skupině nesmí mít prioritu nula nebo jedna; pokud ano, zpráva se zpracuje, jako by měla prioritu dvě. Pole *Priority* ve struktuře MQMD se nezmění.

Další informace o skupinách zpráv naleznete v popisu voleb skupin a segmentů v poli MQGMO-Volby.

Typ indexu, který by měl být použit v různých případech, je uveden v části Tabulka 589 na stránce 852 a Tabulka 590 na stránce 853.

Tabulka 589. Navrhované nebo požadované hodnoty typu indexu fronty, pokud není zadána hodnota MQGMO_LOGICAL_ORDER

Kritéria výběru pro volání MQGET	**Typ indexu pro nesdílenou frontu**	**Typ indexu pro sdílenou frontu**
Není	Libovolný	Libovolný
Výběr pomocí jednoho identifikátoru:		
Identifikátor zprávy	MQIT_MSG_ID navrženo	Je vyžadováno MQIT_NONE nebo MQIT_MSG_ID; navrženo MQIT_MSG_ID
Identifikátor korelace	MQIT_CORREL_ID-navrženo	Je vyžadováno MQIT_CORREL_ID
Identifikátor skupiny	Navrhovaná hodnota MQIT_GROUP_ID	Je vyžadováno MQIT_GROUP_ID
Výběr pomocí dvou identifikátorů:		
Identifikátor zprávy plus identifikátor korelace	Navrhované MQIT_MSG_ID nebo MQIT_CORREL_ID	Je vyžadováno MQIT_NONE nebo MQIT_MSG_ID nebo MQIT_CORREL_ID (Pro efektivitu se doporučuje, aby byl typ indexu vybrán tak, aby odpovídal poli MQMD, které bude mít nejvíce odlišných klíčů)
Identifikátor zprávy plus identifikátor skupiny	Navrženo MQIT_MSG_ID nebo MQIT_GROUP_ID	Nepodporováno
Identifikátor korelace plus identifikátor skupiny	Navrhované MQIT_CORREL_ID nebo MQIT_GROUP_ID	Nepodporováno
Výběr pomocí tří identifikátorů:		
Identifikátor zprávy plus identifikátor korelace plus identifikátor skupiny	Navrhované MQIT_MSG_ID nebo MQIT_CORREL_ID nebo MQIT_GROUP_ID	Nepodporováno
Výběr pomocí kritérií souvisejících se skupinou:		
Identifikátor skupiny plus pořadové číslo zprávy	Je vyžadováno MQIT_GROUP_ID	Je vyžadováno MQIT_GROUP_ID
Pořadové číslo zprávy (musí být 1)	Je vyžadováno MQIT_GROUP_ID	Je vyžadováno MQIT_GROUP_ID

Tabulka 589. Navrhované nebo požadované hodnoty typu indexu fronty, pokud není zadána hodnota MQGMO_LOGICAL_ORDER (pokračování)		
Kritéria výběru pro volání MQGET	**Typ indexu pro nesdílenou frontu**	**Typ indexu pro sdílenou frontu**
Výběr pomocí tokenu zprávy:		
Token zprávy pro použití aplikací	Nepoužívat MQIT_MSG_TOKEN	
Token zprávy pro použití WLM	Požaduje se MQIT_MSG_TOKEN	Nepodporováno

Tabulka 590. Navrhované nebo požadované hodnoty typu indexu fronty při zadání MQGMO_LOGICAL_ORDER		
Kritéria výběru pro volání MQGET	**Typ indexu pro nesdílenou frontu**	**Typ indexu pro sdílenou frontu**
Není	Je vyžadováno MQIT_GROUP_ID	Je vyžadováno MQIT_GROUP_ID
Výběr pomocí jednoho identifikátoru:		
Identifikátor zprávy	Je vyžadováno MQIT_GROUP_ID	Nepodporováno
Identifikátor korelace	Je vyžadováno MQIT_GROUP_ID	Nepodporováno
Identifikátor skupiny	Je vyžadováno MQIT_GROUP_ID	Je vyžadováno MQIT_GROUP_ID
Výběr pomocí dvou identifikátorů:		
Identifikátor zprávy plus identifikátor korelace	Je vyžadováno MQIT_GROUP_ID	Nepodporováno
Identifikátor zprávy plus identifikátor skupiny	Je vyžadováno MQIT_GROUP_ID	Nepodporováno
Identifikátor korelace plus identifikátor skupiny	Je vyžadováno MQIT_GROUP_ID	Nepodporováno
Výběr pomocí tří identifikátorů:		
Identifikátor zprávy plus identifikátor korelace plus identifikátor skupiny	Je vyžadováno MQIT_GROUP_ID	Nepodporováno

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_INDEX_TYPE s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

InhibitGet (MQLONG)

Tato volba určuje, zda jsou povoleny operace získání pro tuto frontu.

Tabulka 591. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X		

Pokud se jedná o alias fronty, operace get musí být povoleny pro alias i základní frontu v době operace získání, aby bylo volání MQGET úspěšné. Hodnota je jedna z následujících:

MQQA_GET_INHIBITED
Operace získání jsou zablokovány.

Volání MQGET se nezdaří s kódem příčiny MQRC_GET_INHIBITED. To zahrnuje volání MQGET, která určují MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT.

Poznámka: Pokud je volání MQGET pracující v rámci pracovní jednotky úspěšně dokončeno, změna hodnoty atributu **InhibitGet** následně na hodnotu MQQA_GET_INHIBITED nezabrání potvrzení transakce.

MQQA_GET_ALLOWED
Operace získání jsou povoleny.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_INHIBITORY s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

InhibitPut (MQLONG)

Tato volba určuje, zda jsou povoleny operace vložení pro tuto frontu.

Tabulka 592. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Pokud je v cestě rozpoznání názvu fronty více než jedna definice, musí být operace vložení povoleny pro *každou* definici v cestě (včetně všech definic aliasů správce front) v době operace vložení, aby bylo volání MQPUT nebo MQPUT1 úspěšné. Hodnota je jedna z následujících:

MQQA_PUT_BLOKOVÁNO
Operace vložení jsou zablokovány.

Volání MQPUT a MQPUT1 se nezdaří s kódem příčiny MQRC_PUT_INHIBITED.

Poznámka: Pokud je volání MQPUT pracující v rámci transakce úspěšně dokončeno, změna hodnoty atributu **InhibitPut** následně na hodnotu MQQA_PUT_INHIBITED nezabrání potvrzení transakce.

MQQA_PUT_POVOLENO
Operace vložení jsou povoleny.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_INHIBITORY put s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

InitiationQName (MQCHAR48)

Jedná se o název fronty definované v lokálním správci front; fronta musí být typu MQQT_LOCAL.

Tabulka 593. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Správce front odešle zprávu spouštěče do inicializační fronty v případě, že je spuštění aplikace vyžadováno v důsledku přijetí zprávy do fronty, do které tento atribut patří. Inicializační fronta musí být monitorována aplikací monitoru spouštěčů, která spustí příslušnou aplikaci po přijetí zprávy spouštěče.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_INITIATION_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

MaxMsgDélka (MQLONG)

Jedná se o horní limit délky nejdelší *fyzické* zprávy, kterou lze umístit do fronty.

Tabulka 594. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Protože však lze atribut fronty **MaxMsgLength** nastavit nezávisle na atributu správce front **MaxMsgLength** , je skutečným horním limitem délky nejdelší fyzické zprávy, kterou lze umístit do fronty, nižší z těchto dvou hodnot.

Pokud správce front podporuje segmentaci, je možné, aby aplikace vložila *logickou* zprávu, která je delší než menší z obou atributů **MaxMsgLength** , ale pouze v případě, že aplikace v MQMD určuje příznak MQMF_SEGMENTATION_ALLOWED. Je-li uveden tento příznak, horní limit pro délku logické zprávy je 999 999 999 bajtů, ale obvykle omezení prostředků vynucené operačním systémem nebo prostředím, ve kterém je aplikace spuštěna, má za následek dolní limit.

Pokus o umístění příliš dlouhé zprávy do fronty selže s jedním z následujících kódů příčiny:

- MQRC_MSG_TOO_BIG_FOR_Q, pokud je zpráva pro frontu příliš velká
- MQRC_MSG_TOO_BIG_FOR_Q_MGR, pokud je zpráva příliš velká pro správce front, ale ne příliš velká pro frontu

Dolní limit pro atribut **MaxMsgLength** je nula; horní limit je 100 MB (104 857 600 bajtů).

Další informace naleznete v tématu Parametr MQPUT- BufferLength.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_MSG_LENGTH s voláním MQINQ.

MaxQDepth (MQLONG)

Toto je definovaný horní limit pro počet fyzických zpráv, které mohou existovat ve frontě v libovolném okamžiku.

Tabulka 595. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Pokus o vložení zprávy do fronty, která již obsahuje zprávy **MaxQDepth** , selže s kódem příčiny MQRC_Q_FULL.

Zpracování jednotky práce a segmentace zpráv mohou způsobit, že skutečný počet fyzických zpráv ve frontě překročí hodnotu **MaxQDepth**. To však nemá vliv na načítatelnost zprávy, protože všechny zprávy ve frontě lze načíst pomocí volání MQGET.

Hodnota tohoto atributu je nula nebo větší. Horní mez je určena prostředím:

- Na následujících platformách nesmí hodnota překročit 999 999 999 999:
  - AIX
  - Linux
  - Windows
  - z/OS
- IBM i V systému IBM inesmí hodnota překročit 640 000.

Poznámka: Úložný prostor, který je k dispozici pro frontu, může být vyčerpán, i když je ve frontě méně než **MaxQDepth** zpráv.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_Q_DEPTH s voláním MQINQ.

Posloupnost MsgDelivery(MQLONG)

Tabulka 596. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Určuje pořadí, ve kterém volání MQGET vrací zprávy aplikaci:

MQMDS_FIFO
Zprávy jsou vráceny v pořadí FIFO (první dovnitř, první ven).

Volání MQGET vrací *první* zprávu, která splňuje kritéria výběru uvedená ve volání, bez ohledu na prioritu zprávy.

MQMDS_PRIORITY
Zprávy jsou vraceny v pořadí podle priority.

Volání MQGET vrací zprávu s *nejvyšší prioritou* , která splňuje kritéria výběru uvedená ve volání. V rámci každé úrovně priority jsou zprávy vraceny v pořadí FIFO (první dovnitř, první ven).

- Pokud má fronta v systému z/OShodnotu *IndexType* MQIT_GROUP_ID, atribut **MsgDeliverySequence** uvádí pořadí, ve kterém jsou skupiny zpráv vráceny aplikaci. Konkrétní posloupnost, ve které jsou skupiny vráceny, je určena pozicí nebo prioritou první zprávy v každé skupině. Fyzické pořadí zpráv ve frontě není definováno, protože fronta je optimalizována pro efektivní načítání zpráv pomocí volby MQGMO_LOGICAL_ORDER ve volání MQGET.
- V systému z/OSplatí, že má-li parametr *IndexType* hodnotu MQIT_GROUP_ID a parametr *MsgDeliverySequence* hodnotu MQMDS_PRIORITY, správce front použije priority zpráv nula a jednu k optimalizaci načítání zpráv v logickém pořadí. V důsledku toho první zpráva ve skupině nesmí mít prioritu nula nebo jedna; pokud ano, zpráva se zpracuje, jako by měla prioritu dvě. Pole *Priority* ve struktuře MQMD se nezmění.

Pokud se příslušné atributy změní, když jsou ve frontě zprávy, pořadí doručení je následující:

- Pořadí, ve kterém jsou zprávy vraceny voláním MQGET, je určeno hodnotami atributů **MsgDeliverySequence** a **DefPriority** platných pro frontu v době, kdy zpráva dorazí do fronty:
  - Má-li parametr *MsgDeliverySequence* při doručení zprávy hodnotu MQMDS_FIFO, bude zpráva umístěna do fronty, jako by její priorita byla *DefPriority*. To nemá vliv na hodnotu pole *Priority* v deskriptoru zprávy; toto pole zachová hodnotu, kterou mělo při prvním vložení zprávy.
  - Má-li parametr *MsgDeliverySequence* při doručení zprávy hodnotu MQMDS_PRIORITY, bude zpráva umístěna do fronty v místě odpovídajícím prioritě dané polem *Priority* v deskriptoru zprávy.

  Pokud se hodnota atributu **MsgDeliverySequence** změní, když jsou ve frontě zprávy, pořadí zpráv ve frontě se nezmění.

  Pokud se hodnota atributu **DefPriority** změní, když jsou ve frontě zprávy, nemusí být zprávy nutně doručeny v pořadí FIFO, i když je atribut **MsgDeliverySequence** nastaven na MQMDS_FIFO; ty, které byly umístěny do fronty s vyšší prioritou, jsou doručeny jako první.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MSG_DELIVERY_SEQUENCE s voláním MQINQ.

NonPersistentMessageClass (MQLONG)
Cíl spolehlivosti pro přechodné zprávy.

Tabulka 597. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Tato volba určuje okolnosti, za kterých jsou přechodné zprávy vkládané do této fronty vyřazeny:

MQNPM_CLASS_NORMAL
Přechodné zprávy jsou omezeny na dobu životnosti relace správce front. Zprávy jsou v případě restartování správce front zrušeny. Tato hodnota je platná pouze pro nesdílené fronty a jedná se o výchozí hodnotu.

MQNPM_CLASS_HIGH
Správce front se pokusí uchovat přechodné zprávy po dobu životnosti fronty. Přechodné zprávy mohou být v případě selhání stále ztraceny. Tato hodnota je vynucena pro sdílené fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_NPM_CLASS s voláním MQINQ.

Počet OpenInput(MQLONG)
Jedná se o počet popisovačů, které jsou aktuálně platné pro odebrání zpráv z fronty pomocí volání MQGET.

Tabulka 598. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Jedná se o celkový počet popisovačů známých *lokálnímu* správci front. Pokud se jedná o sdílenou frontu, počet nezahrnuje otevření pro vstup, který byl proveden pro frontu v jiných správcích front ve skupině sdílení front, do které lokální správce front patří.

Tento počet zahrnuje popisovače, kde byla pro vstup otevřena alias fronta, která se interpretuje na tuto frontu. Počet nezahrnuje popisovače, kde byla fronta otevřena pro akce, které neobsahovaly vstup (například fronta otevřená pouze pro procházení).

Hodnota tohoto atributu kolísá v průběhu činnosti správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_OPEN_INPUT_COUNT s voláním MQINQ.

Počet OpenOutput(MQLONG)
Jedná se o počet popisovačů, které jsou aktuálně platné pro přidávání zpráv do fronty prostřednictvím volání MQPUT.

Tabulka 599. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Jedná se o celkový počet popisovačů známých *lokálnímu* správci front; nezahrnuje otevření pro výstup, která byla provedena pro tuto frontu ve vzdálených správcích front. Pokud se jedná o sdílenou frontu, tento počet nezahrnuje otevření pro výstup, který byl proveden pro frontu v jiných správcích front ve skupině sdílení front, do které lokální správce front patří.

Počet zahrnuje popisovače, kde byla pro výstup otevřena alias fronta, která se interpretuje na tuto frontu. Tento počet nezahrnuje popisovače, kde byla fronta otevřena pro akce, které neobsahovaly výstup (například fronta otevřená pouze pro dotazování).

Hodnota tohoto atributu kolísá v průběhu činnosti správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_OPEN_OUTPUT_COUNT s voláním MQINQ.

ProcessName (MQCHAR48)
Jedná se o název objektu procesu, který je definován v lokálním správci front. Objekt procesu identifikuje program, který může obsluhovat frontu.

Tabulka 600. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_PROCESS_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_PROCESS_NAME_LENGTH.

PropertyControl (MQLONG)

Určuje způsob zpracování vlastností zpráv načtených z front pomocí volání MQGET s volbou MQGMO_PROPERTIES_AS_Q_DEF.

Tabulka 601. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X		

Hodnota je jedna z následujících:

MQPROP_ALL
Všechny vlastnosti zprávy jsou zahrnuty do zprávy, když je doručena do aplikace. Vlastnosti, s výjimkou vlastností obsažených v deskriptoru (či rozšíření) zprávy, budou umístěny v jednom nebo několika záhlavích v datech zprávy. Je-li zadán popisovač zprávy, chování je vrátit vlastnosti v popisovači zprávy.

MQPROP_KOMPATIBILITA
Pokud zpráva obsahuje vlastnost s předponou mcd., jms., usr. v. nebo mqext., všechny vlastnosti zprávy jsou doručeny aplikaci v záhlaví MQRFH2 . Jinak budou všechny vlastnosti zprávy, kromě vlastností obsažených v deskriptoru (či rozšíření) zprávy, zahozeny a nebudou nadále pro aplikaci přístupné. Jedná se o výchozí hodnotu. Umožňuje aplikacím, které očekávají, že se vlastnosti související s produktem JMS budou nacházet v záhlaví MQRFH2 v datech zprávy, aby nadále fungovaly nezměněné. Pokud je zadán popisovač zprávy, chování je vrátit vlastnosti v popisovači zprávy.

MQPROP_FORCE_MQRFH2
Vlastnosti jsou vždy vráceny v datech zprávy v záhlaví MQRFH2, bez ohledu na to, zda aplikace uvádí deskriptor zpráv. Platný popisovač zprávy zadaný v poli MsgHandle struktury MQGMO ve volání MQGET je ignorován. Vlastnosti zprávy nejsou pomocí popisovače zprávy přístupné.

MQPROP_NONE
Všechny vlastnosti zprávy, s výjimkou vlastností nacházejících se v deskriptoru zprávy (či rozšíření), jsou ze zprávy odebrány před jejím doručením do aplikace. Je-li zadán popisovač zprávy, chování je vrátit vlastnosti v popisovači zprávy.

Tento parametr lze použít pro lokální fronty, fronty aliasů a modelové fronty. Chcete-li určit její hodnotu, použijte selektor MQIA_PROPERTY_CONTROL s voláním MQINQ.

Událost QDepthHigh(MQLONG)

Tato volba určuje, zda mají být generovány události Vysoká hloubka fronty.

Tabulka 602. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Událost Vysoká hloubka fronty označuje, že aplikace vložila zprávu do fronty, což způsobilo, že počet zpráv ve frontě byl větší nebo roven horní prahové hodnotě hloubky fronty (viz atribut **QDepthHighLimit**).

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Hodnota je jedna z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_DEPTH_HIGH_EVENT s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení jeho hodnoty.

QDepthHighLimit (MQLONG)

Jedná se o prahovou hodnotu, se kterou je porovnána hloubka fronty při generování události Vysoká hloubka fronty.

Tabulka 603. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Tato událost označuje, že aplikace vložila zprávu do fronty a že to způsobilo, že počet zpráv ve frontě byl větší nebo roven horní prahové hodnotě hloubky fronty. Viz QDepthHigh.

Hodnota je vyjádřena jako procentní část maximální hloubky fronty (atribut**MaxQDepth**) a je větší nebo rovna 0 a menší nebo rovna 100. Výchozí hodnota je 80.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_DEPTH_HIGH_LIMIT s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení jeho hodnoty.

Událost QDepthLow(MQLONG)

Tato volba určuje, zda mají být generovány události dolní hloubky fronty.

Tabulka 604. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Událost Dolní hloubka fronty označuje, že aplikace načetla zprávu z fronty a že to způsobilo, že počet zpráv ve frontě byl menší nebo roven dolní prahové hodnotě hloubky fronty (viz QDepthLowLimit).

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Hodnota je jedna z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_DEPTH_LOW_EVENT s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení jeho hodnoty.

QDepthLowLimit (MQLONG)

Jedná se o prahovou hodnotu, se kterou je porovnána hloubka fronty při generování události dolní hloubky fronty.

Tabulka 605. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Tato událost označuje, že aplikace načetla zprávu z fronty a že to způsobilo, že počet zpráv ve frontě byl menší nebo roven dolní prahové hodnotě hloubky fronty. Viz QDepthLow.

Hodnota je vyjádřena jako procentní část maximální hloubky fronty (atribut**MaxQDepth**) a je větší nebo rovna 0 a menší nebo rovna 100. Výchozí hodnota je 20.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_DEPTH_LOW_LIMIT s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení jeho hodnoty.

Událost QDepthMax(MQLONG)

Tato volba určuje, zda mají být generovány události zaplnění fronty. Událost zaplnění fronty označuje, že vložení do fronty bylo odmítnuto, protože fronta je plná, to znamená, že hloubka fronty již dosáhla své maximální hodnoty.

Tabulka 606. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Hodnota je jedna z následujících:

MQEVR_DISABLED
Hlášení událostí zakázáno.

MQEVR_ENABLED
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_DEPTH_MAX_EVENT s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení jeho hodnoty.

QDesc (MQCHAR64)

Použijte toto pole pro popisný komentář.

Tabulka 607. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Obsah pole nemá pro správce front žádný význam, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou obsaženy ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_Q_DESC s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_DESC_LENGTH.

Název fronty (MQCHAR48)

Jedná se o název fronty definované v lokálním správci front.

Tabulka 608. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Všechny fronty definované ve správci front sdílejí stejný obor názvů fronty. Fronta MQQT_LOCAL a fronta MQQT_ALIAS proto nemohou mít stejný název.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

QServiceInterval (MQLONG)

Jedná se o interval služby použitý pro porovnání pro generování událostí servisního intervalu vysoké a servisního intervalu OK.

Tabulka 609. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Viz QServiceInterval.

Hodnota je v jednotkách milisekund a je větší nebo rovna nule a menší nebo rovna 999 999 999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_SERVICE_INTERVAL s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení jeho hodnoty.

Událost QServiceInterval(MQLONG)

Tato volba určuje, zda mají být generovány události Service Interval High nebo Service Interval OK.

Tabulka 610. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

- Událost vysokého servisního intervalu je generována, když kontrola označuje, že z fronty nebyly načteny žádné zprávy alespoň po dobu označenou atributem **`QServiceInterval`** .
- Událost OK servisního intervalu je generována, když kontrola označuje, že zprávy byly načteny z fronty v čase označeném atributem **`QServiceInterval`** .

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Hodnota je jedna z následujících:

MQQSIE_HIGH
Události vysokého servisního intervalu fronty jsou povoleny.

- Události vysokého servisního intervalu fronty jsou **povoleny** a
- Události OK servisního intervalu fronty jsou **zakázány**.

MQQSIE_OK
Události OK intervalu služby fronty jsou povoleny.

- Události vysokého servisního intervalu fronty jsou **zakázány** a
- Události OK intervalu služby fronty jsou **povoleny**.

MQQSIE_NONE
Nejsou povoleny žádné události intervalu služby fronty.

- Události vysokého servisního intervalu fronty jsou **zakázány** a
- Události OK intervalu služby fronty jsou také **zakázány**.

Pro sdílené fronty je hodnota tohoto atributu ignorována; předpokládá se hodnota MQQSIE_NONE.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_SERVICE_INTERVAL_EVENT s voláním MQINQ.

V systému z/OSnemůžete použít volání MQINQ k určení hodnoty tohoto atributu.

QSGDisp (MQLONG)

Tato volba určuje dispozici fronty.

Tabulka 611. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Hodnota je jedna z následujících:

MQQSGD_Q_MGR
Objekt má dispozici správce front. To znamená, že definice objektu je známa pouze lokálnímu správci front; definice není známa ostatním správcům front ve skupině sdílení front.

Každý správce front ve skupině sdílení front může mít objekt se stejným názvem a typem jako aktuální objekt, ale jedná se o samostatné objekty a neexistuje mezi nimi žádná korelace. Jejich atributy nejsou omezeny tak, aby byly stejné jako ostatní.

MQQSGD_COPY
Objekt je lokální kopií definice hlavního objektu, která existuje ve sdíleném úložišti. Každý správce front ve skupině sdílení front může mít svou vlastní kopii objektu. Na začátku mají všechny kopie stejné atributy, ale pomocí příkazů MQSC můžete každou kopii změnit tak, aby se její atributy lišily od atributů ostatních kopií. Atributy kopií se znovu synchronizují, když se změní hlavní definice ve sdíleném úložišti.

MQQSGD_SHARED
Objekt má sdílenou dispozici. To znamená, že ve sdíleném úložišti existuje jediná instance objektu, která je známa všem správcům front ve skupině sdílení front. Když správce front ve skupině přistupuje k objektu, přistupuje k jediné sdílené instanci objektu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_QSG_DISP s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

QueueAccounting (MQLONG)

Tabulka 612. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	

Tato volba řídí shromažďování dat evidence pro frontu. Chcete-li shromažďovat data evidence pro tuto frontu, musí být povolena také data evidence pro toto připojení pomocí atributu QMGR ACCTQ nebo pole Volby ve struktuře MQCNO ve volání MQCONNX.

Tento atribut může mít některou z následujících hodnot:

MQMON_Q_MGR
Data evidence pro tuto frontu se shromažďují na základě nastavení atributu ACCTQ QMGR. Toto je výchozí nastavení.

MQMON_OFF
Neshromažďovat data evidence pro tuto frontu.

MQMON_ON
Shromážděte data evidence pro tuto frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACCOUNTING_Q s voláním MQINQ.

QueueMonitoring (MQLONG)

Ovládá shromažďování online monitorovacích dat pro fronty.

Tabulka 613. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Hodnota je jedna z následujících:

MQMON_Q_MGR
Shromážděte data monitorování podle nastavení atributu správce front **QueueMonitoring** . Toto je výchozí hodnota.

MQMON_OFF
Shromažďování dat monitorování online je pro tuto frontu vypnuto.

MQMON_LOW
Pokud hodnota atributu správce front **QueueMonitoring** není MQMON_NONE, zapne se shromažďování dat monitorování online s nízkou rychlostí shromažďování dat pro tuto frontu.

MQMON_MEDIUM
Pokud hodnota atributu správce front **QueueMonitoring** není MQMON_NONE, zapne se shromažďování dat monitorování online se střední rychlostí shromažďování dat pro tuto frontu.

MQMON_HIGH-vysoká
Pokud hodnota atributu správce front **QueueMonitoring** není MQMON_NONE, zapne se shromažďování dat monitorování online s vysokou rychlostí shromažďování dat pro tuto frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MONITORING_Q s voláním MQINQ.

QueueStatistics (MQCHAR12)

Tabulka 614. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X	X	

Tato volba řídí shromažďování statistických dat pro frontu.

Tento atribut může mít některou z následujících hodnot:

MQMON_Q_MGR
Data evidence pro tuto frontu jsou shromažďována na základě nastavení atributu QMGR STATQ. Toto je výchozí nastavení.

MQMON_OFF
Vypněte shromažďování statistických dat pro tuto frontu.

MQMON_ON
Povolit shromažďování statistických dat pro tuto frontu.

QType (MQLONG)

Tabulka 615. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	X

Toto je typ fronty; má jednu z následujících hodnot:

MQQT_ALIAS
Definice alias fronty.

MQQT_CLUSTER
Fronta klastru.

MQQT_LOCAL
Lokální fronta.

MQQT_REMOTE
Lokální definice vzdálené fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_TYPE s voláním MQINQ.

RemoteQMgrNázev (MQCHAR48)

Tabulka 616. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
			X	

Jedná se o název vzdáleného správce front, ve kterém je fronta **RemoteQName** definována. Pokud má fronta **RemoteQName** hodnotu **QSGDisp** MQQSGD_COPY nebo MQQSGD_SHARED, **RemoteQMgrName** může být název skupiny sdílení front, která vlastní **RemoteQName**.

Pokud aplikace otevře lokální definici vzdálené fronty, **RemoteQMgrName** nesmí být prázdné a nesmí být název lokálního správce front. Je-li hodnota **XmitQName** prázdná, použije se jako přenosová fronta lokální fronta se stejným názvem jako **RemoteQMgrName** . Pokud neexistuje žádná fronta s názvem **RemoteQMgrName**, použije se fronta určená atributem správce front **DefXmitQName** .

Pokud je tato definice použita pro alias správce front, **RemoteQMgrName** je název správce front, který je aliasem. Může se jednat o název lokálního správce front. Jinak, pokud je **XmitQName** při otevření prázdné, musí existovat lokální fronta s názvem, který je stejný jako **RemoteQMgrName**; tato fronta se používá jako přenosová fronta.

Pokud je tato definice použita pro alias odpovědi, je tento název názvem správce front, který má být **ReplyToQMgr**.

Poznámka: Při vytváření nebo úpravě definice fronty se na hodnotě uvedené pro tento atribut neprovede žádné ověření.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_REMOTE_Q_MGR_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

RemoteQName (MQCHAR48)

Tabulka 617. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
			X	

Jedná se o název fronty tak, jak je znám ve vzdáleném správci front *RemoteQMgrName*.

Pokud aplikace otevře lokální definici vzdálené fronty, nesmí být hodnota *RemoteQName* při otevření prázdná.

Je-li tato definice použita pro definici aliasu správce front, musí být hodnota *RemoteQName* při otevření prázdná.

Pokud je definice použita pro alias odpovědi, tento název je název fronty, která má být *ReplyToQ*.

Poznámka: Při vytváření nebo úpravě definice fronty se na hodnotě uvedené pro tento atribut neprovede žádné ověření.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_REMOTE_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

RetentionInterval (MQLONG)

Jedná se o časové období, po které se má fronta uchovat. Po uplynutí této doby je fronta vhodná pro odstranění.

Tabulka 618. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Čas se měří v hodinách, počítáno od data a času, kdy byla fronta vytvořena. Datum a čas vytvoření fronty jsou zaznamenány v atributech **`CreationDate`** a **`CreationTime`** .

Tyto informace jsou poskytnuty, aby umožnily úklidové aplikaci nebo operátorovi identifikovat a odstranit fronty, které již nejsou požadovány.

Poznámka: Správce front nikdy neprovede žádnou akci k odstranění front na základě tohoto atributu nebo k zabránění odstranění front s intervalem uchování, jehož platnost dosud nevypršela. Je zodpovědností uživatele provést jakoukoli požadovanou akci.

Chcete-li zabránit hromadění trvalých dynamických front, použijte realistický interval uchování (viz DefinitionType). Tento atribut však lze také použít s předdefinovanými frontami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_RETENTION_INTERVAL s voláním MQINQ.

Rozsah (MQLONG)

Tato volba určuje, zda položka pro tuto frontu existuje také v adresáři buňky.

Tabulka 619. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Adresář buňky je poskytován instalovatelnou službou názvů. Hodnota je jedna z následujících:

MQSCO_Q_MGR
Definice fronty má obor správce front: definice fronty nepřekračuje rámec správce front, který ji vlastní. Chcete-li otevřít frontu pro výstup z jiného správce front, musí být zadán buď název vlastnícího správce front, nebo jiný správce front musí mít lokální definici fronty.

MQSCO_CELL
Definice fronty má obor buňky: definice fronty je také umístěna do adresáře buňky, který je k dispozici všem správcům front v buňce. Frontu lze otevřít pro výstup z libovolného správce front v buňce zadáním názvu fronty. Není třeba zadávat název správce front, který frontu vlastní. Definice fronty však není k dispozici žádnému správci front v buňce, která má také lokální definici fronty s tímto názvem, protože lokální definice má přednost.

Adresář buňky je poskytován instalovatelnou službou názvů.

Modelové a dynamické fronty nemohou mít rozsah buňky.

Tato hodnota je platná pouze v případě, že byla konfigurována služba názvů podporující adresář buňky.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SCOPE s voláním MQINQ.

Podpora tohoto atributu podléhá následujícím omezením:

- V systému IBM ije atribut podporován, ale pouze MQSCO_Q_MGR je platný.
- V systému z/OSnení atribut podporován.

Možnost sdílení (MQLONG)
Označuje, zda lze frontu otevřít pro vstup vícekrát souběžně.

Tabulka 620. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Hodnota je jedna z následujících:

MQQA_SHAREABLE
Frontu lze sdílet.

Je povoleno více otevření s volbou MQOO_INPUT_SHARED.

MQQA_NOT_SHAREABLE
Frontu nelze sdílet.

Volání MQOPEN s volbou MQOOO_INPUT_SHARED je považováno za volání MQOO_INPUT_EXCLUSIVE.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SHAREABILITY s voláním MQINQ.

StorageClass (MQCHAR8)
Jedná se o název definovaný uživatelem, který definuje fyzické úložiště používané k zadržení fronty. V praxi se zpráva zapíše na disk pouze v případě, že je třeba ji odstránkovat z vyrovnávací paměti.

Tabulka 621. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_STORAGE_CLASS s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_STORAGE_CLASS_LENGTH.

z/OS Tento atribut je podporován pouze na systému z/OS.

TriggerControl (MQLONG)
Tato volba určuje, zda jsou zprávy spouštěče zapisovány do inicializační fronty za účelem spuštění aplikace pro obsluhu fronty.

Tabulka 622. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o jednu z následujících možností:

MQTC_OFF
Pro tuto frontu se nemají zapisovat žádné zprávy spouštěče. Hodnota *TriggerType* je v tomto případě irelevantní.

MQTC_ON
Zprávy spouštěče se zapíší pro tuto frontu, když dojde k příslušným událostem spouštěče.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TRIGGER_CONTROL s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

TriggerData (MQCHAR64)
Jedná se o data ve volném formátu, která správce front vloží do zprávy spouštěče, když zpráva přicházející do této fronty způsobí zápis zprávy spouštěče do inicializační fronty.

Tabulka 623. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Obsah těchto dat nemá pro správce front žádný význam. Má význam buď pro aplikaci monitoru spouštěčů, která zpracovává inicializační frontu, nebo pro aplikaci, kterou monitor spouštěčů spouští.

Znakový řetězec nesmí obsahovat žádné hodnoty null. V případě potřeby je vyplněna vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_TRIGGER_DATA s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET. Délka tohoto atributu je dána hodnotou MQ_TRIGGER_DATA_LENGTH.

TriggerDepth (MQLONG)

Tabulka 624. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o počet zpráv s prioritou *TriggerMsgPriority* nebo vyšší, které musí být ve frontě, než se zapíše zpráva spouštěče. To platí v případě, že je parametr *TriggerType* nastaven na hodnotu MQTT_DEPTH. Hodnota *TriggerDepth* je jedna nebo větší. Tento atribut není jinak použit.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TRIGGER_DEPTH s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

Priorita TriggerMsg(MQLONG)

Toto je priorita zpráv, pod kterou zprávy nepřispívají ke generování zpráv spouštěče (to znamená, že správce front tyto zprávy ignoruje při rozhodování, zda generovat zprávu spouštěče).

Tabulka 625. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Hodnota *TriggerMsgPriority* může být v rozsahu nula (nejnižší) až *MaxPriority* (nejvyšší; viz MaxPriority); hodnota nula způsobí, že všechny zprávy přispějí ke generování zpráv spouštěče.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TRIGGER_MSG_PRIORITY s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

TriggerType (MQLONG)

To řídí podmínky, za kterých jsou zprávy spouštěče zapisovány jako výsledek zpráv přicházejících do této fronty.

Tabulka 626. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Má jednu z následujících hodnot:

MQTT_NONE
V důsledku zpráv v této frontě nejsou zapsány žádné zprávy spouštěče. To má stejný efekt jako nastavení parametru *TriggerControl* na hodnotu MQTC_OFF.

MQTT_FIRST
Zpráva spouštěče se zapíše vždy, když se počet zpráv s prioritou *TriggerMsgPriority* nebo vyšší ve frontě změní z 0 na 1.

MQTT_EVERY
Zpráva spouštěče se zapíše vždy, když do fronty dorazí zpráva s prioritou *TriggerMsgPriority* nebo vyšší.

Hloubka MQTT_DEPTH
Zpráva spouštěče se zapíše vždy, když se počet zpráv s prioritou *TriggerMsgPriority* nebo vyšší ve frontě rovná nebo překročí hodnotu *TriggerDepth*. Po zápisu zprávy spouštěče je parametr *TriggerControl* nastaven na hodnotu MQTC_OFF, aby se zabránilo dalšímu spouštění, dokud nebude explicitně znovu zapnut.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TRIGGER_TYPE s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

Použití (MQLONG)
Označuje, k čemu se fronta používá.

Tabulka 627. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Hodnota je jedna z následujících:

MQUS_NORMAL
Jedná se o frontu, kterou aplikace používají při vkládání a získávání zpráv. Fronta není přenosovou frontou.

MQUS_PŘENOS
Jedná se o frontu používanou k ukládání zpráv určených pro vzdálené správce front. Když aplikace odešle zprávu do vzdálené fronty, lokální správce front zprávu dočasně uloží do příslušné přenosové fronty ve speciálním formátu. Agent kanálu zpráv poté přečte zprávu z přenosové fronty a přenese ji do vzdáleného správce front. Další informace o konfiguraci vzdálené administrace naleznete v tématu Konfigurace správců front pro vzdálenou administraci.

Pouze privilegované aplikace mohou otevřít přenosovou frontu pro MQOO_OUTPUT a přímo do ní vkládat zprávy. Obvykle to dělají pouze obslužné aplikace. Ujistěte se, že je datový formát zprávy správný (viz “MQXQH-záhlaví přenosové fronty” na stránce 621). nebo se mohou vyskytnout chyby během procesu přenosu. Kontext není předán nebo nastaven, pokud není zadána jedna z voleb kontextu MQPMO_ * _CONTEXT.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_USAGE s voláním MQINQ.

XmitQName (MQCHAR48)
Toto je název přenosové fronty. Je-li tento atribut neprázdný, když dojde k otevření, buď pro vzdálenou frontu, nebo pro definici aliasu správce front, uvádí název lokální přenosové fronty, která se má použít pro postoupení zprávy.

Tabulka 628. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
			X	

Pokud je hodnota **XmitQName** prázdná, jako přenosová fronta se použije lokální fronta s názvem, který je stejný jako **RemoteQMgrName** . Pokud neexistuje žádná fronta s názvem **RemoteQMgrName**, použije se fronta určená atributem správce front **DefXmitQName** .

Tento atribut je ignorován, pokud se definice používá jako alias správce front a **RemoteQMgrName** je název lokálního správce front. Také se ignoruje tehdy, jestliže se definice pouľívá jako definice alias odpovídací fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_XMIT_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

Atributy pro seznamy názvů

Následující tabulka shrnuje atributy, které jsou specifické pro seznamy názvů. Atributy jsou popsány v abecedním pořadí.

Seznamy názvů jsou podporovány na všech systémech IBM MQ plus IBM MQ MQI clients připojených k těmto systémům.

Poznámka: Názvy atributů zobrazené v této sekci jsou popisné názvy používané s voláními MQINQ a MQSET; názvy jsou stejné jako pro příkazy PCF. Při použití příkazů MQSC k definování, změně nebo zobrazení atributů jsou použity alternativní krátké názvy. Další informace naleznete v tématu Příkazy MQSC .

Tabulka 629. Atributy pro seznamy názvů

Atribut	Popis
AlterationDate	Datum, kdy byla definice naposledy změněna
AlterationTime	Čas, kdy byla definice naposledy změněna
NameCount	Počet názvů v seznamu názvů
NamelistDesc	Popis seznamu názvů
NamelistName	Název seznamu názvů
Názvy	Seznam názvů *NameCount*
NamelistType	Typ seznamu názvů
QSGDisp	Dispozice skupiny sdílení front

AlterationDate (MQCHAR12)

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby se délka 12 bajtů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_DATE s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_DATE_LENGTH.

AlterationTime (MQCHAR8)

Toto je čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_TIME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_TIME_LENGTH.

NameCount (MQLONG)

Označuje počet názvů v seznamu názvů. Je větší než nebo rovno nule. Je definována následující hodnota:

MQNC_MAX_NAMELIST_NAME_COUNT
Maximální počet názvů v seznamu názvů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_NAME_COUNT s voláním MQINQ.

NamelistDesc (MQCHAR64)

Toto pole použijte pro popisný komentář; jeho hodnota je stanovena definičním procesem. Obsah pole nemá pro správce front žádný význam, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může toto pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou obsaženy ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_NAMELIST_DESC s voláním MQINQ.

Délka tohoto atributu je dána hodnotou MQ_NAMELIST_DESC_LENGTH.

NamelistName (MQCHAR48)

Jedná se o název seznamu názvů, který je definován v lokálním správci front. Další informace o názvech seznamů názvů naleznete v části Jiné názvy objektů .

Každý seznam názvů má název, který se liší od názvů jiných seznamů názvů náležejících správci front, ale může duplikovat názvy jiných objektů správců front různých typů (například front).

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_NAMELIST_NAME s voláním MQINQ.

Délka tohoto atributu je dána hodnotou MQ_NAMELIST_NAME_LENGTH.

NamelistType (MQLONG)

Tato volba určuje povahu názvů v seznamu názvů a určuje způsob použití seznamu názvů. Jedná se o jednu z následujících hodnot:

MQNT_NONE
Seznam názvů bez přiřazeného typu.

MQNT_Q
Seznam názvů obsahující názvy front.

MQNT_CLUSTER
Seznam názvů obsahující názvy klastrů.

MQNT_AUTH_INFO
Seznam názvů obsahující názvy objektů ověřovacích informací.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_NAMELIST_TYPE s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

Názvy (MQCHAR48xNameCount)

Toto je seznam názvů *NameCount* , kde každý název je názvem objektu, který je definován pro lokálního správce front. Další informace o názvech objektů viz Pravidla pro pojmenování IBM MQ objektů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_NAMES s voláním MQINQ.

Délka každého názvu v seznamu je dána hodnotou MQ_OBJECT_NAME_LENGTH.

QSGDisp (MQLONG)

Tato volba určuje dispozici seznamu názvů. Hodnota je jedna z následujících:

MQQSGD_Q_MGR
Objekt má dispozici správce front: definice objektu je známa pouze lokálnímu správci front; definice není známa ostatním správcům front ve skupině sdílení front.

Každý správce front ve skupině sdílení front může mít objekt se stejným názvem a typem jako aktuální objekt, ale jedná se o samostatné objekty a neexistuje mezi nimi žádná korelace. Jejich atributy nejsou omezeny tak, aby byly stejné jako ostatní.

MQQSGD_COPY
Objekt je lokální kopií definice hlavního objektu, která existuje ve sdíleném úložišti. Každý správce front ve skupině sdílení front může mít svou vlastní kopii objektu. Na začátku mají všechny kopie stejné atributy, ale pomocí příkazů MQSC můžete každou kopii změnit tak, aby se její atributy lišily od ostatních kopií. Atributy kopií se znovu synchronizují, když se změní hlavní definice ve sdíleném úložišti.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_QSG_DISP s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

Atributy pro definice procesu

Následující tabulka shrnuje atributy, které jsou specifické pro definice procesu. Atributy jsou popsány v abecedním pořadí.

Poznámka: Názvy atributů v této sekci jsou popisné názvy používané s voláními MQINQ a MQSET; názvy jsou stejné jako pro příkazy PCF. Při použití příkazů MQSC k definování, změně nebo zobrazení atributů jsou použity alternativní krátké názvy. Další informace naleznete v tématu Příkazy MQSC .

Tabulka 630. Atributy pro definice procesu

Atribut	Popis
AlterationDate	Datum, kdy byla definice naposledy změněna
AlterationTime	Čas, kdy byla definice naposledy změněna
ApplId	Identifikátor aplikace
ApplType	Typ aplikace
EnvData	Data prostředí
ProcessDesc	Popis procesu
ProcessName	Název procesu
QSGDisp	Dispozice skupiny sdílení front
UserData	Data uživatele

AlterationDate (MQCHAR12)

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby se délka 12 bajtů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_DATE s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_DATE_LENGTH.

AlterationTime (MQCHAR8)

Toto je čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_TIME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_TIME_LENGTH.

ApplId (MQCHAR256)

Jedná se o znakový řetězec, který identifikuje aplikaci, která má být spuštěna. Tyto informace jsou určeny pro použití aplikací monitoru spouštěčů, která zpracovává zprávy v inicializační frontě. Informace se odesílají do inicializační fronty jako součást zprávy spouštěče.

Význam parametru *ApplId* je určen aplikací pro monitorování spouštěčů. Monitor spouštěčů poskytovaný produktem IBM MQ vyžaduje, aby *ApplId* byl název spustitelného programu. Následující poznámky platí pro uvedená prostředí:

- V systému z/OSmusí být *ApplId* :
  - Identifikátor transakce CICS pro aplikace spuštěné pomocí CKTI transakce monitoru spouštěčů CICS .
  - Identifikátor transakce IMS pro aplikace spuštěné pomocí monitoru spouštěčů IMS CSQQTRMN
- V systému Windowsmůže mít název programu předponu s cestou k jednotce a adresáři.
- V systému AIX and Linuxmůže mít název programu předponu s cestou k adresáři.

Znakový řetězec nesmí obsahovat žádné hodnoty null. V případě potřeby je vyplněna vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_APPL_ID s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_PROCESS_APPL_ID_LENGTH.

ApplType (MQLONG)

Označuje povahu programu, který se má spustit jako odpověď na přijetí zprávy spouštěče. Tyto informace jsou určeny pro použití aplikací monitoru spouštěčů, která zpracovává zprávy v inicializační frontě. Informace se odesílají do inicializační fronty jako součást zprávy spouštěče.

ApplType může mít libovolnou hodnotu, ale pro standardní typy se doporučují následující hodnoty; omezte typy aplikací definované uživatelem na hodnoty v rozsahu MQAT_USER_FIRST až MQAT_USER_LAST:

MQAT_AIX-operační systém
Aplikace AIX (stejná hodnota jako MQAT_UNIX).

MQAT_BATCH
aplikace pro dávkové úlohy

MQAT_CICS
CICS transakce.

MQAT_IMS
IMS .

MQAT_IMS_BRIDGE
Aplikace mostu IMS .

MQAT_JAVA
Java .

MQAT_MVS
Aplikace MVS nebo TSO (stejná hodnota jako MQAT_ZOS).

MQAT_OS390
Aplikace OS/390 (stejná hodnota jako MQAT_ZOS).

MQAT_OS400
IBM i .

MQAT_UNIX
UNIX .

MQAT_UNKNOWN (neznámý)
Aplikace neznámého typu.

MQAT_USER
Uživatelská aplikace.

MQAT_WINDOWS
64bitová aplikace Windows .

MQAT_WINDOWS_NT
32bitová aplikace Windows .

MQAT_WLM
z/OS aplikace správce pracovní zátěže.

MQAT_ZOS
z/OS .

MQAT_USER_FIRST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

MQAT_USER_LAST
Nejvyšší hodnota pro typ aplikace definovaný uživatelem.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_APPL_TYPE s voláním MQINQ.

EnvData (MQCHAR128)

Jedná se o znakový řetězec, který obsahuje informace související s prostředím týkající se aplikace, která má být spuštěna. Tyto informace jsou určeny pro použití aplikací monitoru spouštěčů, která zpracovává zprávy v inicializační frontě. Informace se odesílají do inicializační fronty jako součást zprávy spouštěče.

Význam parametru *EnvData* je určen aplikací pro monitorování spouštěčů. Monitor spouštěčů poskytovaný pomocí IBM MQ připojení *EnvData* k seznamu parametrů předaný spuštěné aplikaci. Seznam parametrů se skládá ze struktury MQTMC2 , následované jednou mezerou, následovanou *EnvData* s odebranými koncovými mezerami. Následující poznámky platí pro uvedená prostředí:

- V systému z/OS:
  - Produkt *EnvData* není používán aplikacemi monitoru spouštěčů poskytovanými produktem IBM MQ.
  - Je-li ApplType MQAT_WLM, můžete zadat výchozí hodnoty do polí EnvData pro ServiceName a ServiceStep v záhlaví pracovních informací (MQWIH).
- V systému AIX and Linuxlze *EnvData* nastavit na znak & , aby se spuštěná aplikace spustila na pozadí.

Znakový řetězec nesmí obsahovat žádné hodnoty null. V případě potřeby je vyplněna vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ENV_DATA s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_PROCESS_ENV_DATA_LENGTH.

ProcessDesc (MQCHAR64)

Použijte toto pole pro popisný komentář. Obsah pole nemá pro správce front žádný význam, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou obsaženy ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_PROCESS_DESC s voláním MQINQ.

Délka tohoto atributu je dána hodnotou MQ_PROCESS_DESC_LENGTH.

ProcessName (MQCHAR48)

Jedná se o název definice procesu, která je definována v lokálním správci front.

Každá definice procesu má název, který se liší od názvů ostatních definic procesů náležejících správci front. Název definice procesu však může být stejný jako názvy jiných objektů správce front různých typů (například front).

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_PROCESS_NAME s voláním MQINQ.

Délka tohoto atributu je dána hodnotou MQ_PROCESS_NAME_LENGTH.

QSGDisp (MQLONG)

Tato volba určuje dispozici definice procesu. Hodnota je jedna z následujících:

MQQSGD_Q_MGR
Objekt má dispozici správce front: definice objektu je známa pouze lokálnímu správci front; definice není známa ostatním správcům front ve skupině sdílení front.

Každý správce front ve skupině sdílení front může mít objekt se stejným názvem a typem jako aktuální objekt, ale jedná se o samostatné objekty a neexistuje mezi nimi žádná korelace. Jejich atributy nejsou omezeny tak, aby byly stejné jako ostatní.

MQQSGD_COPY
Objekt je lokální kopií definice hlavního objektu, která existuje ve sdíleném úložišti. Každý správce front ve skupině sdílení front může mít svou vlastní kopii objektu. Na začátku mají všechny kopie stejné atributy, ale pomocí příkazů MQSC můžete každou kopii změnit tak, aby se její atributy lišily

od ostatních kopií. Atributy kopií se znovu synchronizují, když se změní hlavní definice ve sdíleném úložišti.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_QSG_DISP s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

UserData (MQCHAR128)

UserData je znakový řetězec, který obsahuje informace o uživateli týkající se aplikace, která má být spuštěna. Tyto informace jsou určeny pro použití aplikací monitoru spouštěčů, která zpracovává zprávy v inicializační frontě, nebo aplikací, která je spuštěna monitorem spouštěčů. Informace se odešlou do inicializační fronty jako součást zprávy spouštěče.

Význam parametru *UserData* je určen aplikací pro monitorování spouštěčů. Monitor spouštěčů poskytovaný produktem IBM MQ předává *UserData* spuštěné aplikaci jako součást seznamu parametrů. Seznam parametrů se skládá ze struktury MQTMC2 (obsahující *UserData*), následované jednou mezerou, následovanou *EnvData* s odebranými koncovými mezerami.

Znakový řetězec nesmí obsahovat žádné hodnoty null. V případě potřeby je vyplněna vpravo mezerami. V případě produktu Microsoft Windowsnesmí znakový řetězec obsahovat dvojité uvozovky, pokud má být definice procesu předána do **runmqtrm**.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_USER_DATA s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_PROCESS_USER_DATA_LENGTH.

Návratové kódy

Pro každé volání rozhraní IBM MQ Message Queue Interface (MQI) a IBM MQ Administration Interface (MQAI) jsou správcem front nebo uživatelskou procedurou vráceny kód **dokončení** a kód **příčiny** , které označují úspěch nebo selhání volání.

Aplikace nesmí záviset na chybách, které jsou kontrolovány v určitém pořadí, s výjimkou případů, kdy je to výslovně uvedeno. Pokud z volání může vzniknout více než jeden kód dokončení nebo kód příčiny, konkrétní ohlášená chyba závisí na implementaci.

Aplikace, které kontrolují úspěšné dokončení po volání rozhraní API IBM MQ , musí vždy zkontrolovat kód dokončení. Nepředpokládejte hodnotu kódu dokončení na základě hodnoty kódu příčiny.

Kódy dokončení

Parametr kódu dokončení (*CompCode*) umožňuje volajícímu rychle zjistit, zda bylo volání úspěšně dokončeno, částečně dokončeno nebo selhalo.Následuje seznam kódů dokončení s více podrobnostmi, než je uvedeno v popisech volání:

MQCC_OK
Volání bylo zcela dokončeno; všechny výstupní parametry byly nastaveny. Parametr **Reason** má v tomto případě vždy hodnotu MQRC_NONE.

MQCC_VAROVÁNÍ
Volání bylo částečně dokončeno. Kromě výstupních parametrů *CompCode* a *Reason* mohly být nastaveny některé výstupní parametry. Parametr **Reason** poskytuje další informace o částečném dokončení.

MQCC_FAILED
Zpracování volání nebylo dokončeno. Stav správce front je nezměněn, s výjimkou případu, kdy je to výslovně uvedeno. Výstupní parametry *CompCode* a *Reason* byly nastaveny; ostatní parametry jsou nezměněny, kromě případů, kdy je to uvedeno.

Příčinou může být porucha v aplikačním programu, nebo to může být výsledek nějaké situace mimo program, například oprávnění uživatele mohlo být odvoláno. Parametr **Reason** poskytuje další informace o chybě.

Kódy příčin

Parametr kódu příčiny (*Reason*) kvalifikuje parametr kódu dokončení (*CompCode*).

Pokud neexistuje žádný zvláštní důvod pro hlášení, vrátí se MQRC_NONE. Úspěšné volání vrátí MQCC_OK a MQRC_NONE.

Pokud je kód dokončení buď MQCC_WARNING, nebo MQCC_FAILED, správce front vždy ohlásí oprávněnou příčinu; podrobnosti jsou uvedeny pod jednotlivými popisy volání.

Pokud uživatelské procedury nastavují kódy dokončení a příčiny, musí dodržovat tato pravidla. Kromě toho musí být hodnoty speciálních příčin definované uživatelskými programy menší než nula, aby se zajistilo, že nebudou v konfliktu s hodnotami definovanými správcem front. Uživatelské procedury mohou v případě potřeby nastavit příčiny, které již správce front definoval.

Kódy příčiny se také vyskytují v:

- Pole *Reason* struktury MQDLH
- Pole *Feedback* struktury MQMD

Úplný popis kódů příčiny naleznete v tématu Zprávy a kódy příčiny.

Pravidla pro ověřování platnosti voleb rozhraní MQI

V tomto oddílu jsou uvedeny situace, které vytvářejí kód příčiny MQRC_OPTIONS_ERROR z volání MQOPEN, MQPUT, MQPUT1, MQGET, MQCLOSE nebo MQSUB.

Volání MQOPEN

Pro volby volání MQOPEN:

- Musí být uveden alespoň *jeden* z následujících:
  - MQOO_BROWSE
  - MQOO_INPUT_EXCLUSIVE [1]
  - MQOO_INPUT_SHARED [1]
  - MQOO_INPUT_AS_Q_DEF [1]
  - MQOO_DOTAZOVAT
  - MQOO_OUTPUT
  - MQOO_SET
  - MQOO_BIND_ON_OPEN [2]
  - MQOO_BIND_NOT_FIXED [2]
  - MQOO_BIND_ON_GROUP [2]
  - MQOO_BIND_AS_Q_DEF [2]
- Je povolena pouze *jedna* z následujících možností:
  - MQOO_READ_AHEAD
  - MQOO_NO_READ_AHEAD
  - MQOO_READ_AHEAD_AS_Q_DEF

1. Je povolena pouze *jedna* z následujících možností:
   - MQOO_INPUT_EXCLUSIVE
   - MQOO_INPUT_SHARED
   - MQOO_INPUT_AS_Q_DEF
2. Je povolena pouze *jedna* z následujících možností:
   - MQOO_BIND_ON_OPEN

- MQOO_BIND_NOT_FIXED
- MQOO_BIND_ON_GROUP
- MQOO_BIND_AS_Q_DEF

  Poznámka: Dříve vypsané volby se vzájemně vylučují. Avšak vzhledem k tomu, že hodnota MQOO_BIND_AS_AS_Q_DEF je nula, její uvedení s některou z dalších dvou voleb vazby nezpůsobí kód příčiny MQRC_OPTIONS_ERROR. MQOO_BIND_AS_Q_DEF je poskytován jako pomůcka pro programovou dokumentaci.

- Je-li zadána volba MQOOO_SAVE_ALL_CONTEXT, musí být zadána také jedna z voleb MQOO_INPUT_ *.
- Je-li zadána jedna z voleb MQOO_SET_ * _CONTEXT nebo MQOO_PASS_ * _CONTEXT, musí být zadána také volba MQOOO_OUTPUT.
- Je-li zadána volba MQOO_CO_OP, musí být zadána také volba MQOO_BROWSE.
- Je-li zadána volba MQOO_NO_MULTICAST, musí být zadána také volba MQOO_OUTPUT.

Volání MQPUT

Pro volby vložení zprávy:

- Kombinace MQPMO_SYNCPOINT a MQPMO_NO_SYNCPOINT není povolena.
- Je povolena pouze *jedna* z následujících možností:
  - MQPMO_DEFAULT_CONTEXT
  - MQPMO_NO_CONTEXT
  - MQPMO_PASS_ALL_CONTEXT
  - MQPMO_PASS_IDENTITY_CONTEXT
  - MQPMO_SET_ALL_CONTEXT
  - MQPMO_SET_IDENTITY_CONTEXT
- Je povolena pouze *jedna* z následujících možností:
  - MQPMO_ASYNC_RESPONSE
  - MQPMO_SYNC_RESPONSE
  - MQPMO_RESPONSE_AS_TOPIC_DEF
  - MQPMO_RESPONSE_AS_Q_DEF
- Volba MQPMO_ALTERNATE_USER_AUTHORITY není povolena (je platná pouze pro volání MQPUT1).

Volání MQPUT1

Pro volby put-message jsou pravidla stejná jako pro volání MQPUT, s výjimkou následujících:

- MQPMO_ALTERNATE_USER_AUTHORITY je povoleno.
- MQPMO_LOGICAL_ORDER není povoleno.

Volání MQGET

Pro volby get-message:

- Je povolena pouze *jedna* z následujících možností:
  - MQGMO_NO_SYNCPOINT
  - MQGMO_SYNCPOINT
  - MQGMO_SYNCPOINT_IF_PERSISTENT
- Je povolena pouze *jedna* z následujících možností:
  - MQGMO_BROWSE_FIRST

  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT není povolen s žádným z následujících:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQGMO_LOCK
  - MQGMO_UNLOCK
- MQGMO_SYNCPOINT_IF_PERSISTENT není povolena s žádnou z následujících možností:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQGMO_COMPLETE_MSG
  - MQGMO_UNLOCK
- MQGMO_MARK_SKIP_BACKOUT vyžaduje zadání MQGMO_SYNCPOINT.
- Kombinace MQGMO_WAIT a MQGMO_SET_SIGNAL není povolena.
- Je-li zadána hodnota MQGMO_LOCK, musí být zadána také jedna z následujících hodnot:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
- Je-li zadána volba MQGMO_UNLOCK, jsou povoleny pouze následující hodnoty:
  - MQGMO_NO_SYNCPOINT
  - MQGMO_NO_WAIT

Volání MQCLOSE

Pro volby volání MQCLOSE:

- Kombinace MQCO_DELETE a MQCO_DELETE_PURGE není povolena.
- Je povolena pouze jedna z následujících možností:
  - MQCO_KEEP_SUB
  - MQCO_REMOVE_SUB

Volání MQSUB

Pro volby volání MQSUB:

- Musí být uveden alespoň jeden z následujících:
  - MQSO_ALTER
  - MQSO_RESUME
  - MQSO_CREATE
- Je povolena pouze jedna z následujících možností:
  - MQSO_TRVALÉ
  - MQSO_NON_TRVALÝ

Poznámka: Dříve vypsané volby se vzájemně vylučují. Avšak vzhledem k tomu, že hodnota MQSO_NON_TRVALÝ je nula, její zadání s MQSO_TRVALÝ nepovede ke kódu příčiny MQRC_OPTIONS_ERROR. MQSO_NON_TRVALÉ je k dispozici pro pomoc s dokumentací programu.

- Kombinace MQSO_GROUP_SUB a MQSO_MANAGED není povolena.
- MQSO_GROUP_SUB vyžaduje zadání MQSO_SET_CORREL_ID.
- Je povolena pouze jedna z následujících možností:
  - MQSO_ANY_USERID
  - MQSO_FIXED_USERID
- MQSO_NEW_PUBLICATIONS_ONLY je povoleno v kombinaci s:
  - MQSO_CREATE
  - MQSO_ALTER, pokud byl parametr MQSO_NEW_PUBLICATIONS_ONLY nastaven na původní odběr
- Kombinace hodnot MQSO_PUBLICATIONS_ON_REQUEST a SubLevel větší než 1 není povolena.
- Je povolena pouze jedna z následujících možností:
  - MQSO_WILDCARD_CHAR
  - MQSO_WILDCARD_TOPIC
- Volba MQSO_NO_MULTICAST vyžaduje zadání volby MQSO_MANAGED.

Zprávy příkazů publikování/odběru ve frontě

Aplikace může používat zprávy příkazů MQRFH2 k řízení aplikace publikování/odběru ve frontě.

Aplikace, která používá produkt MQRFH2 pro publikování/odběr, může do systému SYSTEM.BROKER.CONTROL.QUEUE:

- “Odstranit zprávu publikování” na stránce 879
- “Zpráva o zrušení registrace odběratele” na stránce 880
- “Publikovat zprávu” na stránce 883
- “Zpráva registrace odběratele” na stránce 885
- “Zpráva o aktualizaci požadavku” na stránce 890

Pokud píšete aplikace publikování/odběru zařazené ve frontě, musíte porozumět těmto zprávám, zprávě odezvy správce front a deskriptoru zpráv (MQMD). Viz následující informace:

- “Zpráva s odpovědí správce front” na stránce 892
- “Nastavení MQMD pro publikování předaná správcem front” na stránce 897
- “Nastavení MQMD ve zprávách odpovědí správce front” na stránce 898
- “Kódy příčiny publikování/odběru” na stránce 893

Příkazy jsou obsaženy ve složce psc v poli **NameValueData** záhlaví MQRFH2 . Zpráva, kterou může zprostředkovatel odeslat jako odpověď na zprávu příkazu, je obsažena ve složce pscr .

Popisy jednotlivých příkazů uvádějí vlastnosti, které mohou být obsaženy ve složce. Není-li uvedeno jinak, vlastnosti jsou volitelné a mohou se vyskytovat pouze jednou.

Názvy vlastností se zobrazují jako <Command>.
Hodnoty musí být ve formátu řetězce, například: Publish.
Řetězcová konstanta představující hodnotu vlastnosti je uvedena v závorkách, například: (MQPSC_PUBLISH).
Řetězcové konstanty jsou definovány v souboru záhlaví cmqpsc.h , který je dodáván se správcem front.

Odstranit zprávu publikování

Příkazová zpráva **Delete Publication** je odeslána správci front od vydavatele nebo od jiného správce front, aby správci front sdělila, že má odstranit veškerá zachovaná publikování pro určená témata.

Tato zpráva je odeslána do fronty monitorované rozhraním pro publikování/odběr ve frontě správce front.

Vstupní fronta by měla být frontou, do které bylo odesláno původní publikování.

Pokud máte oprávnění k některým, ale ne ke všem tématům, která jsou uvedena ve zprávě příkazu **Delete Publication** , odstraní se pouze tato témata. Zpráva **Broker Response** označuje, která témata nejsou odstraněna.

Podobně, pokud příkaz **Publish** obsahuje více než jedno téma, příkaz **Delete Publication** odpovídající některým, ale ne všem, z těchto témat odstraní pouze publikování pro témata uvedená v příkazu **Delete Publication** .

Podrobné informace o parametrech deskriptoru zpráv (MQMD), které jsou potřebné při odesílání zprávy příkazu správci front, naleznete v části "Nastavení MQMD pro publikování předaná správcem front" na stránce 897 .

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je `DeletePub` (*MQPSC_DELETE_PUBLETE_PUBLICATION*).

Tato vlastnost musí být určena.

Téma > (*MQPSC_TOPIC*)
Hodnota je řetězec obsahující téma, pro které mají být odstraněna zachovaná publikování. Do řetězce lze zahrnout zástupné znaky pro odstranění publikování ve více než jednom tématu.

Tato vlastnost musí být zadána; může být opakována pro libovolný počet témat.

DelOpt (*MQPSC_DELETE_OPTION*)
Vlastnost voleb odstranění může mít jednu z následujících hodnot:

`Lokální` (*MQPSC_LOCAL*)

Všechna zachovaná publikování pro určená témata jsou odstraněna v lokálním správci front (tj. ve správci front, kterému je tato zpráva odeslána) bez ohledu na to, zda byla publikována s volbou `Lokální` .

Publikování v jiných správcích front nejsou ovlivněna.

Není (*MQPSC_NONE*)

Všechny volby mají své výchozí hodnoty. To má stejný účinek jako vynechání vlastnosti `DelOpt` . Pokud jsou současně zadány i jiné volby, volba Žádný se ignoruje.

Výchozí nastavení, pokud je tato vlastnost vynechána, je, že všechna zachovaná publikování pro zadaná témata jsou odstraněna ve všech správcích front v síti bez ohledu na to, zda byla publikována s volbou `Lokální` .

Příklad

Zde je příklad `NameValueData` pro zprávu příkazu **Delete Publication** . Tuto volbu používá ukázková aplikace k odstranění zachovaného publikování v lokálním správci front, které obsahuje nejnovější skóre v porovnání mezi Team1 a Team2.

```
<psc>
  <Command>DeletePub</Command>
  <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
  <DelOpt>Local</DelOpt>
</psc>
```

Zpráva o zrušení registrace odběratele

Zpráva příkazu **Deregister Subscriber** je odeslána správci front odběratelem nebo jinou aplikací jménem odběratele, aby bylo zřejmé, že již nechce přijímat zprávy odpovídající daným parametrům.

Tato zpráva je odeslána do SYSTEM.BROKER.CONTROL.QUEUE, řídicí fronta správce front. Uživatel musí mít potřebné oprávnění pro vložení zprávy do této fronty.

Podrobné informace o parametrech deskriptoru zpráv (MQMD), které jsou zapotřebí při odesílání zprávy příkazu správci front, naleznete v tématu Nastavení MQMD pro publikování předaná správcem front .

Registraci jednotlivého odběru lze zrušit zadáním odpovídajícího tématu, bodu odběru a hodnot filtru původního odběru. Pokud nebyla v původním odběru zadána žádná z hodnot (tj. převzaly výchozí hodnoty), měly by být při zrušení registrace odběru vynechány.

Všechny odběry pro odběratele nebo skupinu odběratelů lze zrušit pomocí volby `DeregAll` . Pokud je například zadána volba `DeregAll` společně s bodem odběru (ale bez tématu nebo filtru), budou všechny odběry pro odběratele v určeném bodu odběru odregistrovány bez ohledu na téma a filtr. Je povolena libovolná kombinace tématu, filtru a bodu odběru. Jsou-li zadány všechny tři položky, může se shodovat pouze jeden odběr a volba `DeregAll` je ignorována.

Zpráva musí být odeslána odběratelem, který registroval odběr; to je potvrzeno kontrolou ID uživatele odběratele.

Registraci odběrů může zrušit také administrátor systému pomocí příkazů MQSC nebo PCF. Odběry registrované s dočasnou dynamickou frontou jsou však přidruženy k frontě, nikoli pouze k názvu fronty. Pokud je fronta odstraněna, buď explicitně, nebo aplikací odpojenou od správce front, není již možné zrušit registraci odběrů pro tuto frontu pomocí příkazu **Deregister Subscriber** . Registrace odběrů může být zrušena pomocí pracovní plochy vývojáře a správce front je automaticky odebere při příštím přihlášení k odběru nebo při příštím spuštění správce front. Za normálních okolností by aplikace měly zrušit registraci svých odběrů před odstraněním fronty nebo odpojením od správce front.

Pokud odběratel odešle zprávu o zrušení registrace odběru a obdrží zprávu s odpovědí, že tento odběr byl úspěšně zpracován, mohou se některá publikování stále dostat do fronty odběratele, pokud byla zpracována správcem front ve stejnou dobu, kdy byla zrušena registrace odběru. Pokud nejsou zprávy odebrány z fronty, může dojít k nahromadění nezpracovaných zpráv ve frontě odběratele. Pokud aplikace po chvíli spánku provede smyčku, která obsahuje volání MQGET s příslušným `CorrelId` , tyto zprávy se vymažou z fronty.

Podobně, pokud odběratel používá trvalou dynamickou frontu a zruší registraci a zavře frontu pomocí volby *MQCO_DELETE_PURGE* ve volání MQCLOSE, nemusí být tato fronta prázdná. Nejsou-li při odstranění fronty dosud potvrzena některá publikování ze správce front, je návratový kód MQRC_Q_NOT_EMPTY zadán voláním MQCLOSE. Aplikace se tomuto problému může vyhnout tím, že čas od času spí a znovu zadá volání MQCLOSE.

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je `DeregSub` (*MQPSC_DEREGISTER_ODBĚRATEL*).

Tato vlastnost musí být určena.

Téma (*MQPSC_TOPIC*)
Hodnota je řetězec, který obsahuje téma, jehož registrace má být zrušena.

Tuto vlastnost lze volitelně opakovat, pokud má být zrušena registrace více témat. Lze ji vynechat, pokud je v <RegOpt>uvedeno `DeregAll` .

Zadaná témata mohou být podmnožinou témat, která jsou registrována, pokud odběratel chce zachovat odběry pro jiná témata. Zástupné znaky jsou povoleny, ale řetězec tématu, který obsahuje zástupné znaky, se musí přesně shodovat s odpovídajícím řetězcem, který byl uveden ve zprávě příkazu **Deregister Subscriber** .

SubPoint (*MQPSC_SUBSCRIPTION_POINT*)
Hodnota je řetězec, který určuje bod odběru, od kterého má být odběr odpojen.

Tato vlastnost se nesmí opakovat. Lze ji vynechat, pokud je uvedeno < Topic> , nebo pokud je DeregAll uvedeno v <RegOpt>. Pokud tuto vlastnost vynecháte, dojde k následujícímu:

- Pokud **nezadáte** DeregAll, budou odběry odpovídající vlastnosti < Topic> (a vlastnosti < Filtr > , je-li k dispozici) odregistrovány od výchozího bodu odběru.
- Zadáte-li volbu DeregAll, budou všechny odběry (odpovídající vlastnostem < Topic> a < Filtr > , jsou-li k dispozici) odregistrovány ze všech bodů odběru.

Všimněte si, že výchozí bod odběru nelze zadat explicitně. Proto neexistuje žádný způsob, jak zrušit registraci všech odběrů pouze z tohoto bodu odběru; musíte určit témata.

SubIdentity (*MQPSC_SUBSCRIPTION_IDENTITY*)
Jedná se o řetězec s proměnnou délkou s maximální délkou 64 znaků. Používá se k reprezentaci aplikace se zájmem o odběr. Správce front udržuje sadu identit odběratele pro každý odběr. Každý odběr může povolit, aby jeho sada identit uchovala pouze jednu identitu, nebo neomezený počet identit.

Pokud je SubIdentity v sadě identit pro odběr, pak je odebrán ze sady. Pokud se sada identit v důsledku toho stane prázdnou, odběr se odebere ze správce front, pokud není vlastnost LeaveOnly zadána jako hodnota vlastnosti RegOpt . Pokud sada identit stále obsahuje jiné identity, odběr nebude ze správce front odebrán a tok publikování nebude přerušen.

Pokud je zadána volba SubIdentity , ale SubIdentity není v sadě identit pro odběr, příkaz **Deregister Subscriber** selže s návratovým kódem *MQRCCF_SUB_IDENTITY_ERROR*.

Filtr (*MQPSC_FILTER*)
Hodnota je řetězec určující filtr, jehož registrace má být zrušena. Musí se přesně shodovat, včetně případu a všech mezer, s filtrem odběru, který byl dříve registrován.

Tuto vlastnost lze volitelně opakovat, pokud má být zrušena registrace více než jednoho filtru. Lze ji vynechat, pokud je uvedeno < Topic> , nebo pokud je DeregAll uvedeno v <RegOpt>.

Zadané filtry mohou být podmnožinou těch registrovaných, pokud chce odběratel uchovat odběry pro jiné filtry.

RegOpt (*MQPSC_REGISTRATION_OPTION*)
Vlastnost voleb registrace může mít následující hodnoty:

DeregAll
(*MQPSC_DEREGISTER_ALL*)

Všechny odpovídající odběry registrované pro tohoto odběratele mají být odregistrovány.

Pokud zadáte DeregAll:

- < Topic>, <SubPoint>a < Filtr > lze vynechat.
- < Topic> a < Filtr > lze v případě potřeby opakovat.
- <SubPoint> se nesmí opakovat.

Pokud **neuvedete** DeregAll:

- < Topic> musí být uvedeno a v případě potřeby může být opakováno.
- <SubPoint> a < Filtr > lze vynechat.
- <SubPoint> se nesmí opakovat.
- < Filtr > lze v případě potřeby opakovat.

Pokud se témata i filtry opakují, budou odebrány všechny odběry odpovídající všem jejich kombinacím. Například příkaz **Deregister Subscriber** , který uvádí tři témata a tři filtry, se pokusí odebrat devět odběrů.

CorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

CorrelId v deskriptoru zprávy (MQMD), který nesmí být nulový, se používá k identifikaci odběratele. Musí odpovídat CorrelId použitému v původním odběru.

FullResp
(*MQPSC_FULL_RESPONSE*)

Je-li zadána volba `FullResp` , budou ve zprávě odpovědi vráceny všechny atributy odběru, pokud příkaz neselže.

Když je uvedeno `FullResp` `DeregAll` není povoleno v příkazu **Deregister Subscriber** . Není také možné zadat více témat. Příkaz selže s návratovým kódem *MQRCCF_REG_OPTIONS_ERROR*, v obou případech.

LeaveOnly
(*MQPSC_LEAVE_ONLY*)

Zadáte-li tento parametr spolu s položkou `SubIdentity` , která je v sadě identit pro odběr, odebere se produkt `SubIdentity` ze sady identit pro odběr. Odběr není ze správce front odebrán, a to ani v případě, že výsledná sada identit je prázdná. Pokud hodnota `SubIdentity` není v sadě identit, příkaz selže s návratovým kódem *MQRCCF_SUB_IDENTITY_ERROR*.

Pokud je zadána volba `LeaveOnly` bez volby `SubIdentity`, příkaz selže s návratovým kódem *MQRCCF_REG_OPTIONS_ERROR*.

Není-li zadána volba `LeaveOnly` ani volba `SubIdentity` , bude odběr odebrán bez ohledu na obsah sady identit pro odběr.

NONE
(MQPSC_NONE)

Všechny volby mají své výchozí hodnoty. To má stejný účinek jako vynechání vlastnosti voleb registrace. Pokud jsou současně zadány i jiné volby, volba Žádný se ignoruje.

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

Je-li zadána identita odběratele (fronta, správce front a correlid), není omezena na jedno ID uživatele. To se liší od existujícího chování správce front, který přidružuje ID uživatele původní registrační zprávy k identitě odběratele, a od té doby zabraňuje jakémukoli jinému uživateli, který tuto identitu používá. Pokud se nový odběratel pokusí použít stejnou identitu, vrátí se návratový kód *MQRCCF_DUPLICATE_SUBSCRIPTION* .

Každý uživatel může upravit nebo zrušit registraci odběru, pokud má odpovídající oprávnění, a vyhnout se tak stávající kontrole, že ID uživatele musí odpovídat ID původního odběratele.

Chcete-li přidat tuto volbu k existujícímu odběru, musí příkaz pocházet ze stejného ID uživatele jako samotný původní odběr.

Je-li pro odběr, pro který má být zrušena registrace, nastaveno `VariableUserId` , musí být nastaveno v době zrušení registrace, aby bylo možné určit, pro který odběr má být zrušena registrace. Jinak se k identifikaci odběru použije ID uživatele příkazu **Deregister Subscriber** . Pokud je zadán název odběru, je tato volba přepsána spolu s dalšími identifikátory odběratele.

Pokud je tato vlastnost vynechána, výchozí nastavení je, že nejsou nastaveny žádné volby registrace.

QMgrName (*MQPSC_Q_MGR_NAME*)
Hodnota je název správce front pro frontu odběratele. Musí odpovídat hodnotě `QMgrName` použité v původním odběru.

Pokud je tato vlastnost vynechána, je výchozí hodnotou název `ReplyToQMgr` v deskriptoru zprávy (MQMD). Je-li výsledný název prázdný, standardně se použije název správce front.

Název QName (*MQPSC_Q_NAME*)
Hodnota je název fronty odběratele. Musí odpovídat názvu `QName` použitému v původním odběru.

Je-li tato vlastnost vynechána, výchozí hodnotou je název `ReplyToQ` v deskriptoru zprávy (MQMD), který nesmí být prázdný.

SubName (*MQPSC_SUBSCRIPTION_NAME*)
Zadáte-li v příkazu **Deregister Subscriber** hodnotu SubName , bude mít hodnota SubName přednost před všemi ostatními poli identifikátoru kromě ID uživatele, není-li VariableUserId nastaveno na samotný odběr. Není-li VariableUserId nastaveno, příkaz **Deregister Subscriber** uspěje pouze v případě, že ID uživatele zprávy příkazu odpovídá ID uživatele odběru, pokud ne, příkaz selže s návratovým kódem *MQRCCF_DUPLICATE_IDENTITY*.

Pokud existuje odběr, který odpovídá tradiční identitě tohoto příkazu, ale nemá žádný SubName , příkaz **Deregister Subscriber** selže s návratovým kódem *MQRCCF_SUB_NAME_ERROR*. Pokud dojde k pokusu o zrušení registrace odběru, který má SubName , pomocí zprávy příkazu, která odpovídá tradiční identitě, ale bez zadání SubName , příkaz uspěje.

SubUserData (*MQPSC_SUBSCRIPTION_USER_DATA*)
Jedná se o textový řetězec s proměnnou délkou. Hodnota je uložena správcem front s odběrem, ale nemá žádný vliv na doručení publikování odběrateli. Hodnotu lze změnit opětovnou registrací na stejný odběr s novou hodnotou. Tento atribut je určen pro použití v aplikaci.

SubUserData jsou vrácena v Metatopických informacích (MQCACF_REG_SUB_USER_DATA) pro odběr, pokud jsou přítomna data SubUser.

Příklad

Zde je příklad NameValueData pro zprávu příkazu **Deregister Subscriber** . V tomto příkladu ukázková aplikace ruší registraci svého odběru témat, která obsahují nejnovější skóre pro všechny shody. Identita odběratele, včetně CorrelId, je převzata z výchozích hodnot v deskriptoru MQMD.

```
<psc>
  <Command>DeregSub</Command>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

Publikovat zprávu

Zpráva příkazu **Publish** je vložena do fronty nebo ze správce front odběrateli za účelem publikování informací o určeném tématu nebo tématech.

Je nezbytné oprávnění k vložení zprávy do fronty a oprávnění k publikování informací o určeném tématu nebo tématech.

Pokud má uživatel oprávnění publikovat informace o některých, ale ne o všech tématech, použijí se k publikování pouze tato témata; varovná odpověď označuje, která témata se nepoužívají k publikování.

Pokud má odběratel nějaké odpovídající odběry, předá správce front zprávu **Publish** frontám odběratele definovaným v odpovídajících zprávách příkazu **Register Subscriber** .

Podrobné informace o parametrech deskriptoru zpráv (MQMD) potřebných při odesílání příkazové zprávy správci front a používaných při předávání publikování správci front odběrateli naleznete v tématu Zpráva odezvy správce front .

Správce front předává zprávu **Publish** jiným správcům front v síti, kteří mají odpovídající odběry, pokud se nejedná o lokální publikování.

V těle zprávy jsou uvedena případná publikační data. Data lze popsat ve složce <mcd> v poli NameValueData záhlaví MQRFH2 .

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je Publikovat (*MQPSC_PUBLISH*).

Tato vlastnost musí být určena.

Téma (*MQPSC_TOPIC*)
Hodnota je řetězec, který obsahuje téma, které kategorizuje tuto publikaci. Nejsou povoleny žádné zástupné znaky.

Musíte přidat téma do seznamu názvů SYSTEM.QPUBSUB.QUEUE.NAMELIST, viz Přidání proudu , kde získáte pokyny, jak dokončit tuto úlohu.

Tato vlastnost musí být určena a volitelně ji lze opakovat pro libovolný počet témat.

SubPoint (*MQPSC_SUBSCRIPTION_POINT*)
Bod odběru, ve kterém je publikování publikováno.

V produktu WebSphere Event Broker 6.0je hodnotou vlastnosti <SubPoint> hodnota atributu Bod odběru uzlu publikování, který publikování zpracovává.

V produktu IBM WebSphere MQ 7.0.1musí hodnota vlastnosti <SubPoint> odpovídat názvu bodu odběru. Viz Přidání bodu odběru.

PubOpt (*MQPSC_PUBLICATION_OPTION*)
Vlastnost voleb publikování může nabývat následujících hodnot:

RetainPub
(*MQPSC_RETAIN_PUB*)

Správce front si ponechá kopii publikování. Není-li tato volba nastavena, je publikování odstraněno, jakmile správce front odešle publikování všem aktuálním odběratelům.

IsRetainedpublikování
(*MQPSC_IS_RETAINED_PUB*)

(Lze nastavit pouze správcem front.) Toto publikování bylo zachováno správcem front. Správce front nastaví tuto volbu tak, aby informoval odběratele, že toto publikování bylo publikováno dříve a bylo zachováno, za předpokladu, že odběr byl registrován s volbou `InformIfRetained` . Je nastaven pouze jako odpověď na zprávu příkazu `Registrovat odběratele` nebo `Vyžádat aktualizaci` . Zachovaná publikování, která jsou odesílána přímo odběratelům, nemají tuto volbu nastavenou.

Lokální
(*MQPSC_LOCAL*)

Tato volba sděluje správci front, že toto publikování nesmí být odesláno jiným správcům front. Všichni odběratelé, kteří jsou registrováni v tomto správci front, obdrží toto publikování, pokud mají odpovídající odběry.

OtherSubs
(*MQPSC_OTHER_SUBS_ONLY*)

Tato volba umožňuje jednodušší zpracování aplikací typu konference, kde je vydavatel také odběratelem stejného tématu. Informuje správce front, aby neodesílal publikování do fronty odběratele vydavatele, a to ani v případě, že má odpovídající odběr. Fronta odběratele vydavatele se skládá z jeho `QMgrName`, `QNamea` volitelného `CorrelId`, jak je popsáno v následujícím seznamu.

CorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

`CorrelId` v deskriptoru MQMD (který nesmí být nulový) je součástí fronty odběratele vydavatele v aplikacích, kde je vydavatel také odběratelem.

NONE
(*MQPSC_NONE*)

Všechny volby mají své výchozí hodnoty. To má stejný efekt jako vynechání vlastnosti voleb publikování. Pokud jsou současně zadány i jiné volby, volba Žádný se ignoruje.

Můžete mít více než jednu volbu publikování zavedením dalších prvků <PubOpt> .

Je-li tato vlastnost vynechána, výchozí nastavení je, že nejsou nastaveny žádné volby publikování.

PubTime (*MQPSC_PUBLISH_TIMESTAMP*)
Hodnota je volitelné časové razítko publikování nastavené vydavatelem. Je dlouhý 16 znaků s formátem:

```
YYYYMMDDHHMMSSTH
```

pomocí univerzálního času. Tyto informace nejsou před odesláním odběratelům kontrolovány správcem front.

SeqNum (*MQPSC_SEQUENCE_NUMBER*)
Hodnota je volitelné pořadové číslo nastavené vydavatelem.

Musí se zvýšit o 1 s každou publikací. Správce front však tuto kontrolu nekontroluje, což pouze přenáší tyto informace odběratelům.

Pokud jsou publikování ve stejném tématu publikována do různých propojených správců front, je odpovědností vydavatelů zajistit, aby pořadová čísla, jsou-li použita, byla smysluplná.

QMgrName (*MQPSC_Q_MGR_NAME*)
Hodnota je řetězec obsahující název správce front pro frontu odběratele vydavatele v aplikacích, kde je vydavatel také odběratelem (viz `OtherSubs`).

Pokud je tato vlastnost vynechána, je výchozí hodnotou název `ReplyToQMgr` v deskriptoru zprávy (MQMD). Je-li výsledný název prázdný, standardně se použije název správce front.

Název QName (*MQPSC_Q_NAME*)
Hodnota je řetězec obsahující název fronty odběratele vydavatele v aplikacích, kde je vydavatel také odběratelem (viz `OtherSubsOnly`).

Je-li tato vlastnost vynechána, výchozí hodnotou je název `ReplyToQ` v deskriptoru zprávy (MQMD), který nesmí být prázdný, je-li nastaven parametr `OtherSubsOnly` .

Příklad

Zde je několik příkladů *NameValueData* pro zprávu příkazu **Publish** .

První příklad je pro publikaci odeslanou simulátorem shody v ukázkové aplikaci, která označuje, že byla spuštěna shoda.

```
<psc>
  <Command>Publish</Command>
  <Topic>Sport/Soccer/Event/MatchStarted</Topic>
</psc>
```

Druhý příklad je pro zachované publikování. Publikuje se nejnovější skóre ve shodě mezi Team1 a Team2 .

```
<psc>
  <Command>Publish</Command>
  <PubOpt>RetainPub</PubOpt>
  <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
</psc>
```

Zpráva registrace odběratele

Zpráva příkazu **`Register Subscriber`** je odeslána správci front odběratelem nebo jinou aplikací jménem odběratele, aby označila, že se chce přihlásit k odběru jednoho nebo více témat v bodu odběru. Lze také zadat filtr obsahu zprávy.

Ve výrazech filtru publikování/odběru způsobí vnoření závorek exponenciální snížení výkonu. Vyhněte se vnoření závorek do hloubky větší než asi 6.

Zpráva je odeslána do SYSTEM.BROKER.CONTROL.QUEUE, což je řídicí fronta správce front. Kromě přístupových oprávnění (nastavených administrátorem systému správce front) pro dané téma nebo témata v rámci odběru je vyžadováno oprávnění k vložení zprávy do této fronty.

Pokud má uživatel oprávnění k některým, ale ne ke všem tématům, jsou registrovány pouze ty, které mají oprávnění; varovná odpověď označuje ty, které nejsou registrovány.

Podrobné informace o parametrech deskriptoru zpráv (MQMD), které jsou potřebné při odesílání zprávy příkazu správci front, naleznete v části "Nastavení MQMD ve zprávách příkazů pro správce front" na stránce 896 .

Je-li odpověď na frontu dočasnou dynamickou frontou, správce front při zavření fronty automaticky odregistruje odběr.

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je RegSub (*MQPSC_REGISTER_ODBĚRATEL*). Tato vlastnost musí být určena.

Téma (*MQPSC_TOPIC*)
Téma, pro které chce odběratel přijímat publikování. Jako součást tématu lze zadat zástupné znaky.

Použijete-li příkaz MQSC **display sub** ke kontrole takto vytvořeného odběru, hodnota značky <Topic> se zobrazí jako vlastnost TOPICSTR odběru.

Tato vlastnost je povinná a lze ji volitelně opakovat pro libovolný počet témat.

SubPoint (*MQPSC_SUBSCRIPTION_POINT*)
Hodnota je bod odběru, ke kterému je odběr připojen.

Je-li tato vlastnost vynechána, použije se výchozí bod odběru.

V produktu WebSphere Event Broker 6.0se hodnota vlastnosti <SubPoint> musí shodovat s hodnotou atributu Bod odběru v uzlech publikování, které jsou přihlášeny k odběru.

V produktu IBM WebSphere MQ 7.0.1musí hodnota vlastnosti <SubPoint> odpovídat názvu bodu odběru. Viz Přidání bodu odběru.

Filtr (*MQPSC_FILTER*)
Hodnota je výraz SQL, který se používá jako filtr pro obsah publikačních zpráv. Pokud se publikování v zadaném tématu shoduje s filtrem, odešle se odběrateli. Tato vlastnost odpovídá řetězci výběru, který se používá ve voláních MQSUB a MQOPEN. Další informace naleznete v tématu Výběr obsahu zprávy .

Je-li tato vlastnost vynechána, nebude provedeno žádné filtrování obsahu.

RegOpt (*MQPSC_REGISTRATION_OPTION*)
Tato vlastnost Volby registrace může mít následující hodnoty:

AddName
(*MQPSC_ADD_NAME*)

Je-li zadán pro existující odběr, který odpovídá tradiční identitě tohoto příkazu Registrovat odběr, ale nemá aktuální hodnotu SubName , přidá se k odběru hodnota SubName uvedená v tomto příkazu.

Je-li uvedeno AddName , pole SubName je povinné, jinak se vrátí MQRCCF_REG_OPTIONS_ERROR.

CorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

CorrelId v deskriptoru zprávy (MQMD) se používá při odesílání odpovídajících publikování do fronty odběratele. CorrelId nesmí být nula,

FullResp
(*MQPSC_FULL_RESPONSE*)

Jsou-li zadány všechny atributy odběru, jsou vráceny ve zprávě odpovědi, pokud příkaz neselže.

Volba FullResp je platná pouze v případě, že zpráva příkazu odkazuje na jeden odběr. Proto je v příkazu povoleno pouze jedno téma; jinak příkaz selže s návratovým kódem *MQRCCF_REG_OPTIONS_ERROR*.

InformIfRet
(*MQPSC_INFORM_IF_ZACHOVÁNO*)

Správce front informuje odběratele v případě, že je při odesílání zprávy publikování v reakci na zprávu příkazu **Register Subscriber** nebo **Request Update** zachováno publikování. Správce front to provede tak, že do zprávy zahrnou volbu publikování `IsRetainedPub` .

JoinExcl
(*MQPSC_JOIN_EXCLUSIVE*)

Tato volba označuje, že uvedená `SubIdentity` by měla být přidána jako výlučný člen sady identit pro odběr a že do sady nelze přidat žádné další identity.

Pokud se identita již připojila k 'sdílenému' a je jedinou položkou v sadě, sada se změní na výlučný zámek držený touto identitou. Jinak, pokud má odběr v současné době jiné identity v sadě identit (se sdíleným přístupem), příkaz selže s návratovým kódem *MQRCCF_SUBSCRIPTION_IN_USE*.

JoinShared
(*MQPSC_JOIN_SHARED*)

Tato volba označuje, že uvedená `SubIdentity` by měla být přidána do sady identit pro odběr.

Pokud je odběr aktuálně výlučně uzamčen (pomocí volby `JoinExcl`), příkaz selže s návratovým kódem *MQRCCF_SUBSCRIPTION_LOCKED*, pokud identita, která má uzamčený odběr, není stejná jako identita v této zprávě příkazu. V tomto případě je zámek automaticky upraven na sdílený zámek.

Lokální
(*MQPSC_LOCAL*)

Odběr je lokální a není distribuován jiným správcům front v síti. Publikování provedené v jiných správcích front nejsou doručena tomuto odběrateli, pokud nemá také odpovídající globální odběr.

NewPubs
(*MQPSC_NEW_PUBS_ONLY*)

Zachovaná publikování, která existují v době registrace odběru, nejsou odběrateli odesílána; odesílají se pouze nová publikování.

Pokud se odběratel znovu zaregistruje a změní tuto volbu tak, aby již nebyla nastavena, může být znovu odesláno publikování, které do něj již bylo odesláno.

NoAlter
(*MQPSC_NO_ALTER*)

Atributy existujícího odpovídajícího odběru se nezmění.

Při vytváření odběru je tato volba ignorována. Všechny ostatní zadané volby se vztahují na nový odběr.

Pokud má `SubIdentity` také jednu z voleb spojení (`JoinExcl` nebo `JoinShared`) je identita přidána do sady identit bez ohledu na to, zda je zadána hodnota `NoAlter` .

NONE
(*MQPSC_NONE*)

Všechny volby registrace mají své výchozí hodnoty.

Pokud je odběratel již registrován, jeho volby jsou resetovány na výchozí hodnoty (mějte na paměti, že to nemá stejný vliv jako vynechání vlastnosti voleb registrace) a vypršení platnosti odběru je aktualizováno z deskriptoru MQMD zprávy **Register Subscriber** .

Pokud jsou současně zadány další volby registrace, volba Žádný se ignoruje.

NonPers
(*MQPSC_NON_PERSISTENT*)

Publikování odpovídající tomuto odběru jsou odběrateli doručena jako dočasné zprávy.

Pers
(*MQPSC_PERSISTENT*)

Publikování odpovídající tomuto odběru jsou odběrateli doručena jako trvalé zprávy.

PersAsPublikování
(*MQPSC_PERSISTENT_AS_PUBLISH*)

Publikování odpovídající tomuto odběru jsou doručena odběrateli s perzistencí určenou vydavatelem. Toto chování je výchozí.

PersAsFronta
(*MQPSC_PERSISTENT_AS_Q*)

Publikování odpovídající tomuto odběru jsou doručena odběrateli s perzistencí určenou ve frontě odběratele.

PubOnReqOnly
(*MQPSC_PUB_ON_REQUEST_ONLY*)

Správce front neodesílá publikování odběrateli, s výjimkou odezvy na zprávu příkazu **Request Update** .

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

Je-li zadána identita odběratele (fronta, správce front a correlid), není omezena na jedno ID uživatele. To se liší od existujícího chování správce front, který přidružuje ID uživatele původní registrační zprávy k identitě odběratele, a od té doby zabraňuje jakémukoli jinému uživateli, který tuto identitu používá. Pokud se nový odběratel pokusí použít stejnou identitu, vrátí se hodnota *MQRCCF_DUPLICATE_SUBSCRIPTION* .

To umožňuje jakémukoli uživateli upravit nebo zrušit registraci odběru, pokud má uživatel vhodné oprávnění. Proto není nutné kontrolovat, zda se ID uživatele shoduje s ID uživatele původního odběratele.

Chcete-li přidat tuto volbu k existujícímu odběru, musí příkaz pocházet ze stejného ID uživatele jako samotný původní odběr.

Má-li odběr příkazu **Request Update** nastaveno `VariableUserId` , musí být nastaveno v době aktualizace požadavku, aby se označilo, na který odběr se odkazuje. Jinak se k identifikaci odběru použije ID uživatele příkazu **Request Update** . Pokud je zadán název odběru, je tato volba přepsána spolu s dalšími identifikátory odběratele.

Pokud zpráva příkazu **Register Subscriber** bez této sady voleb odkazuje na existující odběr, který má tuto sadu voleb, bude tato volba z tohoto odběru odebrána a ID uživatele odběru bude nyní opraveno. Pokud již existuje odběratel se stejnou identitou (fronta, správce front a identifikátor korelace), ale s jiným ID uživatele, které je k němu přidruženo, příkaz selže s návratovým kódem *MQRCCF_DUPLICATE_IDENTITY* , protože k identitě odběratele může být přidruženo pouze jedno ID uživatele.

Pokud je vlastnost voleb registrace vynechána a odběratel je již registrován, jeho volby registrace se nezmění a vypršení platnosti odběru se aktualizuje z deskriptoru MQMD zprávy **Register Subscriber** .

Není-li odběratel dosud registrován, bude vytvořen nový odběr se všemi možnostmi registrace, které budou mít výchozí hodnoty.

Výchozí hodnoty jsou `PersAsPub` a nejsou nastaveny žádné další volby.

QMgrName (*MQPSC_Q_MGR_NAME*)
Hodnota představuje název správce front pro frontu odběratele, do které jsou odesílána odpovídající publikování správcem front.

Pokud je tato vlastnost vynechána, je výchozí hodnotou název `ReplyToQMgr` v deskriptoru zprávy (MQMD). Pokud je výsledný název prázdný, standardně se použije název správce front `QMgrName`.

Název QName (*MQPSC_Q_NAME*)
Hodnota je název fronty odběratele, do které správce front odesílá odpovídající publikování.

Je-li tato vlastnost vynechána, výchozí hodnotou je název `ReplyToQ` v deskriptoru zprávy (MQMD), který v tomto případě nesmí být prázdný.

Pokud se jedná o dočasnou dynamickou frontu, dočasné doručení publikací (`NonPers`). musí být uvedeno ve vlastnosti <RegOpt> .

Jedná-li se o dočasnou dynamickou frontu, správce front při zavření fronty automaticky odregistruje odběr.

SubName (*MQPSC_SUBSCRIPTION_NAME*)
Jedná se o název daný konkrétnímu odběru. Můžete jej použít místo správce front, fronty a volitelného ID correlId k odkazování na odběr.

Pokud odběr s tímto produktem **SubName** již existuje, všechny ostatní atributy odběru (Topic, QMgrName, QName, CorrelId, UserId, RegOpts, UserSubData a Expiry) jsou přepsány atributy, pokud jsou zadány, které jsou předány v nové zprávě příkazu `Register Subscriber` . Je-li však použit parametr **SubName** bez zadaného pole QName a v záhlaví MQMD je zadána hodnota ReplyToQ, bude fronta odběratele změněna na hodnotu ReplyToQ.

Pokud odběr, který odpovídá tradiční identitě tohoto příkazu, již existuje, ale nemá žádný **SubName** , příkaz Registration se nezdaří s návratovým kódem *MQRCCF_DUPLICATE_SUBSCRIPTION*, pokud není zadána volba **AddName** .

Pokusíte-li se změnit existující pojmenovaný odběr pomocí jiného příkazu `Registrovat odběratele` , který určuje stejný **SubName** , a hodnoty tématu QMgrName, QName a CorrelId v novém příkazu se shodují s jiným existujícím odběrem, s nebo bez definovaného SubName , příkaz selže s návratovým kódem *MQRCCF_DUPLICATE_SUBSCRIPTION*. Tím zabráníte tomu, aby dva názvy odběrů odkazovaly na stejný odběr.

SubIdentity (*MQPSC_SUBSCRIPTION_IDENTITY*)
Tento řetězec se používá k reprezentaci aplikace se zájmem o odběr. Jedná se o znakový řetězec s proměnnou délkou s maximální délkou 64 znaků a je volitelný. Správce front udržuje sadu identit odběratele pro každý odběr. Každý odběr může povolit, aby jeho sada identit obsahovala pouze jednu identitu nebo neomezený počet identit (viz volby **JoinShared** a **JoinExcl**).

Příkaz odběru, který uvádí volbu **JoinShared** nebo **JoinExcl** , přidá **SubIdentity** do sady identit odběru, pokud již neexistuje a pokud existující sada identit umožňuje takovou akci; to znamená, že žádný jiný odběratel se exkluzivně nepřipojil nebo je sada identit prázdná.

Jakákoli změna atributů odběru jako výsledek příkazu `Registrovat odběratele` , ve kterém je zadána **SubIdentity** , bude úspěšná pouze v případě, že bude jediným členem sady identit pro tento odběr. Jinak příkaz selže s návratovým kódem *MQRCCF_SUBSCRIPTION_IN_USE*. Tím zabráníte změnám atributů odběru, aniž by si to ostatní zainteresovaní odběratelé uvědomovali.

Zadáte-li řetězec znaků delší než 64 znaků, příkaz selže s návratovým kódem *MQRCCF_SUB_IDENTITY_ERROR*.

SubUserData (*MQPSC_SUBSCRIPTION_USER_DATA*)
Jedná se o textový řetězec s proměnnou délkou. Hodnota je uložena správcem front s odběrem, ale nemá žádný vliv na doručení publikování odběrateli. Hodnotu lze změnit opětovnou registrací na stejný odběr s novou hodnotou. Tento atribut je zde pro použití aplikace.

SubUser jsou vrácena v Metatopických informacích (*MQCACF_REG_SUB_USER_DATA*) pro odběr, je-li k dispozici.

Zadáte-li více než jednu z hodnot voleb registrace `NonPers, PersAsPub, PersAsQueue, and Pers`, použije se pouze poslední. Tyto volby nelze kombinovat v jednotlivých odběrech.

Příklad

Zde je příklad `NameValueData` pro zprávu příkazu **`Register Subscriber`** . V ukázkové aplikaci používá služba výsledků tuto zprávu k registraci odběru témat obsahujících nejnovější skóre ve všech shodách s nastavenou volbou 'Trvalé jako publikování'. Identita odběratele, včetně `CorrelId`, je převzata z výchozích hodnot v deskriptoru MQMD.

```
<psc>
  <Command>RegSub</Command>
  <RegOpt>PersAsPub</RegOpt>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

Zpráva o aktualizaci požadavku

Zpráva příkazu **`Request Update`** je odeslána od odběratele správci front s žádostí o aktuální zachovaná publikování pro určené téma a bod odběru, které odpovídají danému (volitelnému) filtru.

Tato zpráva je odeslána do *SYSTEM.BROKER.CONTROL.QUEUE*, řídicí fronta správce front. Kromě oprávnění pro přístup k tématu v aktualizaci požadavku je vyžadováno oprávnění pro vložení zprávy do této fronty. Toto oprávnění nastavuje administrátor systému správce front.

Tento příkaz se obvykle používá, pokud odběratel při registraci zadává volbu `PubOnReqOnly` . Pokud má správce front odpovídající zachovaná publikování, jsou odeslána odběrateli. Pokud správce front nemá odpovídající zachovaná publikování, požadavek selže s návratovým kódem *MQRCCF_NO_RETAINED_MSG*. Žadatel musí mít dříve registrovaný odběr se stejným tématem, SubPointa hodnotami filtru.

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je `ReqUpdate` (*MQPSC_REQUEST_UPDATE*). Tato vlastnost musí být určena.

Téma (*MQPSC_TOPIC*)
Hodnota je téma, které odběratel požaduje; zástupné znaky jsou povoleny.

Tato vlastnost musí být určena, ale v této zprávě je povolen pouze jeden výskyt.

SubPoint (*MQPSC_SUBSCRIPTION_POINT*)
Hodnota je bod odběru, ke kterému je odběr připojen.

Je-li tato vlastnost vynechána, použije se výchozí bod odběru.

Filtr (*MQPSC_FILTER*)
Hodnota je výraz ESQL, který se používá jako filtr pro obsah publikačních zpráv. Pokud se publikování v zadaném tématu shoduje s filtrem, odešle se odběrateli.

Vlastnost `< Filtr >` by měla mít stejnou hodnotu jako vlastnost zadaná v původním odběru, pro který nyní požadujete aktualizaci.

Je-li tato vlastnost vynechána, nebude provedeno žádné filtrování obsahu.

RegOpt (*MQPSC_REGISTRATION_OPTION*)
Vlastnost voleb registrace může mít následující hodnotu:

`CorrelAs`
(*MQPSC_CORREL_ID_AS_IDENTITY*)

`CorrelId` v deskriptoru zprávy (MQMD), který nesmí být nulový, se používá při odesílání odpovídajících publikování do fronty odběratele.

`NONE`
(*MQPSC_NONE*)

Všechny volby mají své výchozí hodnoty. To má stejný efekt jako vynechání vlastnosti <RegOpt> . Pokud jsou současně zadány i jiné volby, volba Žádný se ignoruje.

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

Je-li zadána identita odběratele (fronta, správce front a korelace), není omezena na jedno ID uživatele. To se liší od existujícího chování správce front, který přidružuje ID uživatele původní registrační zprávy k identitě odběratele, a od té doby zabraňuje jakémukoli jinému uživateli, který tuto identitu používá. Pokud se nový odběratel pokusí použít stejnou identitu, příkaz selže s návratovým kódem *MQRCCF_DUPLICATE_SUBSCRIPTION*.

To umožňuje jakémukoli uživateli upravit nebo zrušit registraci odběru, pokud má odpovídající oprávnění. Proto není nutné kontrolovat, zda se ID uživatele shoduje s ID uživatele původního odběratele.

Chcete-li přidat tuto volbu k existujícímu odběru, příkaz musí pocházet ze stejného ID uživatele jako původní odběr.

Má-li odběr příkazu **Request Update** nastaveno VariableUserId , musí být nastaveno v době aktualizace požadavku, aby se označilo, na který odběr se odkazuje. Jinak se k identifikaci odběru použije ID uživatele příkazu **Request Update** . Pokud je zadán název odběru, je tato volba přepsána spolu s dalšími identifikátory odběratele.

Pokud je tato vlastnost vynechána, výchozí nastavení je, že nejsou nastaveny žádné volby registrace.

QMgrName (*MQPSC_Q_MGR_NAME*)
Hodnota je název správce front pro frontu odběratele, do které je odpovídající zachované publikování odesláno správcem front.

Pokud je tato vlastnost vynechána, je výchozí hodnotou název ReplyToQMgr v deskriptoru zprávy (MQMD). Pokud je výsledný název prázdný, standardně se použije název správce front QMgrName.

Název QName (*MQPSC_Q_NAME*)
Hodnota je název fronty odběratele, do které správce front odešle odpovídající zachované publikování.

Je-li tato vlastnost vynechána, výchozí hodnotou je název ReplyToQ v deskriptoru zprávy (MQMD), který v tomto případě nesmí být prázdný.

SubName (*MQPSC_SUBSCRIPTION_NAME*)
Jedná se o název daný konkrétnímu odběru. Je-li zadána v příkazu **Request Update** , má hodnota SubName přednost před všemi ostatními poli identifikátoru kromě ID uživatele, není-li VariableUserId nastaveno na samotný odběr. Pokud není nastaveno VariableUserId , příkaz *Request Update* uspěje pouze v případě, že se ID uživatele zprávy příkazu shoduje s ID uživatele odběru. Pokud ID uživatele zprávy příkazu neodpovídá ID uživatele odběru, příkaz se nezdaří s návratovým kódem *MQRCCF_DUPLICATE_IDENTITY*.

Je-li nastaveno VariableUserId a ID uživatele se liší od ID uživatele odběru, příkaz uspěje, pokud má ID uživatele nové zprávy příkazu oprávnění procházet frontu proudu a vkládat se do fronty odběratele odběru. Jinak příkaz selže s návratovým kódem *MQRCCF_NOT_AUTHORIZED*.

Pokud existuje odběr, který odpovídá tradiční identitě tohoto příkazu, ale nemá žádný SubName, příkaz **Request Update** selže s návratovým kódem *MQRCCF_SUB_NAME_ERROR*.

Pokud se pokusíte požádat o aktualizaci pro odběr, který má SubName , pomocí zprávy příkazu, která odpovídá tradiční identitě, ale není zadán SubName , příkaz uspěje.

Příklad

Zde je příklad NameValueData pro zprávu příkazu **Request Update** . V ukázkové aplikaci používá služba výsledků tuto zprávu k vyžádání zachovaných publikování obsahujících nejnovější skóre pro všechny týmy. Identita odběratele, včetně CorrelId, je převzata z výchozích hodnot v deskriptoru MQMD.

```
<psc>
```

```
  <Command>ReqUpdate</Command>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

Zpráva s odpovědí správce front

Zpráva **Queue Manager Response** je odeslána ze správce front do fronty ReplyToQ vydavatele nebo odběratele, aby označila úspěch nebo selhání zprávy příkazu přijaté správcem front, pokud deskriptor zprávy příkazu uvedl, že je vyžadována odpověď.

Zpráva odpovědi je obsažena v poli NameValueData záhlaví MQRFH2 ve složce <pscr> .

V případě varování nebo chyby obsahuje zpráva odpovědi složku <psc> ze zprávy příkazu i složku <pscr> . Data zprávy, pokud existují, nejsou obsažena ve zprávě s odpovědí správce front. V případě chyby nebyla zpracována žádná zpráva, která způsobila chybu; v případě varování mohla být některá zpráva úspěšně zpracována.

Pokud dojde k selhání při odesílání odpovědi:

- V případě zpráv publikování se správce front pokusí odeslat odpověď do fronty nedoručených zpráv IBM MQ v případě, že operace MQPUT selže. To umožňuje odeslání publikování odběratelům i v případě, že odpověď nelze odeslat zpět vydavateli.
- V případě jiných zpráv nebo v případě, že odpověď publikování nelze odeslat do fronty nedoručených zpráv, je zaprotokolována chyba a zpráva příkazu je obvykle odvolána. To, zda k tomu dojde, závisí na způsobu konfigurace uzlu MQInput.

Vlastnosti

Dokončení (*MQPSCR_COMPLETION*)
Kód dokončení, který může mít jednu ze tří hodnot:

OK
Příkaz byl úspěšně dokončen

varování
Příkaz byl dokončen, ale s varováním

Chyba
Příkaz selhal

Odezva (*MQPSCR_RESPONSE*)
Odezva na zprávu příkazu, pokud tento příkaz vytvořil kód dokončení warning nebo error. Obsahuje vlastnost < Reason> a může obsahovat další vlastnosti, které označují příčinu varování nebo chyby.

V případě jedné nebo více chyb existuje pouze jedna složka odpovědí, která označuje pouze příčinu první chyby. V případě jednoho nebo více varování existuje složka odpovědí pro každé varování.

Příčina (*MQPSCR_REASON*)
Kód příčiny, který kvalifikuje kód dokončení, pokud je kód dokončení varování nebo chyba. Je nastaven na jeden z kódů chyb uvedených v následujícím příkladu. Vlastnost < Reason> je obsažena ve složce < Response> . Za kódem příčiny může následovat jakákoli platná vlastnost ze složky <psc> (například název tématu), která označuje příčinu chyby nebo varování. Pokud dostanete kód příčiny? Zkontrolujte správnost dat, například odpovídající lomené závorky (< >).

Příklady

Zde je několik příkladů NameValueData ve zprávě **Queue Manager Response** . Úspěšná odpověď může být následující:

```
<pscr>
  <Completion>ok</Completion>
</pscr>
```

Zde je příklad odezvy selhání; selhání je chyba filtru. První řetězec NameValueData obsahuje odezvu; druhý obsahuje původní příkaz.

```
<pscr>
  <Completion>error</Completion>
  <Response>
    <Reason>3150</Reason>
  </Reponse>
</pscr>

<psc>
 ...
 command message (to which
 the queue manager is responding)
 ...
</psc>
```

Zde je příklad varovné odpovědi (kvůli neautorizovaným tématům). První řetězec NameValueData obsahuje odpověď; druhý řetězec NameValueData obsahuje původní příkaz.

```
<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>
```

Kódy příčiny publikování/odběru

Tyto kódy příčiny mohou být vráceny do pole Příčina ve složce odezvy publikování/odběru <pscr> . Uvedeny jsou také konstanty, které lze použít k reprezentaci těchto kódů v programovacích jazycích C nebo C + +.

Konstanty MQRC_ vyžadují soubor záhlaví IBM MQ cmqc.h . Konstanty MQRCCF_ vyžadují soubor záhlaví IBM MQ cmqcfc.h (kromě *MQRCCF_FILTER_ERROR* a *MQRCCF_WRONG_USER*, které vyžadují soubor záhlaví cmqpsc.h).

Kód příčiny a text	Vysvětlení	Vydal(a)
2336 MQRC_RFH_COMMAND_CHYBA	Platné hodnoty pro pole < Command> složky <psc> jsou: RegSub, DeregSub, Publikovat, DeletePuba ReqUpdate. Všechny ostatní hodnoty vedou k vydání tohoto kódu chyby.	Libovolný příkaz
2337 MQRC_RFH_ PARM_ERROR	Složky <psc> a <mcd> mají sadu platných parametrů, které lze v nich zadat. Zkontrolujte popisy těchto složek a ujistěte se, že jste nezadali nesprávné parametry.	Libovolný příkaz

Kód příčiny a text	Vysvětlení	Vydal(a)
2338 MQRC_RFH_DUPLICATE_PARM	Některé parametry (například Téma) ve složce <psc> se mohou opakovat, ale jiné (například Příkaz) se opakovat nemohou. Zkontrolujte, zda jste neduplikovali neopakovatelný parametr.	Libovolný příkaz
2339 MQRC_RFH_PARM_MISSING	Některé parametry ve složkách <psc> nebo <mcd> jsou volitelné a lze je vynechat; některé jsou povinné a nesmí být vynechány. Zkontrolujte, zda jste zahrnuli všechny povinné parametry do složek <psc> a <mcd> .	Libovolný příkaz
2551 MQRC_SELECTION_NOT_AVAILABLE	Nebyl k dispozici žádný poskytovatel rozšířeného výběru zpráv, který by určil, kteří odběratelé s uvedeným filtrem by měli obdržet publikování.	Publikovat, registrovat odběratele a požadovat aktualizaci
	Nebyl k dispozici žádný poskytovatel rozšířeného výběru zpráv pro zpracování filtru uvedeného odběratele.	Registrovat odběratele a požádat o aktualizaci
2554 MQRC_CONTENT_ERROR	Poskytovatel rozšířeného výběru zpráv nalezl chybu v aktuálním nebo zachovaném publikování.	Publikovat a požadovat aktualizaci
3008 MQRCCF_COMMAND_FAILED	Došlo k interní chybě, která zabránila správnému provedení příkazu. K chybě může dojít, pokud je příkaz znovu vydán. Protokol systémových událostí pro správce front obsahuje informace, které by měly být použity při ohlašování problému do adresáře IBM.	Libovolný příkaz
3072 MQRCCF_TOPIC_ERROR	Jedna nebo více hodnot, které jste zadali pro parametr Topic, jsou nesprávné. Zkontrolujte, zda vaše hodnoty pro téma odpovídají uvedeným omezením.	Libovolný příkaz
3073 MQRCCF_NOT_REGISTERED	Kombinace položek SubPoint, Téma a Filtr, které jste zadali v příkazu DeregSub nebo ReqUpdate , buď nebyla kombinací, se kterou jste již zaregistrovali, nebo pro příkaz DeregSub , pokud byla zadána volba DeregAll , nebyla jedna z vlastností SubPoint, Téma nebo Filtr použita ke zrušení registrace žádného odběru.	Příkazy pro zrušení registrace odběratele a vyžádání aktualizace
3074 MQRCCF_Q_MGR_NAME_ERROR	Určený správce front nebyl platný nebo nebyl k dispozici nebo neexistoval.	Příkazy pro zrušení registrace odběratele, publikování, registraci odběratele a vyžádání aktualizace

Kód příčiny a text	Vysvětlení	Vydal(a)
3076 MQRCCF_Q_NAME_ERROR	Zadaný název fronty byl neplatný nebo fronta v zadaném správci front neexistovala.	Příkazy pro zrušení registrace odběratele, publikování, registraci odběratele a vyžádání aktualizace
3077 MQRCCF_NO_RETAINED_MSG	Pro zadané téma nebyly zachovány žádné zprávy. V závislosti na návrhu aplikačního programu se může nebo nemusí jednat o chybu.	Příkaz pro aktualizaci požadavku
3079 MQRCCF_INCORRECT_Q	Příkazy RegSub, DeregSuba ReqUpdate jsou vždy odeslány do SYSTEM.BROKER.CONTROL.QUEUE správce front, pro kterého jsou určeny. Příkazy publikování a odstranění jsou odesílány do vstupní fronty pro konkrétní tok zpráv publikování/odběru, pro který jsou určeny. Toto je určeno při návrhu toku zpráv. Tento kód chyby je vrácen, pokud je příkaz odeslán do nesprávné fronty.	Libovolný příkaz
3080 MQRCCF_CORREL_ID_ERROR	Uvedli jste ID CorrelAsjako jeden z parametrů RegOpt . Pole CorrelId deskriptoru MQMD však neobsahuje platný identifikátor korelace (tj. je nastaveno na hodnotu MQCI_NONE).	Příkazy pro zrušení registrace odběratele a registraci odběratele
3081 MQRCCF_NOT_AUTHORIZED	Nemáte autorizaci k provedení požadované akce. Nastavení autorizace pro správce front zpracovává administrátor systému pomocí editoru Hierarchie témat.	Příkazy pro publikování a registraci odběratele
3083 MQRCCF_REG_OPTIONS_ERROR	Zadali jste nerozpoznaný parametr RegOpt ve složce <psc> , která obsahuje příkaz RegSub nebo DeregSub .	Příkazy pro zrušení registrace odběratele a registraci odběratele
3084 MQRCCF_PUB_OPTIONS_ERROR	Zadali jste nerozpoznaný parametr PubOpt ve složce <psc> , která obsahuje příkaz Publikovat.	Příkaz pro publikování
3087 MQRCCF_DEL_OPTIONS_ERROR	Zadali jste nerozpoznaný parametr DelOpt ve složce <psc> , která obsahuje příkaz DeletePub .	Příkaz Odstranit publikování
3150 MQRCCF_FILTER_ERROR	Hodnota uvedená pro parametr Filtr není platná. Zkontrolujte sekci, která popisuje platnou syntaxi pro výrazy filtru, a ujistěte se, že váš výraz odpovídá.	Příkazy pro zrušení registrace odběratele, pro registraci odběratele a pro vyžádání aktualizace

Kód příčiny a text	Vysvětlení	Vydal(a)
3151 MQRCCF_WRONG_USER	Odběr, který odpovídá zadanému odběru, již existuje; byl však registrován jiným uživatelem. Odběr může změnit nebo zrušit pouze uživatel, který jej původně registroval.	Příkazy pro zrušení registrace odběratele, pro registraci odběratele a pro vyžádání aktualizace
3152 MQRCCF_DUPLICITNÍ_ODBĚR	Odpovídající odběr již existuje s jiným názvem odběru.	
3153 MQRCCF_SUB_NAME_ERROR	Buď není platný formát názvu odběru, nebo již existuje odpovídající odběr bez názvu odběru.	
3154 MQRCCF_SUB_IDENTITY_ERROR	Parametr identity odběru je chybný. Buď zadaná hodnota překračuje maximální povolenou délku, nebo identita odběru není v současné době členem sady identit odběru a nebyla zadána volba registrace spojení.	
3155 MQRCCF_SUBSCRIPTION_IN_USE	Došlo k pokusu o úpravu nebo zrušení registrace odběru členem sady identit, který nebyl jediným členem této sady.	
3156 MQRCCF_SUBSCRIPTION_LOCKED	Odběr je v současné době výhradně uzamčen jinou identitou.	
3157 MQRCCF_ALREADY_JOINED	Byla zadána volba registrace spojení, ale identita odběratele již byla členem sady identit odběru.	

Nastavení MQMD ve zprávách příkazů pro správce front

Aplikace, které odesílají zprávy příkazů do správce front, používají následující nastavení polí v deskriptoru zpráv (MQMD). Pole, která jsou ponechána jako výchozí hodnota, nebo je lze nastavit na libovolnou platnou hodnotu obvyklým způsobem, zde nejsou uvedena.

Sestava
Viz `MsgType` a `CorrelId`.

MsgType
Parametr `MsgType` by měl být nastaven na hodnotu *MQMT_REQUEST* nebo *MQMT_DATAGRAM*. Hodnota *MQRC_MSG_TYPE_ERROR* bude vrácena, pokud `MsgType` není nastavena na jednu z těchto hodnot.

Hodnota `MsgType` by měla být pro zprávu příkazu nastavena na *MQMT_REQUEST* , pokud je odpověď vždy požadována. Příznaky MQRO_PAN a MQRO_NAN v poli `Sestava` nejsou v tomto případě významné.

Je-li volba `MsgType` nastavena na hodnotu *MQMT_DATAGRAM*, závisí odpovědi na nastavení příznaků MQRO_PAN a MQRO_NAN v poli `Sestava` :

- Samotný MQRO_PAN znamená, že správce front odešle odezvu pouze v případě, že je příkaz úspěšný.
- Samotný MQRO_NAN znamená, že správce front odešle odpověď pouze v případě, že příkaz selže.
- Pokud je příkaz dokončen s varováním, odešle se odezva, pokud je nastaven buď MQRO_PAN, nebo MQRO_NAN.

- MQRO_PAN + MQRO_NAN znamená, že správce front odešle odpověď, zda příkaz uspěje nebo selže. To má stejný efekt z perspektivy správce front jako nastavení MsgType na MQMT_REQUEST.
- Není-li nastavena hodnota MQRO_PAN ani MQRO_NAN, není nikdy odeslána žádná odpověď.

Formát
Nastavit na hodnotu MQFMT_RF_HEADER_2

MsgId
Toto pole je obvykle nastaveno na hodnotu MQMI_NONE, takže správce front vygeneruje jedinečnou hodnotu.

CorrelId
Toto pole lze nastavit na libovolnou hodnotu. Pokud identita odesilatele obsahuje CorrelId, zadejte tuto hodnotu společně s hodnotou MQRO_PASS_CORREL_ID v poli Sestava , abyste se ujistili, že je nastavena ve všech zprávách odpovědí odeslaných správcem front odesilateli.

ReplyToQ
Toto pole definuje frontu, do které mají být odesílány případné odpovědi. Může se jednat o frontu odesilatele. To má tu výhodu, že parametr QName lze ze zprávy vynechat. Pokud však mají být odpovědi odeslány do jiné fronty, je třeba zadat parametr QName .

ReplyToQMgr
Toto pole definuje správce front pro odpovědi. Ponecháte-li toto pole prázdné (výchozí hodnota), lokální správce front vloží do tohoto pole svůj vlastní název.

Nastavení MQMD pro publikování předaná správcem front

Správce front používá tato nastavení polí v deskriptoru zpráv (MQMD) při odesílání publikování odběrateli. Všechna ostatní pole v deskriptoru MQMD jsou nastavena na výchozí hodnoty.

Sestava
Sestava je nastavena na hodnotu MQRO_NONE.

MsgType
MsgType je nastaven na MQMT_DATAGRAM.

Vypršení
Vypršení platnosti je nastaveno na hodnotu ve zprávě Publikovat přijaté od vydavatele. V případě uchované zprávy se zbývající čas zkracuje o přibližný čas, kdy byla zpráva ve správci front.

Formát
Formát je nastaven na hodnotu MQFMT_RF_HEADER_2

MsgId
MsgId je nastaven na jedinečnou hodnotu.

CorrelId
Pokud je CorrelId součástí identity odběratele, jedná se o hodnotu určenou odběratelem při registraci. Jinak se jedná o nenulovou hodnotu zvolenou správcem front.

Priorita
Priorita přebírá hodnotu nastavenou vydavatelem nebo jako vyřešenou, pokud vydavatel zadá hodnotu MQPRI_PRIORITY_AS_Q_DEF.

Trvání
Perzistence přebírá hodnotu nastavenou vydavatelem nebo vyhodnocenou v případě, že vydavatel zadal hodnotu MQPER_PERSISTENCE_AS_Q_DEF, není-li ve zprávě Registrovat odběratele pro odběratele, kterému je toto publikování odesíláno, uvedeno jinak.

ReplyToQ
ReplyToQ je nastaveno na mezery.

ReplyToQMgr
ReplyToQMgr je nastaven na název správce front.

UserIdentifier
UserIdentifier je identifikátor uživatele odběratele nastavený při registraci odběratele.

AccountingToken
AccountingToken je účtovací token odběratele nastavený při první registraci odběratele.

ApplIdentityData
ApplIdentityData jsou data identity aplikace odběratele, která jsou nastavena při první registraci odběratele.

PutApplType
PutApplTyp je nastaven na hodnotu MQAT_BROKER.

PutApplName
PutApplNázev je nastaven na prvních 28 znaků názvu správce front.

PutDate
PutDate je datum, kdy byla zpráva vložena.

PutTime
PutTime je čas, kdy byla zpráva vložena.

ApplOriginData
Volba ApplOriginData je nastavena na mezery.

Nastavení MQMD ve zprávách odpovědí správce front

Správce front používá tato nastavení polí v deskriptoru zpráv (MQMD) při odesílání odpovědi na zprávu publikování. Všechna ostatní pole v deskriptoru MQMD jsou nastavena na výchozí hodnoty.

Sestava
Sestava je nastavena na všechny nuly.

MsgType
Parametr MsgType je nastaven na hodnotu MQMT_REPLY.

Formát
Formát je nastaven na hodnotu MQFMT_RF_HEADER_2

MsgId
Nastavení MsgId závisí na volbách Report v původní zprávě příkazu. Standardně je nastaven na MQMI_NONE, takže správce front vygeneruje jedinečnou hodnotu.

CorrelId
Nastavení parametru CorrelId závisí na volbách Sestava v původní zprávě příkazu. Standardně to znamená, že CorrelId je nastaveno na stejnou hodnotu jako MsgId zprávy příkazu. To lze použít ke korelaci příkazů s jejich odpověďmi.

Priorita
Priorita je nastavena na stejnou hodnotu jako v původní zprávě příkazu.

Trvání
Perzistence je nastavena na hodnotu nastavenou v původní zprávě příkazu.

Vypršení
Vypršení platnosti je nastaveno na stejnou hodnotu jako v původní zprávě příkazu přijaté správcem front.

PutApplType
PutApplTyp je nastaven na hodnotu MQAT_BROKER.

PutApplName
PutApplNázev je nastaven na prvních 28 znaků názvu správce front.

Ostatní kontextová pole jsou nastavena tak, jako by byla generována s MQPMO_PASS_IDENTITY_CONTEXT.

Strojové kódování

Tento oddíl popisuje strukturu pole *Encoding* v deskriptoru zprávy.

Souhrn polí ve struktuře viz “MQMD-Deskriptor zpráv” na stránce 424 .

Pole *Encoding* je 32bitové celé číslo, které je rozděleno do čtyř samostatných dílčích polí; tato dílčí pole identifikují:

- Kódování použité pro binární celá čísla
- Kódování použité pro pakovaná desetinná celá čísla
- Kódování použité pro čísla s pohyblivou řádovou čárkou
- Vyhrazené bity

Každé dílčí pole je identifikováno bitovou maskou, která má 1 bity na pozicích odpovídajících podpoli, a 0 bity jinde. Bity jsou očíslovány tak, že bit 0 je nejvýznamnější bit, a bit 31 nejméně významný bit. Jsou definovány následující masky:

MASKA MQENC_INTEGER_MASK
Maska pro kódování binárních čísel.

Toto podpole zabírá pozice bitů 28 až 31 v poli *Encoding* .

MQENC_DECIMAL_MASK
Maska pro kódování packed-decimal-integer.

Toto podpole zabírá bitové pozice 24 až 27 v poli *Encoding* .

MQENC_FLOAT_MASK
Maska pro kódování s pohyblivou řádovou čárkou.

Toto podpole zabírá pozice bitů 20 až 23 v poli *Encoding* .

MQENC_RESERVED_MASK
Maska pro vyhrazené bity.

Toto podpole zabírá pozice bitů 0 až 19 v poli *Encoding* .

Kódování binárních čísel

Následující hodnoty jsou platné pro kódování binárního celého čísla:

MQENC_INTEGER_UNDEFINED
Binární celá čísla jsou reprezentována pomocí nedefinovaného kódování.

MQENC_INTEGER_NORMAL
Binární celá čísla jsou reprezentována konvenčním způsobem:

- Nejméně významný bajt v čísle má nejvyšší adresu libovolného z bajtů v čísle; nejvýznamnější bajt má nejnižší adresu
- Nejméně významný bit v každém bajtu je sousedící s bajtem s další vyšší adresou; nejvýznamnější bit v každém bajtu je sousedící s bajtem s další nižší adresou

MQENC_INTEGER_REVERSED
Binární celá čísla jsou reprezentována stejným způsobem jako MQENC_INTEGER_NORMAL, ale s bajty uspořádanými v opačném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako MQENC_INTEGER_NORMAL.

Kódování packed-decimal-integer

Následující hodnoty jsou platné pro kódování packed-decimal-integer:

MQENC_DECIMAL_UNDEFINED
Celá čísla ve formátu packed-decimal jsou reprezentována nedefinovaným kódováním.

MQENC_DECIMAL_NORMAL
Pakovaná desetinná celá čísla jsou reprezentována konvenčním způsobem:

- Každá desetinná číslice v tisknutelném tvaru čísla je reprezentována v desítkovém balení jednou hexadecimální číslicí v rozsahu X' 0 'až X' 9'. Každá hexadecimální číslice zabírá čtyři bity, a tak každý bajt v pakovaném desítkovém čísle představuje dvě desetinná čísla v tisknutelné podobě čísla.

- Nejméně významný bajt v pakovaných desetinných číslech je bajt, který obsahuje nejméně významnou desetinnou číslici. V rámci tohoto bajtu nejvýznamnější čtyři bity obsahují nejméně významnou desetinnou číslici a nejméně významné čtyři bity obsahují znaménko. Znaménko je buď X'C '(kladné), X 'D' (záporné), nebo X'F ' (nepodepsané).
- Nejméně významný bajt v čísle má nejvyšší adresu libovolného z bajtů v čísle; nejvýznamnější bajt má nejnižší adresu.
- Nejméně významný bit v každém bajtu je sousedící s bajtem s další vyšší adresou; nejvýznamnější bit v každém bajtu je sousedící s bajtem s další nižší adresou.

MQENC_DECIMAL_REVERSED
Pakovaná desetinná celá čísla jsou reprezentována stejným způsobem jako MQENC_DECIMAL_NORMAL, ale s bajty uspořádanými v obráceném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako MQENC_DECIMAL_NORMAL.

Kódování s pohyblivou řádovou čárkou

Následující hodnoty jsou platné pro kódování s pohyblivou řádovou čárkou:

MQENC_FLOAT_UNDEFINED
Čísla s pohyblivou řádovou čárkou jsou reprezentována pomocí nedefinovaného kódování.

MQENC_FLOAT_IEEE_NORMAL
Čísla s pohyblivou řádovou čárkou jsou reprezentována pomocí standardu IEEE.[4]formát s pohyblivou řádovou čárkou, s bajty uspořádanými takto:

- Nejméně významný bajt v mantise má nejvyšší adresu libovolného z bajtů v čísle; bajt obsahující exponent má nejnižší adresu
- Nejméně významný bit v každém bajtu je sousedící s bajtem s další vyšší adresou; nejvýznamnější bit v každém bajtu je sousedící s bajtem s další nižší adresou

Podrobnosti o kódování typu float IEEE lze nalézt v normě IEEE 754.

MQENC_FLOAT_IEEE_REVERSED
Čísla s pohyblivou řádovou čárkou jsou reprezentována stejným způsobem jako MQENC_FLOAT_IEEE_NORMAL, ale s bajty uspořádanými v opačném pořadí. Bity v rámci každého bajtu jsou uspořádány stejným způsobem jako MQENC_FLOAT_IEEE_NORMAL.

MQENC_FLOAT_S390
Čísla s pohyblivou řádovou čárkou jsou reprezentována pomocí standardního formátu System/390 s pohyblivou řádovou čárkou. Tento formát používá také systém System/370.

Konstrukce kódování

Chcete-li vytvořit hodnotu pro pole *Encoding* v deskriptoru MQMD, lze příslušné konstanty, které popisují požadované kódování, sečíst (nepřidávat stejnou konstantu více než jednou) nebo kombinovat pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Bez ohledu na použitou metodu zkombinujte pouze jedno z kódování MQENC_INTEGER_ * s jedním z kódování MQENC_DECIMAL_ * a jedním z kódování MQENC_FLOAT_ *.

Analýza kódování

Pole *Encoding* obsahuje dílčí pole; z tohoto důvodu musí aplikace, které potřebují prozkoumat celé číslo, pakované desetinné číslo nebo kódování s pohyblivou řádovou čárkou, použít jednu z popsaných metod.

Použití bitových operací

Pokud programovací jazyk podporuje bitové operace, postupujte takto:

[4] Ústav elektrických a elektronických inženýrů

1. Vyberte jednu z následujících hodnot podle typu požadovaného kódování:
   - MQENC_INTEGER_MASK pro kódování binárního celého čísla
   - MQENC_DECIMAL_MASK pro kódování pakovaného desetinného celého čísla
   - MQENC_FLOAT_MASK pro kódování s pohyblivou řádovou čárkou

   Zavolejte hodnotu A.
2. Zkombinujte pole *Encoding* s polem A pomocí bitové operace AND; zavolejte výsledek B.
3. B je požadované kódování a lze jej testovat na shodu s každou z hodnot, které jsou platné pro daný typ kódování.

Použití aritmetické

Pokud programovací jazyk *nepodporuje* bitové operace, proveďte následující kroky s použitím celočíselné aritmetiky:

1. Vyberte jednu z následujících hodnot podle typu požadovaného kódování:
   - 1 pro kódování binárních celých čísel
   - 16 pro kódování pakovaného desetinného celého čísla
   - 256 pro kódování s pohyblivou řádovou čárkou

   Zavolejte hodnotu A.
2. Vydělte hodnotu pole *Encoding* hodnotou A ; Zavolejte výsledek B.
3. Vydělte B 16; vyvolejte výsledek C.
4. Vynásobte C číslem 16 a odečtěte od B ; Zavolejte výsledek D.
5. Vynásobte D hodnotou A ; Zavolejte výsledek E.
6. E je požadované kódování a lze jej testovat na shodu s každou z hodnot, které jsou platné pro daný typ kódování.

Souhrn kódování architektury počítače

Kódování pro architektury počítačů jsou uvedeny v části Tabulka 631 na stránce 901.

Tabulka 631. Souhrn kódování pro architektury počítačů

Architektura počítače	Kódování binárních celých čísel	Kódování desetinných celých čísel	Kódování čísel s pohyblivou řádovou čárkou
IBM i	Normální	Normální	IEEE normální
Intel x86	Převrácené	Převrácené	IEEE obráceno
PowerPC	Normální	Normální	IEEE normální
System/390	Normální	Normální	System/390

Volby sestavy a příznaky zpráv

Tento oddíl popisuje pole *Report* a *MsgFlags* , která jsou součástí deskriptoru zprávy MQMD určeného ve voláních MQGET, MQPUT a MQPUT1 .

Témata v této části popisují:

- Struktura pole sestavy a způsob, jakým ji správce front zpracovává
- Jak aplikace analyzuje pole sestavy
- Struktura pole příznaků zprávy

Další informace o deskriptoru zpráv MQMD viz “MQMD-Deskriptor zpráv” na stránce 424.

Struktura pole sestavy

Tyto informace popisují strukturu pole sestavy.

Pole *Report* je 32bitové celé číslo, které je rozděleno do tří samostatných dílčích polí. Tato podpole identifikují:

- Volby sestavy, které jsou odmítnuty, pokud je lokální správce front nerozpozná
- Volby sestavy, které jsou vždy přijaty, i když je lokální správce front nerozpozná
- Volby sestavy, které jsou přijaty pouze v případě, že jsou splněny určité další podmínky

Každé dílčí pole je identifikováno bitovou maskou, která má 1 bity na pozicích odpovídajících podpoli, a 0 bity jinde. Bity v podpoli nemusí nutně sousedící. Bity jsou očíslovány tak, že bit 0 je nejvýznamnější bit, a bit 31 nejméně významný bit. Pro identifikaci dílčích polí jsou definovány následující masky:

MQRO_REJECT_UNSUP_MASK
Tato maska identifikuje bitové pozice v poli *Report* , kde volby sestavy, které nejsou podporovány lokálním správcem front, způsobí selhání volání MQPUT nebo MQPUT1 s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_REPORT_OPTIONS_ERROR.

Toto podpole zabírá bitové pozice 3 a 11 až 13.

MQRO_ACCEPT_UNSUP_MASK
Tato maska identifikuje bitové pozice v poli *Report* , kde jsou volby sestavy, které nejsou podporovány lokálním správcem front, přesto přijímány ve volání MQPUT nebo MQPUT1 . V tomto případě je vrácen kód dokončení MQCC_WARNING s kódem příčiny MQRC_UNKNOWN_REPORT_OPTION.

Toto podpole zabírá bitové pozice 0 až 2, 4 až 10 a 24 až 31.

V tomto podpoli jsou zahrnuty následující volby sestavy:

- MQRO_ACTIVITY
- MQRO_COPY_MSG_ID_TO_CORREL_ID
- MQRO_DEAD_LETTER_Q
- MQRO_DISCARD_MSG
- MQRO_EXCEPTION
- MQRO_EXCEPTION_WITH_DATA
- MQRO_EXCEPTION_WITH_FULL_DATA
- MQRO_EXPIRATION
- MQRO_EXPIRATION_WITH_DATA
- MQRO_EXPIRATION_WITH_FULL_DATA
- MQRO_NAN
- MQRO_NEW_MSG_ID
- MQRO_NONE
- MQRO_PAN
- MQRO_PASS_CORREL_ID
- MQRO_PASS_MSG_ID

MQRO_ACCEPT_UNSUP_IF_XMIT_MASK
Tato maska identifikuje bitové pozice v poli *Report* , kde jsou volby sestavy, které nejsou podporovány lokálním správcem front, přesto přijímány ve volání MQPUT nebo MQPUT1 *za předpokladu* , že jsou splněny obě následující podmínky:

- Zpráva je určena pro vzdáleného správce front.

- Aplikace nevkládá zprávu přímo do lokální přenosové fronty (tj. fronta určená poli *ObjectQMgrName* a *ObjectName* v deskriptoru objektu určeném ve volání MQOPEN nebo MQPUT1 není lokální přenosovou frontou).

Kód dokončení MQCC_WARNING s kódem příčiny MQRC_UNKNOWN_REPORT_OPTION jsou vráceny, pokud jsou tyto podmínky splněny, a MQCC_FAILED s kódem příčiny MQRC_REPORT_OPTIONS_ERROR, pokud ne.

Toto podpole zabírá bitové pozice 14 až 23.

V tomto podpoli jsou zahrnuty následující volby sestavy:

- MQRO_COA
- MQRO_COA_WITH_DATA
- MQRO_COA_WITH_FULL_DATA
- MQRO_COD
- MQRO_COD_WITH_DATA
- MQRO_COD_WITH_FULL_DATA

Pokud jsou v poli *Report* zadány nějaké volby, které správce front nerozpozná, správce front postupně zkontroluje jednotlivá dílčí pole pomocí bitové operace AND a zkombinuje pole *Report* s maskou pro dané dílčí pole. Není-li výsledek této operace nulový, vrátí se dříve popsaný kód dokončení a kódy příčiny.

Je-li vrácena hodnota MQCC_WARNING, není definováno, který kód příčiny je vrácen v případě, že existují jiné varovné podmínky.

Schopnost určit a mít přijaté volby sestavy, které nejsou rozpoznány lokálním správcem front, je užitečná při odesílání zprávy s volbou sestavy, která je rozpoznána a zpracována *vzdáleným* správcem front.

Analýza pole sestavy

Pole *Report* obsahuje dílčí pole; z tohoto důvodu musí aplikace, které potřebují zkontrolovat, zda odesilatel zprávy požadoval určitou sestavu, použít jednu z popsaných technik.

Použití bitových operací

Pokud programovací jazyk podporuje bitové operace, postupujte takto:

1. Vyberte jednu z následujících hodnot podle typu sestavy, která se má zkontrolovat:
   - MQRO_COA_WITH_FULL_DATA pro sestavu COA
   - MQRO_COD_WITH_FULL_DATA pro sestavu COD
   - MQRO_EXCEPTION_WITH_FULL_DATA pro sestavu výjimek
   - MQRO_EXPIRATION_WITH_FULL_DATA pro sestavu vypršení platnosti

   Zavolejte hodnotu A.

   V systému z/OSpoužijte hodnoty MQRO_ * _WITH_DATA místo hodnot MQRO_ * _WITH_FULL_DATA.
2. Zkombinujte pole *Report* s polem A pomocí bitové operace AND; zavolejte výsledek B.
3. Otestujte B pro shodu s každou hodnotou, která je možná pro daný typ sestavy.

   Pokud je například A MQRO_EXCEPTION_WITH_FULL_DATA, otestujte B shodu s každým z následujících způsobů, abyste určili, co bylo určeno odesilatelem zprávy:

   - MQRO_NONE
   - MQRO_EXCEPTION
   - MQRO_EXCEPTION_WITH_DATA
   - MQRO_EXCEPTION_WITH_FULL_DATA

   Testy lze provádět v jakémkoli pořadí, které je pro logiku aplikace nejvhodnější.

Použijte podobnou metodu k testování pro volby MQRO_PASS_MSG_ID nebo MQRO_PASS_CORREL_ID; vyberte jako hodnotu A , podle toho, která z těchto dvou konstant je vhodná, a pak pokračujte podle výše uvedeného popisu.

Použití aritmetické

Pokud programovací jazyk *nepodporuje* bitové operace, proveďte následující kroky s použitím celočíselné aritmetiky:

1. Vyberte jednu z následujících hodnot podle typu sestavy, která se má zkontrolovat:
   - MQRO_COA pro sestavu COA
   - MQRO_COD pro sestavu COD
   - MQRO_EXCEPTION pro sestavu výjimek
   - MQRO_EXPIRATION pro sestavu vypršení platnosti

   Zavolejte hodnotu A.
2. Pole *Report* vydělte hodnotou A ; Zavolejte výsledek B.
3. Vydělte B hodnotou 8 ; Zavolejte výsledek C.
4. Vynásobte C hodnotou 8 a odečtěte od B ; Zavolejte výsledek D.
5. Vynásobte D hodnotou A ; Zavolejte výsledek E.
6. Otestujte E pro shodu s každou hodnotou, která je možná pro daný typ sestavy.

   Pokud je například A MQRO_EXCEPTION, otestujte E pro shodu s každým z následujících způsobů, abyste určili, co bylo určeno odesilatelem zprávy:
   - MQRO_NONE
   - MQRO_EXCEPTION
   - MQRO_EXCEPTION_WITH_DATA
   - MQRO_EXCEPTION_WITH_FULL_DATA

   Testy lze provádět v jakémkoli pořadí, které je pro logiku aplikace nejvhodnější.

Následující pseudokód ilustruje tuto techniku pro zprávy hlášení výjimek:

```
A = MQRO_EXCEPTION
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

Chcete-li testovat volby MQRO_PASS_MSG_ID nebo MQRO_PASS_CORREL_ID, použijte podobnou metodu. Vyberte hodnotu A podle toho, která z těchto dvou konstant je vhodná, a pak pokračujte podle popisu výše, ale nahraďte hodnotu 8 v předchozích krocích hodnotou 2.

Struktura pole příznaků zprávy

Tyto informace popisují strukturu pole příznaků zprávy.

Pole *MsgFlags* je 32bitové celé číslo, které je rozděleno do tří samostatných dílčích polí. Tato podpole identifikují:

- Příznaky zpráv, které jsou odmítnuty, pokud je lokální správce front nerozpozná
- Příznaky zpráv, které jsou vždy přijaty, a to i v případě, že je lokální správce front nerozpozná.
- Příznaky zprávy, které jsou přijaty pouze v případě, že jsou splněny určité další podmínky

Poznámka: Všechna dílčí pole v souboru *MsgFlags* jsou vyhrazena pro použití správcem front.

Každé dílčí pole je identifikováno bitovou maskou, která má 1 bity na pozicích odpovídajících podpoli, a 0 bity jinde. Bity jsou očíslovány tak, že bit 0 je nejvýznamnější bit, a bit 31 nejméně významný bit. Pro identifikaci dílčích polí jsou definovány následující masky:

MQMF_REJECT_UNSUP_MASK
Tato maska identifikuje bitové pozice v poli *MsgFlags* , kde příznaky zpráv, které nejsou podporovány lokálním správcem front, způsobí selhání volání MQPUT nebo MQPUT1 s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_MSG_FLAGS_ERROR.

Toto podpole zabírá bitové pozice 20 až 31.

V tomto dílčím poli jsou zahrnuty následující příznaky zprávy:

- MQMF_LAST_MSG_IN_GROUP
- MQMF_LAST_SEGMENT
- MQMF_MSG_IN_GROUP
- MQMF_SEGMENT
- MQMF_SEGMENTATION_ALLOWED
- MQMF_SEGMENTATION_INHIBITED

MQMF_ACCEPT_UNSUP_MASK
Tato maska identifikuje bitové pozice v poli *MsgFlags* , kde jsou příznaky zpráv, které nejsou podporovány lokálním správcem front, přesto přijímány ve volání MQPUT nebo MQPUT1 . Kód dokončení je MQCC_OK.

Toto podpole zabírá bitové pozice 0 až 11.

MQMF_ACCEPT_UNSUP_IF_XMIT_MASK
Tato maska identifikuje bitové pozice v poli *MsgFlags* , kde jsou příznaky zpráv, které nejsou podporovány lokálním správcem front, přesto přijímány ve volání MQPUT nebo MQPUT1 *za předpokladu* , že jsou splněny obě následující podmínky:

- Zpráva je určena pro vzdáleného správce front.
- Aplikace nevkládá zprávu přímo do lokální přenosové fronty (tj. fronta určená poli *ObjectQMgrName* a *ObjectName* v deskriptoru objektu určeném ve volání MQOPEN nebo MQPUT1 není lokální přenosovou frontou).

Kód dokončení MQCC_OK je vrácen, pokud jsou tyto podmínky splněny, a MQCC_FAILED s kódem příčiny MQRC_MSG_FLAGS_ERROR, pokud není.

Toto podpole zabírá bitové pozice 12 až 19.

Pokud jsou v poli *MsgFlags* uvedeny příznaky, které správce front nerozpozná, správce front postupně zkontroluje jednotlivá dílčí pole pomocí bitové operace AND a zkombinuje pole *MsgFlags* s maskou pro dané dílčí pole. Není-li výsledek této operace nulový, vrátí se dříve popsaný kód dokončení a kódy příčiny.

Uživatelská procedura převodu dat

Tato kolekce témat popisuje rozhraní uživatelské procedury pro převod dat a zpracování prováděné správcem front v případě, že je vyžadován převod dat.

Další informace o převodu dat viz *Převod dat v části IBM MQ* na adrese https://www.ibm.com/support/pages/node/317869.

Uživatelská procedura převodu dat je vyvolána jako součást zpracování volání MQGET za účelem převodu dat zprávy aplikace na reprezentaci vyžadovanou přijímající aplikací. Převod dat zprávy aplikace je volitelný; vyžaduje zadání volby MQGMO_CONVERT ve volání MQGET.

Jsou popsány následující předměty:

- Zpracování provedené správcem front v reakci na volbu MQGMO_CONVERT; viz “Zpracování převodu” na stránce 906.

- Konvence zpracování používané správcem front při zpracování vestavěného formátu. Tyto konvence se doporučují i pro uživatelské procedury napsané uživatelem. Viz "Konvence zpracování" na stránce 907.
- Speciální pokyny pro převod zpráv sestavy; viz "Převod zpráv sestavy" na stránce 911.
- Parametry předané uživatelské proceduře pro převod dat; viz "MQ_DATA_CONV_EXIT-uživatelská procedura převodu dat" na stránce 923.
- Volání, které lze použít z uživatelské procedury pro převod znakových dat mezi různými reprezentacemi; viz "MQXCNVC-Převod znaků" na stránce 917.
- Parametr datové struktury, který je specifický pro uživatelskou proceduru; viz "MQDXP-Parametr uživatelské procedury pro převod dat" na stránce 912.

Zpracování převodu

Tyto informace popisují zpracování provedené správcem front v reakci na volbu MQGMO_CONVERT.

Správce front provádí následující akce, pokud je ve volání MQGET určena volba MQGMO_CONVERT a existuje zpráva, která má být vrácena aplikaci:

1. Je-li splněna jedna nebo více následujících podmínek, není převod nutný:
   - Data zprávy jsou již ve znakové sadě a kódování vyžadované aplikací vydávající volání MQGET. Aplikace musí před zadáním volání nastavit pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** volání MQGET na požadované hodnoty.
   - Délka dat zprávy je nula.
   - Délka parametru **Buffer** volání MQGET je nula.

   V těchto případech je zpráva vrácena bez převodu do aplikace vydávající volání MQGET; hodnoty *CodedCharSetId* a *Encoding* v parametru **MsgDesc** jsou nastaveny na hodnoty v řídicích informacích ve zprávě a volání je dokončeno jednou z následujících kombinací kódu dokončení a kódu příčiny:

Tabulka 632. Kombinace kódu dokončení a kódu příčiny

Kód dokončení	Kód příčiny
MQCC_OK	MQRC_NONE
MQCC_VAROVÁNÍ	MQRC_TRUNCATED_MSG_ACCEPTED
MQCC_VAROVÁNÍ	MQRC_TRUNCATED_MSG_FAILED

   Následující kroky jsou provedeny pouze v případě, že se znaková sada nebo kódování dat zprávy liší od odpovídající hodnoty v parametru **MsgDesc** a existují data, která mají být převedena:
2. Má-li pole *Format* v řídicích informacích ve zprávě hodnotu MQFMT_NONE, vrátí se zpráva nepřevedená s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_FORMAT_ERROR.

   Ve všech ostatních případech zpracování převodu pokračuje.
3. Zpráva je odebrána z fronty a umístěna do dočasné vyrovnávací paměti, která má stejnou velikost jako parametr **Buffer** . Pro operace procházení je zpráva zkopírována do dočasné vyrovnávací paměti, místo aby byla odebrána z fronty.
4. Pokud má být zpráva zkrácena tak, aby se vešla do vyrovnávací paměti, provede se následující:
   - Pokud nebyla zadána volba MQGMO_ACCEPT_TRUNCATED_MSG, bude vrácena nepřevedená zpráva s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_FAILED.
   - Pokud byla volba MQGMO_ACCEPT_TRUNCATED_MSG *zadána* , je kód dokončení nastaven na MQCC_WARNING, kód příčiny je nastaven na MQRC_TRUNCATED_MSG_ACCEPTED a zpracování převodu pokračuje.
5. Pokud lze zprávu umístit do vyrovnávací paměti bez oříznutí nebo byla zadána volba MQGMO_ACCEPT_TRUNCATED_MSG, provede se následující:

- Pokud je formát vestavěným formátem, je vyrovnávací paměť předána službě pro převod dat správce front.
- Není-li formát vestavěným formátem, je vyrovnávací paměť předána uživatelské proceduře se stejným názvem jako formát. Pokud uživatelskou proceduru nelze nalézt, bude vrácena nepřevedená zpráva s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_FORMAT_ERROR.

Pokud nedojde k žádné chybě, bude výstupem ze služby pro převod dat nebo z uživatelské procedury převedená zpráva plus kód dokončení a kód příčiny, který má být vrácen aplikaci vydávající volání MQGET.

6. Je-li převod úspěšný, vrátí správce front převedené zprávy aplikaci. V tomto případě je kód dokončení a kód příčiny vrácený voláním MQGET jednou z následujících kombinací:

Tabulka 633. Kombinace kódu dokončení a kódu příčiny

Kód dokončení	Kód příčiny
MQCC_OK	MQRC_NONE
MQCC_VAROVÁNÍ	MQRC_TRUNCATED_MSG_ACCEPTED

Pokud je však konverze provedena uživatelskou procedurou, mohou být vráceny i jiné kódy příčiny, i když je konverze úspěšná.

Pokud se převod nezdaří, správce front vrátí nepřevedenou zprávu aplikaci s poli *CodedCharSetId* a *Encoding* v parametru **MsgDesc** nastavenými na hodnoty v řídicích informacích ve zprávě a s kódem dokončení MQCC_WARNING.

Konvence zpracování

Při převodu vestavěného formátu se správce front řídí popsanými konvencemi zpracování.

Uživatelské procedury napsané uživatelem by se měly řídit také těmito konvencemi, i když to správce front nevynucuje. Vestavěné formáty převedené správcem front jsou:

- MQFMT_ADMIN
- MQFMT_CICS (pouze z/OS)
- MQFMT_COMMAND_1
- MQFMT_COMMAND_2
- MQFMT_DEAD_LETTER_HEADER
- MQFMT_DIST_HEADER
- MQFMT_EVENT verze 1
- MQFMT_EVENT verze 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 (pouze z/OS)
- MQFMT_XMIT_Q_HEADER

1. Pokud se zpráva během převodu rozbalí a překročí velikost parametru **Buffer** , provede se následující:
   - Pokud nebyla zadána volba MQGMO_ACCEPT_TRUNCATED_MSG, bude vrácena nepřevedená zpráva s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_CONVERTED_MSG_TOO_BIG.
   - Pokud byla volba MQGMO_ACCEPT_TRUNCATED_MSG *zadána* , zpráva je oříznuta, kód dokončení je nastaven na MQCC_WARNING, kód příčiny je nastaven na MQRC_TRUNCATED_MSG_ACCEPTED a zpracování převodu pokračuje.
2. Pokud dojde k oříznutí (buď před, nebo během konverze), počet platných bajtů vrácených v parametru **Buffer** může být menší než délka vyrovnávací paměti.

   K tomu může dojít například v případě, že konec vyrovnávací paměti je zakončen čtyřbajtovým celým číslem nebo znakem DBCS. Neúplný prvek informací není převeden a tyto bajty ve vrácené zprávě neobsahují platné informace. K tomu může dojít i v případě, že se během převodu zmenší zpráva, která byla oříznuta před převodem.

   Pokud je počet vrácených platných bajtů menší než délka vyrovnávací paměti, jsou nepoužívané bajty na konci vyrovnávací paměti nastaveny na hodnotu null.
3. Pokud pole nebo řetězec rozkročuje konec vyrovnávací paměti, převede se co největší část dat; nepřevede se pouze určitý prvek pole nebo znak DBCS, který je neúplný; převedou se předcházející prvky pole nebo znaky.
4. Pokud dojde k oříznutí (buď před, nebo během konverze), délka vrácená pro parametr **DataLength** je délka nepřevedené zprávy před oříznutím.
5. Když jsou řetězce převáděny mezi jednobajtovými znakovými sadami (SBCS), dvoubajtovými znakovými sadami (DBCS) nebo vícebajtovými znakovými sadami (MBCS), mohou se řetězce rozšiřovat nebo uzavírat.
   - Ve formátech PCF MQFMT_ADMIN, MQFMT_EVENT a MQFMT_PCF se řetězce ve strukturách MQCFST a MQCFSL rozšiřují nebo uzavíhají podle potřeby, aby se přizpůsobily řetězci po převodu.

     Pro strukturu seznamu řetězců MQCFSL se mohou řetězce v seznamu rozšiřovat nebo uzavírat v různých množstvích. Pokud k tomu dojde, správce front vyplní kratší řetězce mezerami, aby byly stejné délky jako nejdelší řetězec po převodu.
   - Ve formátu MQFMT_REF_MSG_HEADER se řetězce adresované poli `SrcEnvOffset`, `SrcNameOffset`, `DestEnvOffset`a `DestNameOffset` rozšiřují nebo uzavíhají podle potřeby pro umístění řetězců po převodu.
   - Ve formátu MQFMT_RF_HEADER se pole `NameValueString` rozbalí nebo podle potřeby uzavírá, aby se po převodu vešel do dvojic název-hodnota.
   - Ve strukturách s pevnými velikostmi polí umožňuje správce front řetězcům rozšiřovat nebo uzavírat smlouvy v rámci svých pevných polí za předpokladu, že nebudou ztraceny žádné významné informace. V tomto ohledu jsou koncové mezery a znaky následující za prvním znakem null v poli považovány za nevýznamné.
     - Pokud se řetězec rozbalí, ale je třeba vyřadit pouze nevýznamné znaky, aby se v poli vešel převedený řetězec, převod se zdaří a volání se dokončí s MQCC_OK a kódem příčiny MQRC_NONE (nepředpokládá se žádné další chyby).
     - Pokud se řetězec rozbalí, ale převedený řetězec vyžaduje vyřazení významných znaků, aby se vešel do pole, zpráva se vrátí nepřevedená a volání se dokončí s MQCC_WARNING a kódem příčiny MQRC_CONVERTED_STRING_TOO_BIG.

       Poznámka: Kód příčiny MQRC_CONVERTED_STRING_TOO_BIG vyústí v tomto případě bez ohledu na to, zda byla zadána volba MQGMO_ACCEPT_TRUNCATED_MSG.
     - Pokud se řetězec uzavírá, správce front vyplní řetězec mezerami na délku pole.
6. U zpráv sestávajících z jedné nebo více struktur záhlaví MQ následovaných uživatelskými daty může být převedena jedna nebo více struktur záhlaví, zatímco zbytek zprávy nikoli. Avšak (se dvěma výjimkami) pole *CodedCharSetId* a *Encoding* v každé struktuře záhlaví vždy správně označují znakovou sadu a kódování dat, která následují za strukturou záhlaví.

Dvě výjimky jsou struktury MQCIH a MQIIH, kde hodnoty v polích *CodedCharSetId* a *Encoding* v těchto strukturách nejsou významné. Pro tyto struktury jsou data následující za strukturou ve stejné znakové sadě a kódování jako samotná struktura MQCIH nebo MQIIH.

7. Pokud pole *CodedCharSetId* nebo *Encoding* v řídicích informacích načítané zprávy nebo v parametru **MsgDesc** uvádějí hodnoty, které nejsou definovány nebo nejsou podporovány, může správce front ignorovat chybu, pokud nedefinovaná nebo nepodporovaná hodnota nemusí být použita při převodu zprávy.

   Pokud například pole *Encoding* ve zprávě uvádí nepodporované kódování typu float, ale zpráva obsahuje pouze celočíselná data, nebo obsahuje data s pohyblivou řádovou čárkou, která nevyžadují převod (protože zdrojové a cílové kódování typu float jsou identické), chyba nemusí být diagnostikována.

   Je-li chyba diagnostikována, je vrácena nepřevedená zpráva s kódem dokončení MQCC_WARNING a jedním z kódů příčiny MQRC_SOURCE_ * _ERROR nebo MQRC_TARGET_ * _ERROR (podle potřeby); pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** jsou nastavena na hodnoty v řídicích informacích ve zprávě.

   Není-li chyba diagnostikována a převod byl úspěšně dokončen, hodnoty vrácené v polích *CodedCharSetId* a *Encoding* v parametru **MsgDesc** jsou hodnoty určené aplikací vydávající volání MQGET.

8. Ve všech případech, pokud je zpráva vrácena nepřevedené aplikaci, je kód dokončení nastaven na MQCC_WARNING a pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** jsou nastaveny na hodnoty odpovídající nepřevedeným datům. To se provádí také pro MQFMT_NONE.

   Parametr **Reason** je nastaven na kód, který označuje, proč nebylo možné převod provést, pokud zpráva nebyla také oseknuta; kódy příčiny související s oseknutím mají přednost před kódy příčiny souvisejícími s převodem. (Chcete-li určit, zda byla oříznutá zpráva převedena, zkontrolujte hodnoty vrácené v polích *CodedCharSetId* a *Encoding* v parametru **MsgDesc** .)

   Při diagnostikování chyby je vrácen specifický kód příčiny nebo obecný kód příčiny MQRC_NOT_CONVERTED. Vrácený kód příčiny závisí na diagnostických schopnostech základní služby pro převod dat.

9. Je-li vrácen kód dokončení MQCC_WARNING a je-li relevantní více než jeden kód příčiny, pořadí priorit je následující:

   a. Následující důvody mají přednost před všemi ostatními; může vzniknout pouze jeden z důvodů v této skupině:

      - MQRC_SIGNAL_REQUEST_ACCEPTED
      - MQRC_TRUNCATED_MSG_ACCEPTED

   b. Pořadí priorit ve zbývajících kódech příčiny není definováno.

10. Po dokončení volání MQGET:

    - Následující kód příčiny označuje, že zpráva byla úspěšně převedena:
      - MQRC_NONE
    - Následující kódy příčiny označují, že zpráva *mohla* být úspěšně převedena (zkontrolujte pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** , abyste zjistili):
      - MQRC_MSG_MARKED_BROWSE_CO_OP
      - MQRC_TRUNCATED_MSG_ACCEPTED
    - Všechny ostatní kódy příčiny označují, že zpráva nebyla převedena.

    Následující zpracování je specifické pro vestavěné formáty; nevztahuje se na formáty definované uživatelem:

11. S výjimkou následujících formátů:

    - MQFMT_ADMIN
    - MQFMT_COMMAND_1

- MQFMT_COMMAND_2
- MQFMT_EVENT
- MQFMT_IMS_VAR_STRING
- MQFMT_PCF
- MQFMT_STRING

žádný z vestavěných formátů nelze převést ze znakových sad nebo do znakových sad, které nemají znaky SBCS pro znaky platné v názvech front. Dojde-li k pokusu o provedení takového převodu, vrátí se zpráva nepřevedená s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_SOURCE_CCSID_ERROR nebo MQRC_TARGET_CCSID_ERROR podle potřeby.

Znaková sada Unicode UTF-16 je příkladem znakové sady, která neobsahuje znaky SBCS pro znaky platné v názvech front.

12. Pokud jsou data zprávy pro vestavěný formát oseknuta, pole ve zprávě, která obsahují délku řetězců nebo počet prvků nebo struktur, nejsou upravena tak, aby odrážela délku dat skutečně vrácených aplikaci; hodnoty vrácené pro tato pole v datech zprávy jsou hodnoty použitelné pro zprávu *před oseknutím*.

    Při zpracování zpráv, jako je například zkrácená zpráva MQFMT_ADMIN, se ujistěte, že se aplikace nepokouší o přístup k datům za koncem vrácených dat.

13. Je-li název formátu MQFMT_DEAD_LETTER_HEADER, data zprávy začínají strukturou MQDLH, za kterou může následovat nula nebo více bajtů dat zprávy aplikace. Formát, znaková sada a kódování dat zprávy aplikace jsou definovány poli `Format`, `CodedCharSetId`a `Encoding` ve struktuře MQDLH na začátku zprávy. Vzhledem k tomu, že struktura MQDLH a data zpráv aplikace mohou mít různé znakové sady a kódování, může jedna, druhá nebo obě struktury MQDLH a data zpráv aplikace vyžadovat převod.

    Správce front nejprve převede strukturu MQDLH podle potřeby. Pokud je převod úspěšný nebo struktura MQDLH převod nevyžaduje, správce front zkontroluje pole `CodedCharSetId` a `Encoding` ve struktuře MQDLH, aby zjistil, zda je vyžadován převod dat zprávy aplikace. Je-li převod vyžadován, správce front vyvolá uživatelskou proceduru s názvem zadaným v poli `Format` ve struktuře MQDLH nebo provede převod sám (pokud `Format` je název vestavěného formátu).

    Pokud volání MQGET vrátí kód dokončení MQCC_WARNING a kód příčiny je jedním z těch, které indikují, že převod nebyl úspěšný, platí jedna z následujících možností:

    - Strukturu MQDLH nelze převést. V tomto případě nebudou převedena ani data zprávy aplikace.
    - Struktura MQDLH byla převedena, ale data zprávy aplikace nikoli.

    Aplikace může zkontrolovat hodnoty vrácené v polích `CodedCharSetId` a `Encoding` v parametru **MsgDesc** a hodnoty ve struktuře MQDLH, aby určila, která z výše uvedených hodnot se použije.

14. Je-li název formátu MQFMT_XMIT_Q_HEADER, data zprávy začínají strukturou MQXQH, za kterou může následovat nula nebo více bajtů dalších dat. Tato další data jsou obvykle data zprávy aplikace (která mohou mít nulovou délku), ale na začátku dalších dat může být přítomna také jedna nebo více dalších struktur záhlaví produktu MQ .

    Struktura MQXQH musí být ve znakové sadě a kódování správce front. Formát, znaková sada a kódování dat následujících po struktuře MQXQH jsou dány poli `Format`, `CodedCharSetId`a `Encoding` ve struktuře MQMD obsažené v rámci MQXQH. Pro každou další přítomnou strukturu záhlaví MQ pole `Format`, `CodedCharSetId`a `Encoding` ve struktuře popisují data, která následují za touto strukturou; tato data jsou buď jinou strukturou záhlaví MQ , nebo daty zprávy aplikace.

    Je-li pro zprávu MQFMT_XMIT_Q_HEADER zadána volba MQGMO_CONVERT, dojde k převodu dat zprávy aplikace a určitých struktur záhlaví MQ *, ale data ve struktuře MQXQH nejsou*. Při návratu z volání MQGET tedy:

    - Hodnoty polí `Format`, `CodedCharSetId`a `Encoding` v parametru **MsgDesc** popisují data ve struktuře MQXQH, nikoli data zprávy aplikace. Tyto hodnoty se tedy liší od hodnot určených aplikací, které vyvolaly volání MQGET.

Výsledkem je, že aplikace, která opakovaně získává zprávy z přenosové fronty se zadanou volbou MQGMO_CONVERT, musí před každým voláním MQGET resetovat pole `CodedCharSetId` a `Encoding` v parametru **MsgDesc** na hodnoty požadované pro data zprávy aplikace.

- Hodnoty polí `Format`, `CodedCharSetId`a `Encoding` v poslední struktuře záhlaví MQ popisují data zprávy aplikace. Pokud nejsou k dispozici žádné jiné struktury záhlaví produktu MQ , jsou data zprávy aplikace popsána těmito poli ve struktuře MQMD v rámci struktury MQXQH. Je-li převod úspěšný, hodnoty budou stejné jako hodnoty zadané v parametru **MsgDesc** aplikací, která vydala volání MQGET.

Pokud se jedná o zprávu distribučního seznamu, je struktura MQXQH následována strukturou MQDH (plus její pole záznamů MQOR a MQPMR), za kterou může následovat nula nebo více dalších struktur záhlaví MQ a nula nebo více bajtů dat zprávy aplikace. Podobně jako struktura MQXQH musí být struktura MQDH ve znakové sadě a kódování správce front a není převedena ve volání MQGET, i když je zadána volba MQGMO_CONVERT.

Zpracování dříve popsaných struktur MQXQH a MQDH je primárně určeno pro použití agenty kanálů zpráv při získání zpráv z přenosových front.

Převod zpráv sestavy

Obecně může zpráva sestavy obsahovat různé množství dat zprávy aplikace podle voleb sestavy určených odesilatelem původní zprávy. Avšak sestava aktivity může obsahovat data, ale bez volby sestavy uvádějící * _WITH_DATA v konstantě.

Zpráva sestavy může obsahovat zejména:

1. Žádná data zprávy aplikace
2. Některá data zprávy aplikace z původní zprávy

   K tomu dochází, když odesilatel původní zprávy uvádí MQRO_ * _WITH_DATA a zpráva je delší než 100 bajtů.
3. Všechna data zprávy aplikace z původní zprávy

   K tomu dochází, když odesílatel původní zprávy uvádí MQRO_ * _WITH_FULL_DATA nebo uvádí MQRO_ * _WITH_DATA a zpráva je 100 bajtů nebo kratší.

Když správce front nebo agent kanálu zpráv vygeneruje zprávu sestavy, zkopíruje název formátu z původní zprávy do pole *`Format`* v řídicích informacích ve zprávě sestavy. Název formátu ve zprávě sestavy proto může znamenat délku dat, která se liší od délky skutečně přítomné ve zprávě sestavy (případy 1 a 2 dříve).

Je-li při načítání zprávy sestavy zadána volba MQGMO_CONVERT:

- V předchozím případě 1 není uživatelská procedura převodu dat vyvolána (protože zpráva sestavy neobsahuje žádná data).
- V případě 3 dříve název formátu správně implikuje délku dat zprávy.
- Pro případ 2 je však dříve vyvolána uživatelská procedura převodu dat za účelem převodu zprávy, která je *kratší* než délka odvozená z názvu formátu.

  Kromě toho je kód příčiny předaný uživatelské proceduře obvykle MQRC_NONE (to znamená, že kód příčiny neoznačuje, že zpráva byla zkrácena). K tomu dochází proto, že data zprávy byla zkrácena *odesilatelem* zprávy sestavy, a nikoli správcem front příjemce v reakci na volání MQGET.

Vzhledem k těmto možnostem nesmí uživatelská procedura převodu dat použít název formátu k odpočtu délky dat, která jí byla předána; místo toho musí uživatelská procedura zkontrolovat délku poskytnutých dat a musí být připravena převést méně dat, než je délka odvozená z názvu formátu. Pokud lze data úspěšně převést, uživatelská procedura musí vrátit kód dokončení MQCC_OK a kód příčiny MQRC_NONE. Délka dat zprávy, která se mají převést, je předána uživatelské proceduře jako parametr **InBufferLength** .

Programovací rozhraní citlivé na produkt

MQDXP-Parametr uživatelské procedury pro převod dat

Struktura MQDXP je parametr, který správce front předá uživatelské proceduře pro převod dat při vyvolání uživatelské procedury pro převod dat v rámci zpracování volání MQGET. Podrobnosti o uživatelské proceduře pro převod dat naleznete v popisu volání MQ_DATA_CONV_EXIT.

Znaková data v MQDXP jsou ve znakové sadě lokálního správce front. Tato sada je určena atributem správce front **CodedCharSetId** . Číselná data v MQDXP jsou v nativním kódování počítače; toto je dáno MQENC_NATIVE.

Uživatelskou procedurou lze změnit pouze pole *DataLength*, *CompCode*, *Reason*a *ExitResponse* v MQDXP; změny ostatních polí jsou ignorovány. Pole *DataLength* však nelze změnit, pokud převáděná zpráva je segmentem, který obsahuje pouze část logické zprávy.

Když se řízení vrátí do správce front z uživatelské procedury, zkontroluje správce front hodnoty vrácené v MQDXP. Pokud vrácené hodnoty nejsou platné, správce front pokračuje ve zpracování, jako by uživatelská procedura vrátila příkaz MQXDR_CONVERSION_FAILED in *ExitResponse* ; Správce front však v tomto případě ignoruje hodnoty polí *CompCode* a *Reason* vrácené uživatelskou procedurou a místo toho použije hodnoty, které tato pole měla na *vstupu* do uživatelské procedury. Následující hodnoty v MQDXP způsobují toto zpracování:

- Pole *ExitResponse* není MQXDR_OK a není MQXDR_CONVERSION_FAILED
- Pole *CompCode* není MQCC_OK a není MQCC_WARNING
- Pole *DataLength* menší než nula nebo *DataLength* se změnilo, když je převáděná zpráva segmentem, který obsahuje pouze část logické zprávy.

Následující tabulka shrnuje pole ve struktuře.

Tabulka 634. Pole v MQDXP

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
AppOptions	Volby aplikace	AppOptions
Encoding	Číselné kódování požadované aplikací	Kódování
CodedCharSetId	Znaková sada vyžadovaná aplikací	CodedCharSetId
DataLength	Délka dat zprávy v bajtech	DataLength
CompCode	Kód dokončení	CompCode
Reason	Kvalifikující se kód příčiny *CompCode*	Příčina
ExitResponse	Odezva z ukončení	ExitResponse
Hconn	Manipulátor připojení	Připojení
pEntryPoints	Adresa struktury MQIEP	pEntryBody

Pole

Struktura MQDXP obsahuje následující pole; pole jsou popsána v abecedním pořadí.

AppOptions
Typ: MQLONG

Jedná se o kopii pole *Options* struktury MQGMO určené aplikací vydávající volání MQGET. Uživatelská procedura možná bude muset zkontrolovat, zda byla zadána volba MQGMO_ACCEPT_TRUNCATED_MSG.

Toto je vstupní pole pro ukončení.

CodedCharSetId
Typ: MQLONG

Jedná se o identifikátor kódované znakové sady znakové sady požadované aplikací vydávající volání MQGET. Další podrobnosti naleznete v poli *CodedCharSetId* ve struktuře MQMD. Pokud aplikace určuje speciální hodnotu MQCCSI_Q_MGR pro volání MQGET, správce front ji před vyvoláním uživatelské procedury změní na skutečný identifikátor znakové sady znakové sady používané správcem front.

Je-li převod úspěšný, uživatelská procedura musí toto zkopírovat do pole *CodedCharSetId* v deskriptoru zprávy.

Toto je vstupní pole pro ukončení.

CompCode
Typ: MQLONG

Je-li uživatelská procedura vyvolána, obsahuje kód dokončení vrácený aplikaci, která vydala volání MQGET, pokud uživatelská procedura nic neprovede. Vždy se jedná o MQCC_WARNING, protože buď byla zpráva oříznuta, nebo zpráva vyžaduje převod, a to ještě nebylo provedeno.

Ve výstupu uživatelské procedury toto pole obsahuje kód dokončení, který má být vrácen aplikaci v parametru **CompCode** volání MQGET; platné jsou pouze hodnoty MQCC_OK a MQCC_WARNING. Chcete-li získat návrhy, jak může uživatelská procedura nastavit toto pole na výstupu, prohlédněte si popis pole *Reason* .

Toto je vstupní/výstupní pole pro ukončení.

DataLength
Typ: MQLONG

Při vyvolání uživatelské procedury toto pole obsahuje původní délku dat zprávy aplikace. Pokud byla zpráva oříznuta tak, aby se vešla do vyrovnávací paměti poskytované aplikací, velikost zprávy poskytnuté uživatelské proceduře je *menší* než hodnota *DataLength*. Velikost zprávy poskytnuté uživatelské proceduře je vždy dána parametrem **InBufferLength** uživatelské procedury, bez ohledu na jakékoli oříznutí, které se vyskytlo.

Oříznutí je označeno polem *Reason* , které má hodnotu MQRC_TRUNCATED_MSG_ACCEPTED na vstupu do uživatelské procedury.

Většina převodů nemusí tuto délku měnit, ale uživatelská procedura tak může učinit v případě potřeby; hodnota nastavená uživatelskou procedurou je vrácena aplikaci v parametru **DataLength** volání MQGET. Tuto délku však nelze změnit, pokud převáděná zpráva je segmentem, který obsahuje pouze část logické zprávy. Důvodem je, že změna délky by způsobila, že by posuny pozdějších segmentů v logické zprávě byly nesprávné.

Všimněte si, že pokud uživatelská procedura chce změnit délku dat, mějte na paměti, že správce front již rozhodl, zda se data zprávy vejdou do vyrovnávací paměti aplikace, a to na základě délky *nepřevedených* dat. Toto rozhodnutí určuje, zda je zpráva odebrána z fronty (nebo zda je přesunut kurzor procházení pro požadavek na procházení) a není ovlivněna žádnou změnou délky dat způsobenou konverzí. Z tohoto důvodu se doporučuje, aby uživatelské procedury převodu nezpůsobily změnu délky dat zprávy aplikace.

Pokud převod znaků znamená změnu délky, lze řetězec převést na jiný řetězec se stejnou délkou v bajtech, oseknout koncové mezery nebo podle potřeby vyplňovat mezerami.

Uživatelská procedura není vyvolána, pokud zpráva neobsahuje žádná data zprávy aplikace; proto je hodnota *DataLength* vždy větší než nula.

Toto je vstupní/výstupní pole pro ukončení.

Encoding
Typ: MQLONG

Číselné kódování požadované aplikací.

Jedná se o číselné kódování vyžadované aplikací vydávající volání MQGET. Další podrobnosti naleznete v poli *Encoding* ve struktuře MQMD.

Je-li převod úspěšný, uživatelská procedura jej zkopíruje do pole *Encoding* v deskriptoru zprávy.

Toto je vstupní pole pro ukončení.

ExitOptions
Typ: MQLONG

Toto je vyhrazené pole; jeho hodnota je `0`.

ExitResponse
Typ: MQLONG

Odezva z ukončení.Toto je nastaveno uživatelskou procedurou, která označuje úspěch nebo neúspěch převodu. Musí to být jeden z následujících:

MQXDR_OK
Převod byl úspěšný.

Pokud uživatelská procedura určuje tuto hodnotu, vrátí aplikaci, která vydala volání MQGET, následující příkaz:

- Hodnota pole *CompCode* na výstupu z ukončení
- Hodnota pole *Reason* na výstupu z ukončení
- Hodnota pole *DataLength* na výstupu z ukončení
- Obsah výstupní vyrovnávací paměti uživatelské procedury *OutBuffer*. Počet vrácených bajtů je menší z parametru **OutBufferLength** uživatelské procedury a hodnota pole *DataLength* na výstupu z uživatelské procedury.

  Pokud jsou pole *Encoding* a *CodedCharSetId* v parametru deskriptoru zprávy uživatelské procedury *obojí* beze změny, správce front vrátí:
- Hodnota polí *Encoding* a *CodedCharSetId* ve struktuře MQDXP na *vstupu* do uživatelské procedury.

  Pokud došlo ke změně jednoho nebo obou polí *Encoding* a *CodedCharSetId* v parametru deskriptoru zprávy uživatelské procedury, vrátí správce front:
- Hodnota polí *Encoding* a *CodedCharSetId* v parametru deskriptoru zprávy uživatelské procedury na výstupu uživatelské procedury

MQXDR_CONVERSION_FAILED
Převod byl neúspěšný.

Pokud uživatelská procedura určuje tuto hodnotu, vrátí aplikaci, která vydala volání MQGET, následující příkaz:

- Hodnota pole *CompCode* na výstupu z ukončení
- Hodnota pole *Reason* na výstupu z ukončení
- Hodnota pole *DataLength* na *vstupu* do uživatelské procedury
- Obsah vstupní vyrovnávací paměti uživatelské procedury *InBuffer*. Počet vrácených bajtů je dán parametrem **InBufferLength** .

Pokud uživatelská procedura změnila hodnotu *InBuffer*, výsledky nejsou definovány.

ExitResponse je výstupní pole z ukončení.

Hconn
Typ: MQHCONN

Jedná se o manipulátor připojení, který lze použít pro volání MQXCNVC. Tento manipulátor nemusí být nutně stejný jako manipulátor určený aplikací, která zadala volání MQGET.

pEntryPoints
Typ: PMQIEP

Adresa struktury MQIEP, jejímž prostřednictvím lze provádět volání MQI a DCI.

Reason
Typ: MQLONG

Kód příčiny kvalifikující *CompCode*.

Při vyvolání uživatelské procedury obsahuje kód příčiny vrácený aplikaci, která vydala volání MQGET, pokud se uživatelská procedura rozhodne nic nedělat. Mezi možné hodnoty patří MQRC_TRUNCATED_MSG_ACCEPTED, což znamená, že zpráva byla oříznuta, aby se vešla do vyrovnávací paměti poskytované aplikací, a MQRC_NOT_CONVERTED, což znamená, že zpráva vyžaduje převod, ale ještě nebyla provedena.

Ve výstupu uživatelské procedury toto pole obsahuje důvod, proč se má vrátit aplikaci v parametru **Reason** volání MQGET; doporučuje se následující:

- Má-li *Reason* hodnotu MQRC_TRUNCATED_MSG_ACCEPTED na vstupu do uživatelské procedury, pole *Reason* a *CompCode* nesmí být změněna, bez ohledu na to, zda je převod úspěšný nebo se nezdařil.

  (Pokud pole *CompCode* není MQCC_OK, aplikace, která načte zprávu, může identifikovat selhání převodu porovnáním vrácených hodnot *Encoding* a *CodedCharSetId* v deskriptoru zprávy s požadovanými hodnotami; aplikace naopak nemůže odlišit oseknutou zprávu od zprávy, která vyrovnávací paměť obsahuje. Z tohoto důvodu musí být příkaz MQRC_TRUNCATED_MSG_ACCEPTED vrácen přednostně před kteroukoli z příčin, které indikují selhání převodu.)
- Pokud má parametr *Reason* na vstupu do uživatelské procedury jinou hodnotu:
  - Pokud je převod úspěšný, *CompCode* musí být nastaven na MQCC_OK a *Reason* na MQRC_NONE.
  - Pokud se převod nezdaří nebo se zpráva rozbalí a musí být oříznuta, aby se vešla do vyrovnávací paměti, *CompCode* musí být nastaveno na hodnotu MQCC_WARNING (nebo ponecháno beze změny) a *Reason* musí být nastavena na jednu z uvedených hodnot, aby se označila povaha selhání.

    Všimněte si, že pokud je zpráva po převodu pro vyrovnávací paměť příliš velká, musí být oříznuta pouze v případě, že aplikace, která vydala volání MQGET, určila volbu MQGMO_ACCEPT_TRUNCATED_MSG:

    - Pokud tato volba byla zadána, bude vrácena příčina MQRC_TRUNCATED_MSG_ACCEPTED.
    - Pokud tato volba nebyla uvedena, zpráva se vrátí nepřevedená s kódem příčiny MQRC_CONVERTED_MSG_TOO_BIG.

Vypsané kódy příčiny jsou doporučeny pro použití uživatelskou procedurou k označení příčiny selhání konverze, ale uživatelská procedura může vrátit jiné hodnoty ze sady kódů MQRC_ *, pokud to považuje za vhodné. Kromě toho je rozsah hodnot MQRC_APPL_FIRST až MQRC_APPL_LAST alokován pro použití uživatelskou procedurou k označení podmínek, za kterých chce uživatelská procedura komunikovat s aplikací vydávající volání MQGET.

Poznámka: Pokud zprávu nelze úspěšně převést, musí uživatelská procedura vrátit hodnotu MQXDR_CONVERSION_FAILED v poli *ExitResponse* , aby správce front vrátil nepřevedenou zprávu. To platí bez ohledu na kód příčiny vrácený v poli *Reason* .

MQRC_APPL_FIRST
(900, X'384 ') Nejnižší hodnota pro kód příčiny definovaný aplikací.

MQRC_APPL_LAST
(999, X'3E7') Nejvyšší hodnota pro kód příčiny definovaný aplikací.

MQRC_CONVERTED_MSG_TOO_BIG
(2120, X'848 ') Převedená data jsou příliš velká pro vyrovnávací paměť.

MQRC_NOT_CONVERTED
(2119, X'847 ') Data zprávy nebyla převedena.

MQRC_SOURCE_CCSID_ERROR
(2111, X'83F') Identifikátor zdrojové kódované znakové sady není platný.

MQRC_SOURCE_DECIMAL_ENC_ERROR
(2113, X'841 ') kódování Packed-decimal ve zprávě nebylo rozpoznáno.

MQRC_SOURCE_FLOAT_ENC_ERROR
(2114, X'842 ') kódování s pohyblivou řádovou čárkou ve zprávě nebylo rozpoznáno.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840 ') kódování zdrojového celého čísla nebylo rozpoznáno.

MQRC_TARGET_CCSID_ERROR
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

MQRC_TARGET_DECIMAL_ENC_ERROR
(2117, X'845 ') kódování Packed-decimal určené příjemcem nebylo rozpoznáno.

MQRC_TARGET_FLOAT_ENC_ERROR
(2118, X'846 ') kódování s pohyblivou řádovou čárkou určené příjemcem nebylo rozpoznáno.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844 ') kódování cílového celého čísla nebylo rozpoznáno.

MQRC_TRUNCATED_MSG_ACCEPTED
(2079, X'81F') Vrácena zkrácená zpráva (zpracování dokončeno).

Toto je vstupní/výstupní pole pro ukončení.

StrucId
Typ: MQCHAR4

Identifikátor struktury.Hodnota musí být:

MQDXP_STRUC_ID
Identifikátor pro strukturu parametru uživatelské procedury konverze dat.

Pro programovací jazyk C je definována také konstanta MQDXP_STRUC_ID_ARRAY; má stejnou hodnotu jako MQDXP_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro ukončení.

Version
Typ: MQLONG

Číslo verze struktury.Hodnota musí být:

MQDXP_VERSION_1
Číslo verze pro strukturu výstupních parametrů konverze dat.

Následující konstanta určuje číslo verze aktuální verze:

MQDXP_CURRENT_VERSION
Aktuální verze struktury parametrů uživatelské procedury konverze dat.

Poznámka: Při zavedení nové verze této struktury se nezmění rozvržení stávající části. Uživatelská procedura proto musí zkontrolovat, zda je pole *Version* stejné nebo větší než nejnižší verze, která obsahuje pole, která uživatelská procedura potřebuje použít.

Toto je vstupní pole pro ukončení.

C prohlášení

```
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 */
};
```

Deklarace jazyka COBOL (pouze IBM i)

```
**   MQDXP structure
  10 MQDXP.
**    Structure identifier
   15 MQDXP-STRUCID        PIC X(4).
**    Structure version number
   15 MQDXP-VERSION        PIC S9(9) BINARY.
**    Reserved
   15 MQDXP-EXITOPTIONS    PIC S9(9) BINARY.
**    Application options
   15 MQDXP-APPOPTIONS     PIC S9(9) BINARY.
**    Numeric encoding required by application
   15 MQDXP-ENCODING       PIC S9(9) BINARY.
**    Character set required by application
   15 MQDXP-CODEDCHARSETID PIC S9(9) BINARY.
**    Length in bytes of message data
   15 MQDXP-DATALENGTH     PIC S9(9) BINARY.
**    Completion code
   15 MQDXP-COMPCODE       PIC S9(9) BINARY.
**    Reason code qualifying COMPCODE
   15 MQDXP-REASON         PIC S9(9) BINARY.
**    Response from exit
   15 MQDXP-EXITRESPONSE   PIC S9(9) BINARY.
**    Connection handle
   15 MQDXP-HCONN          PIC S9(9) BINARY.
```

System/390 deklarace modulu asembleru

```
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-Převod znaků

Volání MQXCNVC převádí znaky z jedné znakové sady do jiné pomocí programovacího jazyka C.

Toto volání je součástí rozhraní DCI (IBM MQ Data Conversion Interface), které je jedním z rozhraní rámce IBM MQ .

Poznámka: Volání lze použít jak z prostředí aplikace, tak z prostředí uživatelských procedur pro převod dat.

Syntaxe

MQXCNVC (*Hconn*, *Volby*, *SourceCCSID*, *SourceLength*, *SourceBuffer*, *TargetCCSID*, *TargetLength*, *TargetBuffer*, *DataLength*, *CompCode*, *Příčina*)

Parametry

Hconn (připojení)
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front.

V uživatelské proceduře pro převod dat je Hconn obvykle manipulátor předaný uživatelské proceduře pro převod dat v poli Hconn struktury MQDXP. Tento manipulátor nemusí být nutně stejný jako manipulátor určený aplikací, která vydala volání MQGET.

V systému IBM ilze pro Hconnzadat následující speciální hodnotu:

MQHC_DEF_HCONN
Výchozí manipulátor připojení.

Pokud spustíte aplikaci CICS TS 3.2 nebo vyšší, ujistěte se, že uživatelský program převodu znaků, který vyvolává volání MQXCNVC, je definován jako OPENAPI. Tato definice zabraňuje chybě 2018 MQRC_HCONN_ERROR způsobenou nesprávným připojením a umožňuje dokončení MQGET.

Volby
Typ: MQLONG-vstup

Volby, které řídí akci MQXCNVC.

Lze uvést nula nebo více popsaných voleb. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Výchozí-volba převodu: Následující volba řídí použití výchozího převodu znaků:

MQDCC_DEFAULT_CONVERSION
Výchozí převod.

Tato volba určuje, že lze použít výchozí znakovou konverzi, pokud není podporována jedna nebo obě znakové sady uvedené ve volání. To umožňuje správci front při převodu řetězce použít výchozí znakovou sadu určenou instalací, která se blíží určené znakové sadě.

Poznámka: Výsledkem použití přibližné znakové sady pro převod řetězce je, že některé znaky mohou být nesprávně převedeny. Tomu se lze vyhnout tím, že v řetězci budou použity pouze znaky, které jsou společné jak pro zadanou znakovou sadu, tak pro výchozí znakovou sadu.

Výchozí znakové sady jsou definovány volbou konfigurace při instalaci nebo restartování správce front.

Není-li zadána volba MQDCC_DEFAULT_CONVERSION, použije správce front k převodu řetězce pouze určené znakové sady a volání se nezdaří, není-li podporována jedna nebo obě znakové sady.

Tato volba je podporována v následujících prostředích:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

Volba vyplnění: Následující volba umožňuje správci front zaplnit převedený řetězec mezerami nebo zahodit nevýznamné koncové znaky, aby převedený řetězec odpovídal cílové vyrovnávací paměti:

MQDCC_FILL_TARGET_BUFFER
Vyplnit cílovou vyrovnávací paměť.

Tato volba vyžaduje, aby se převod uskutečnil tak, aby cílová vyrovnávací paměť byla zcela vyplněna:

- Pokud se řetězec při převodu uzavírá, přidají se koncové mezery, aby se vyplnila cílová vyrovnávací paměť.
- Pokud se řetězec při převodu rozbalí, koncové znaky, které nejsou významné, se zahodí, aby převedený řetězec odpovídal cílové vyrovnávací paměti. Pokud to lze úspěšně provést, volání se dokončí s MQCC_OK a kódem příčiny MQRC_NONE.

  Pokud existuje příliš málo nevýznamných koncových znaků, je do cílové vyrovnávací paměti vložena velká část řetězce a volání je dokončeno s MQCC_WARNING a kódem příčiny MQRC_CONVERTED_MSG_TOO_BIG.

  Nevýznamné znaky jsou:

  – Koncové mezery

  – Znaky následující za prvním znakem null v řetězci (ale bez samotného prvního znaku null)
- Pokud jsou řetězce `TargetCCSID`a `TargetLength` takové, že cílovou vyrovnávací paměť nelze zcela nastavit s platnými znaky, volání selže s MQCC_FAILED a kódem příčiny MQRC_TARGET_LENGTH_ERROR. K tomu může dojít, když je `TargetCCSID` ryzí znaková sada DBCS (například UTF-16), ale `TargetLength` uvádí délku, která je lichý počet bajtů.
- `TargetLength` může být menší nebo větší než `SourceLength`. Při návratu z MQXCNVC má `DataLength` stejnou hodnotu jako `TargetLength`.

Není-li tato volba uvedena:

- Řetězec může podle potřeby uzavřít smlouvu nebo rozšířit v cílové vyrovnávací paměti. Nevýznamné koncové znaky se nepřidávají ani neodstraňují.

  Pokud se převedený řetězec vejde do cílové vyrovnávací paměti, volání se dokončí s MQCC_OK a kódem příčiny MQRC_NONE.

  Je-li převedený řetězec pro cílovou vyrovnávací paměť příliš velký, je do cílové vyrovnávací paměti umístěna velká část řetězce a volání se dokončí s MQCC_WARNING a kódem příčiny MQRC_CONVERTED_MSG_TOO_BIG. V tomto případě lze vrátit méně než `TargetLength` bajtů.
- `TargetLength` může být menší nebo větší než `SourceLength`. Při návratu z MQXCNVC je hodnota `DataLength` menší nebo rovna hodnotě `TargetLength`.

Tato volba je podporována v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

Volby kódování: popsané volby lze použít k určení celočíselného kódování zdrojového a cílového řetězce. Příslušné kódování se použije pouze tehdy, když odpovídající identifikátor znakové sady označuje, že reprezentace znakové sady v hlavní paměti závisí na kódování použitém pro binární celá čísla. To se týká pouze určitých vícebajtových znakových sad (například znakových sad UTF-16).

Kódování je ignorováno, pokud je znaková sada jednobajtová znaková sada (SBCS) nebo vícebajtová znaková sada se znázorněním v hlavní paměti, která není závislá na kódování celých čísel.

Musí být zadána pouze jedna z hodnot MQDCC_SOURCE_ * v kombinaci s jednou z hodnot MQDCC_TARGET_ *:

MQDCC_SOURCE_ENC_NATIVE
Kódování zdroje je výchozí pro prostředí a programovací jazyk.

MQDCC_SOURCE_ENC_NORMAL
Kódování zdroje je normální.

MQDCC_SOURCE_ENC_REVERSED
Kódování zdroje je obrácené.

MQDCC_SOURCE_ENC_UNDEFINED
Kódování zdroje není definováno.

MQDCC_TARGET_ENC_NATIVE
Cílové kódování je výchozí pro prostředí a programovací jazyk.

MQDCC_TARGET_ENC_NORMAL
Cílové kódování je normální.

MQDCC_TARGET_ENC_REVERSED
Cílové kódování je obrácené.

MQDCC_TARGET_ENC_UNDEFINED
Cílové kódování není definováno.

Dříve definované hodnoty kódování lze přidat přímo do pole `Options` . Pokud je však zdrojové nebo cílové kódování získáno z pole `Encoding` v deskriptoru MQMD nebo jiné struktuře, je třeba provést následující zpracování:

1. Kódování celých čísel musí být extrahováno z pole `Encoding` odstraněním kódování typu float a packed-decimal. Podrobnosti o tomto postupu naleznete v části “Analýza kódování” na stránce 900 .
2. Kódování celých čísel, které je výsledkem kroku 1, musí být před přidáním do pole `Options` vynásobeno příslušným faktorem. Tyto faktory jsou:
   - MQDCC_SOURCE_ENC_FACTOR pro kódování zdroje
   - MQDCC_TARGET_ENC_FACTOR pro cílové kódování

Následující ukázkový kód ilustruje, jak to může být kódováno v programovacím jazyce C:

```
Options = (MsgDesc.Encoding & MQENC_INTEGER_MASK)
           * MQDCC_SOURCE_ENC_FACTOR
        + (DataConvExitParms.Encoding & MQENC_INTEGER_MASK)
           * MQDCC_TARGET_ENC_FACTOR;
```

Není-li uvedeno, volby kódování jsou standardně nedefinované (MQDCC_ * _ENC_UNDEFINED). Ve většině případů to neovlivní úspěšné dokončení volání MQXCNVC. Pokud je však odpovídající znaková sada vícebajtová znaková sada se znázorněním, které závisí na kódování (například znaková sada UTF-16), volání selže s kódem příčiny MQRC_SOURCE_INTEGER_ENC_ERROR nebo MQRC_TARGET_INTEGER_ENC_ERROR podle potřeby.

Volby kódování jsou podporovány v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows
- z/OS

Výchozí volba: Pokud není zadána žádná z dříve popsaných voleb, lze použít následující volbu:

MQDCC_NONE
Nejsou uvedeny žádné volby.

MQDCC_NONE je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby se tato volba používala s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

SourceCCSID
Typ: MQLONG-vstup

Jedná se o identifikátor kódované znakové sady vstupního řetězce v souboru `SourceBuffer`.

SourceLength
Typ: MQLONG-vstup

Jedná se o délku vstupního řetězce v bajtech v souboru `SourceBuffer` ; musí být nula nebo větší.

SourceBuffer
Typ: MQCHAR x SourceLength -vstup

Jedná se o vyrovnávací paměť obsahující řetězec, který má být převeden z jedné znakové sady na jinou.

TargetCCSID
Typ: MQLONG-vstup

Jedná se o identifikátor kódované znakové sady znakové sady, na kterou se má převést `SourceBuffer` .

TargetLength
Typ: MQLONG-vstup

Jedná se o délku výstupní vyrovnávací paměti v bajtech `TargetBuffer` ; musí být nula nebo větší. Může být menší nebo větší než `SourceLength`.

TargetBuffer
Typ: MQCHAR x TargetLength -výstup

Jedná se o řetězec poté, co byl převeden na znakovou sadu definovanou parametrem `TargetCCSID`. Převedený řetězec může být kratší nebo delší než nepřevedený řetězec. Parametr **`DataLength`** označuje počet vrácených platných bajtů.

DataLength
Typ: MQLONG-výstup

Jedná se o délku řetězce vráceného ve výstupní vyrovnávací paměti `TargetBuffer`. Převedený řetězec může být kratší nebo delší než nepřevedený řetězec.

CompCode
Typ: MQLONG-výstup

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující `CompCode`.

Pokud je `CompCode` MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr `CompCode` hodnotu MQCC_VAROVÁNÍ:

MQRC_CONVERTED_MSG_TOO_BIG
(2120, X'848 ') Převedená data jsou příliš velká pro vyrovnávací paměť.

Má-li parametr CompCode hodnotu MQCC_FAILED, postupujte takto:

MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

MQRC_DBCS_ERROR
(2150, X'866 ') Řetězec DBCS není platný.

MQRC_HCONN_ERROR
(2018, X'7E2') popisovač připojení není platný.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

MQRC_SOURCE_BUFFER_ERROR
(2145, X'861 ') Parametr zdrojové vyrovnávací paměti není platný.

MQRC_SOURCE_CCSID_ERROR
(2111, X'83F') Identifikátor zdrojové kódované znakové sady není platný.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840 ') kódování zdrojového celého čísla nebylo rozpoznáno.

MQRC_SOURCE_LENGTH_ERROR
(2143, X'85F') Parametr délky zdroje není platný.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Nedostatek dostupného úložiště.

MQRC_TARGET_BUFFER_ERROR
(2146, X'862 ') Parametr cílové vyrovnávací paměti není platný.

MQRC_TARGET_CCSID_ERROR
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844 ') kódování cílového celého čísla nebylo rozpoznáno.

MQRC_TARGET_LENGTH_ERROR
(2144, X'860 ') Parametr délky cíle není platný.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQXCNVC (Hconn, Options, SourceCCSID, SourceLength, SourceBuffer,
        TargetCCSID, TargetLength, TargetBuffer, &DataLength,
        &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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 */
```

Deklarace jazyka COBOL (pouze IBM i)

IBM i

```
     CALL 'MQXCNVC' USING HCONN, OPTIONS, SOURCECCSID, SOURCELENGTH,
                          SOURCEBUFFER, TARGETCCSID, TARGETLENGTH,
                          TARGETBUFFER, DATALENGTH, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   Connection handle
 01  HCONN         PIC S9(9) BINARY.
**   Options that control the action of MQXCNVC
 01  OPTIONS       PIC S9(9) BINARY.
**   Coded character set identifier of string before conversion
 01  SOURCECCSID   PIC S9(9) BINARY.
**   Length of string before conversion
 01  SOURCELENGTH  PIC S9(9) BINARY.
**   String to be converted
 01  SOURCEBUFFER  PIC X(n).
**   Coded character set identifier of string after conversion
 01  TARGETCCSID   PIC S9(9) BINARY.
**   Length of output buffer
 01  TARGETLENGTH  PIC S9(9) BINARY.
**   String after conversion
 01  TARGETBUFFER  PIC X(n).
**   Length of output string
 01  DATALENGTH    PIC S9(9) BINARY.
**   Completion code
 01  COMPCODE      PIC S9(9) BINARY.
**   Reason code qualifying COMPCODE
 01  REASON        PIC S9(9) BINARY.
```

S/390 deklarace modulu sestavení

```
         CALL MQXCNVC,(HCONN,OPTIONS,SOURCECCSID,SOURCELENGTH,         X
               SOURCEBUFFER,TARGETCCSID,TARGETLENGTH,TARGETBUFFER,     X
               DATALENGTH,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
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-uživatelská procedura převodu dat

Volání MQ_DATA_CONV_EXIT popisuje parametry předané uživatelské proceduře pro převod dat.

Správce front neposkytl žádný vstupní bod s názvem MQ_DATA_CONV_EXIT (viz poznámka k použití 11).

Tato definice je součástí rozhraní DCI (IBM MQ Data Conversion Interface), které je jedním z rozhraní rámce IBM MQ .

Syntaxe

MQ_DATA_CONV_EXIT (*DataConvExitParms*, *MsgDesc*, *InBufferDélka*, *InBuffer*, *OutBufferDélka*, *OutBuffer*)

Parametry

DataConvExitParms
Typ: MQDXP-vstup/výstup

Tato struktura obsahuje informace týkající se vyvolání uživatelské procedury. Uživatelská procedura nastaví informace v této struktuře, aby označila výsledek převodu. Podrobnosti o polích v této struktuře viz “MQDXP-Parametr uživatelské procedury pro převod dat” na stránce 912 .

MsgDesc
Typ: MQMD-vstup/výstup

Na vstupu do uživatelské procedury se jedná o deskriptor zprávy přidružený k datům zprávy předaným uživatelské proceduře v parametru **InBuffer** .

Poznámka: Parametr **MsgDesc** předaný uživatelské proceduře je vždy nejnovější verzí MQMD podporovanou správcem front, který uživatelskou proceduru vyvolává. Pokud je uživatelská procedura určena k tomu, aby byla přenosná mezi různými prostředími, zkontroluje pole `Version` v souboru `MsgDesc` , aby ověřila, že pole, ke kterým uživatelská procedura potřebuje přístup, jsou ve struktuře přítomna.

V následujících prostředích je uživatelské proceduře předáno rozhraní MQMD version-2 :

- AIX
- IBM i
- Linux
- Windows

Ve všech ostatních prostředích, která podporují uživatelskou proceduru převodu dat, je uživatelské proceduře předáno rozhraní MQMD version-1 .

Ve výstupu uživatelská procedura změní pole `Encoding` a `CodedCharSetId` na hodnoty požadované aplikací, pokud byl převod úspěšný; tyto změny se odrazí zpět do aplikace. Jakékoli další změny, které uživatelská procedura provede ve struktuře, se ignorují; neprojeví se zpět do aplikace.

Pokud uživatelská procedura vrátí hodnotu MQXDR_OK v poli `ExitResponse` struktury MQDXP, ale nezmění pole `Encoding` nebo `CodedCharSetId` v deskriptoru zprávy, vrátí správce front pro tato pole hodnoty, které měla odpovídající pole ve struktuře MQDXP na vstupu do uživatelské procedury.

InBufferdélka
Typ: MQLONG-vstup

Délka v bajtech `InBuffer`.

Jedná se o délku vstupní vyrovnávací paměti `InBuffera` určuje počet bajtů, které má uživatelská procedura zpracovat. `InBufferLength` je menší z hodnot délky dat zprávy před převodem a délky vyrovnávací paměti poskytované aplikací při volání MQGET.

Hodnota je vždy větší než nula.

InBuffer
Typ: MQBYTExInBufferLength -vstup

Vyrovnávací paměť obsahující nepřevedenou zprávu.

Obsahuje data zprávy před převodem. Pokud uživatelská procedura nemůže převést data, vrátí po dokončení uživatelské procedury aplikaci obsah této vyrovnávací paměti.

Poznámka: Ukončení by nemělo měnit `InBuffer` ; pokud je tento parametr změněn, výsledky nejsou definovány.

V programovacím jazyku C je tento parametr definován jako ukazatel na prázdno.

OutBuffer
Typ: MQLONG-vstup

Délka v bajtech `OutBuffer`.

Jedná se o délku výstupní vyrovnávací paměti `OutBuffer`a je shodná s délkou vyrovnávací paměti poskytované aplikací při volání MQGET.

Hodnota je vždy větší než nula.

OutBuffer
Typ: MQBYTExOutBufferLength -výstup

Vyrovnávací paměť obsahující převedené zprávy.

Ve výstupu uživatelské procedury, pokud byl převod úspěšný (jak uvádí hodnota MQXDR_OK v poli `ExitResponse` parametru **DataConvExitParms**), obsahuje `OutBuffer` data zprávy, která mají být doručena aplikaci, v požadované reprezentaci. Pokud byl převod neúspěšný, všechny změny, které uživatelská procedura provedla v této vyrovnávací paměti, jsou ignorovány.

V programovacím jazyku C je tento parametr definován jako ukazatel na prázdno.

Poznámky k použití

1. Uživatelská procedura pro převod dat je uživatelská procedura, která přijímá řízení během zpracování volání MQGET. Funkce prováděná uživatelskou procedurou pro převod dat je definována poskytovatelem uživatelské procedury; uživatelská procedura však musí být v souladu s pravidly popsanými zde a v přidružené struktuře parametrů MQDXP.

   Programovací jazyky, které lze použít pro uživatelskou proceduru převodu dat, jsou určeny prostředím.
2. Uživatelská procedura je vyvolána pouze v případě, že jsou splněny všechny následující podmínky:
   - Volba MQGMO_CONVERT je určena ve volání MQGET.
   - Pole `Format` v deskriptoru zprávy není MQFMT_NONE
   - Zpráva již není v požadované reprezentaci, tj. jedna nebo obě hodnoty `CodedCharSetId` a `Encoding` zprávy se liší od hodnoty určené aplikací v deskriptoru zprávy zadaném ve volání MQGET.
   - Správce front dosud úspěšně neprovedl převod.
   - Délka vyrovnávací paměti aplikace je větší než nula.
   - Délka dat zprávy je větší než nula.
   - Kód příčiny dosud během operace MQGET je MQRC_NONE nebo MQRC_TRUNCATED_MSG_ACCEPTED
3. Při zápisu uživatelské procedury zvažte kódování uživatelské procedury způsobem, který jí umožňuje převádět zprávy, které byly oseknuty. Zkrácené zprávy mohou vzniknout následujícími způsoby:
   - Přijímající aplikace poskytuje vyrovnávací paměť, která je menší než zpráva, ale uvádí volbu MQGMO_ACCEPT_TRUNCATED_MSG pro volání MQGET.

     V tomto případě má pole `Reason` v parametru **DataConvExitParms** na vstupu do uživatelské procedury hodnotu MQRC_TRUNCATED_MSG_ACCEPTED.
   - Odesilatel zprávy ji před odesláním ořízne. K tomu může dojít například u zpráv sestavy (další podrobnosti viz “Převod zpráv sestavy” na stránce 911).

     V tomto případě má pole `Reason` v parametru **DataConvExitParms** na vstupu do uživatelské procedury hodnotu MQRC_NONE (pokud přijímající aplikace poskytla vyrovnávací paměť, která byla dostatečně velká pro zprávu).

   Proto hodnotu pole `Reason` na vstupu do uživatelské procedury nelze vždy použít k rozhodnutí, zda byla zpráva oříznuta.

   Rozlišovací charakteristikou zkrácené zprávy je, že délka poskytnutá uživatelské proceduře v parametru **InBufferLength** je menší než délka odvozená z názvu formátu obsaženého v poli `Format` v deskriptoru zprávy. Uživatelská procedura by proto měla před pokusem o převod jakýchkoli

dat zkontrolovat hodnotu `InBufferLength` ; uživatelská procedura by neměla předpokládat, že bylo poskytnuto úplné množství dat odvozených z názvu formátu.

Pokud uživatelská procedura nebyla zapsána pro převod zkrácených zpráv a hodnota `InBufferLength` je menší než očekávaná hodnota, uživatelská procedura vrátí hodnotu MQXDR_CONVERSION_FAILED v poli `ExitResponse` parametru **DataConvExitParms** s poli `CompCode` a `Reason` nastavenými na MQCC_WARNING a MQRC_FORMAT_ERROR.

Pokud byla uživatelská procedura zapsána pro převod oseknutých zpráv, uživatelská procedura převede co nejvíce dat (viz další poznámka k použití), přičemž dbá na to, aby se nepokoušela zkoumat nebo převádět data po konci `InBuffer`. Pokud je převod úspěšně dokončen, uživatelská procedura ponechá pole `Reason` v parametru **DataConvExitParms** beze změny. Vrací MQRC_TRUNCATED_MSG_ACCEPTED, pokud byla zpráva zkrácena správcem front příjemce, a MQRC_NONE, pokud byla zpráva zkrácena odesilatelem zprávy.

Je také možné, aby se zpráva během převodu rozšířila do bodu, kde je větší než `OutBuffer`. V tomto případě musí uživatelská procedura rozhodnout, zda zprávu oříznout; pole `AppOptions` v parametru **DataConvExitParms** označuje, zda přijímající aplikace zadala volbu MQGMO_ACCEPT_TRUNCATED_MSG.

4. Obecně se převedou všechna data ve zprávě poskytnuté uživatelské proceduře v produktu `InBuffer` nebo žádná z nich. Výjimka se však vyskytne, pokud je zpráva oseknuta, buď před konverzí, nebo během konverze; v tomto případě může být na konci vyrovnávací paměti neúplná položka (například: 1 bajt dvoubajtového znaku nebo 3 bajty 4bajtového celého čísla). V této situaci zvažte vynechání neúplné položky a nastavte nepoužívané bajty v souboru `OutBuffer` na hodnoty null. Nicméně, úplné prvky nebo znaky v poli nebo řetězci by měly být převedeny.
5. Když je uživatelská procedura poprvé potřebná, pokusí se správce front načíst objekt, který má stejný název jako formát (kromě rozšíření). Načtený objekt musí obsahovat uživatelskou proceduru, která zpracovává zprávy s tímto názvem formátu. Zvažte možnost vytvoření názvu uživatelské procedury a názvu objektu, který obsahuje stejnou uživatelskou proceduru, i když to ne všechna prostředí vyžadují.
6. Nová kopie uživatelské procedury se načte, když se aplikace pokusí načíst první zprávu, která používá tento soubor `Format` od připojení aplikace ke správci front. V případě aplikací CICS nebo IMS to znamená, že se subsystém CICS nebo IMS připojí ke správci front. Novou kopii lze načíst i jindy, pokud správce front zrušil dříve načtenou kopii. Z tohoto důvodu se uživatelská procedura nesmí pokoušet o použití statického úložiště pro komunikaci informací z jednoho vyvolání uživatelské procedury do druhého-uživatelskou proceduru lze mezi dvěma vyvoláními uvolnit.
7. Pokud existuje uživatelská procedura se stejným názvem jako jeden z vestavěných formátů podporovaných správcem front, uživatelská procedura nenahradí vestavěnou převodní rutinu. Jedinými okolnostmi, za kterých je takový výstup vyvolán, jsou:
   - Pokud vestavěná převodní rutina nemůže zpracovat převody do nebo ze zapojených `CodedCharSetId` nebo `Encoding` , nebo
   - Pokud se vestavěné převodní rutině nepodařilo převést data (například proto, že existuje pole nebo znak, který nelze převést).
8. Rozsah uživatelské procedury je závislý na prostředí. Názvy `Format` musí být zvoleny, aby se minimalizovalo riziko kolizí s jinými formáty. Zvažte začátek se znaky, které identifikují aplikaci definující název formátu.
9. Uživatelská procedura převodu dat je spuštěna v prostředí, jako je prostředí programu, který vydal volání MQGET; prostředí zahrnuje adresní prostor a profil uživatele (je-li to možné). Program může být agentem kanálu zpráv odesílajícím zprávy do cílového správce front, který nepodporuje převod zpráv. Uživatelská procedura nemůže ohrozit integritu správce front, protože není spuštěna v prostředí správce front.
10. Jediným voláním MQI, které může uživatelská procedura použít, je MQXCNVC; pokus o použití jiných volání MQI selže s kódem příčiny MQRC_CALL_IN_PROGRESS nebo s jinými nepředvídatelnými chybami.
11. Správce front neposkytl žádný vstupní bod s názvem MQ_DATA_CONV_EXIT. Pro název MQ_DATA_CONV_EXIT v programovacím jazyce C je však k dispozici položka `typedef` , kterou lze

použít k deklaraci uživatelské procedury napsané uživatelem s cílem zajistit správnost parametrů. Název uživatelské procedury musí být stejný jako název formátu (název obsažený v poli `Format` v deskriptoru MQMD), ačkoli to není vyžadováno ve všech prostředích.

Následující příklad ukazuje, jak lze uživatelskou proceduru, která zpracovává formát `MYFORMAT` , deklarovat v programovacím jazyku 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. z/OS Pokud je v systému z/OStaké v platnosti uživatelská procedura rozhraní API, je volána po uživatelské proceduře pro převod dat.

Vyvolání jazyka C

```
exitname (&DataConvExitParms, &MsgDesc, InBufferLength,
        InBuffer, OutBufferLength, OutBuffer);
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
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 */
```

Deklarace jazyka COBOL (pouzeIBM i)

IBM i

```
     CALL 'exitname' USING DATACONVEXITPARMS, MSGDESC, INBUFFERLENGTH,
                           INBUFFER, OUTBUFFERLENGTH, OUTBUFFER.
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
**   Data-conversion exit parameter block
 01  DATACONVEXITPARMS.
     COPY CMQDXPV.
**   Message descriptor
 01  MSGDESC.
     COPY CMQMDV.
**   Length in bytes of INBUFFER
 01  INBUFFERLENGTH     PIC S9(9) BINARY.
**   Buffer containing the unconverted message
 01  INBUFFER           PIC X(n).
**   Length in bytes of OUTBUFFER
 01  OUTBUFFERLENGTH    PIC S9(9) BINARY.
```

```
**   Buffer containing the converted message
 01  OUTBUFFER          PIC X(n).
```

System/390 deklarace modulu asembleru

```
         CALL EXITNAME,(DATACONVEXITPARMS,MSGDESC,INBUFFERLENGTH,      X
               INBUFFER,OUTBUFFERLENGTH,OUTBUFFER)
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
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
```

Vlastnosti určené jako prvky MQRFH2

Vlastnosti deskriptoru bez zprávy lze zadat jako prvky ve složkách záhlaví MQRFH2 . Přehled prvků MQRFH2 zadaných jako vlastnosti.

Tím si zachovává kompatibilitu s předchozími verzemi klientů IBM MQ JMS a XMS . Tento oddíl popisuje, jak určit vlastnosti v záhlavích MQRFH2 .

Chcete-li použít prvky MQRFH2 jako vlastnosti, zadejte prvky podle popisu v části Použití IBM MQ classes for Java . Tyto informace doplňují informace popsané v části “MQRFH2 -Pravidla a formátování záhlaví 2” na stránce 529.

Mapování datových typů vlastností na datové typy MQRFH2

Toto téma poskytuje informace o typech vlastností zpráv mapovaných na příslušné datové typy MQRFH2 .

Tabulka 635. Podporované datové typy MQRFH2

Typ vlastnosti zprávy	**Datový typ MQRFH2**
MQBYTE []	bin.hex
MQBOOL	typ boolean
MQINT8	i1
MQINT16	i2
MQINT32	i4
MQINT64	i8
MQFLOAT32	r4
MQFLOAT64	r8
MQCHAR []	řetězec

Předpokládá se, že jakýkoli prvek bez datového typu je typu "string".

Datový typ MQRFH2 `int`, což znamená celé číslo nespecifikované velikosti, je považován za typ `i8`.

Atribut prvku `xsi:nil='true'` označuje hodnotu null. Nepoužívejte atribut `xsi:nil='false'` pro nenulové hodnoty.

Například následující vlastnost má hodnotu null:

```
<NullProperty xsi:nil='true'></NullProperty>
```

Vlastnost bajtového nebo znakového řetězce může mít prázdnou hodnotu. To představuje prvek MQRFH2 s hodnotou prvku s nulovou délkou.

Například následující vlastnost má prázdnou hodnotu:

```
<EmptyProperty></EmptyProperty>
```

Podporované složky MQRFH2

Přehled použití polí deskriptoru zpráv jako vlastností.

Složky <jms>, <mcd>, <mqext>a <usr> jsou popsány v části Záhlaví MQRFH2 a JMS. Složka <usr> se používá k přenosu vlastností definovaných aplikací JMS , které jsou přidruženy ke zprávě. Skupiny nejsou povoleny ve složce <usr> .

Záhlaví MQRFH2 a JMS podporují následující další složky:

- <mq>

  Tato složka je používána a vyhrazena pro vlastnosti definované produktem MQ, které jsou používány produktem IBM MQ.

- <mq_usr>

  Tuto složku lze použít k přenosu vlastností definovaných aplikací, které nejsou vystaveny jako vlastnosti definované uživatelem JMS , protože tyto vlastnosti nemusí splňovat požadavky vlastnosti JMS . Tato složka může obsahovat skupiny, které složka <usr> neobsahuje.

- Libovolná složka označená atributem content='properties' .

  Taková složka je ekvivalentní složce <mq_usr> v obsahu.

- <mqps>

  Tato složka se používá pro vlastnosti IBM MQ publikování/odběru.

Produkt IBM MQ také podporuje následující složky, které jsou již používány produktem WAS/SIB:

- <sib>

  Tato složka je používána a vyhrazena pro vlastnosti systémových zpráv WAS/SIB, které nejsou vystaveny jako vlastnosti JMS , nebo jsou mapovány na vlastnosti JMS_IBM_ *, ale jsou vystaveny aplikacím WAS/SIB; mezi tyto vlastnosti patří cesty dopředného a zpětného směrování.

  Alespoň některé z nich nelze vystavit jako vlastnosti JMS , protože se jedná o bajtová pole. Pokud vaše aplikace přidá vlastnosti do této složky, hodnota se buď ignoruje, nebo odebere.

- <sib_usr>

  Tato složka je používána a vyhrazena pro vlastnosti uživatelských zpráv WAS/SIB, které nelze vystavit jako uživatelské vlastnosti JMS , protože nejsou podporovaného typu; jsou vystaveny aplikacím WAS/SIB.

  Jedná se o uživatelské vlastnosti, které můžete získat nebo nastavit prostřednictvím rozhraní SIMessage, ale obsah bajtového pole je mapován na požadovanou hodnotu vlastnosti.

  Pokud vaše aplikace IBM MQ zapíše do složky libovolný prvek bin.hex , aplikace pravděpodobně obdrží prvek IOException, protože nemá očekávaný formát obnovy. Pokud přidáte něco jiného než prvek bin.hex , obdržíte ClassCastException.

  Nepokoušejte se zpřístupnit vlastnosti pro soubor WAS/SIB pomocí této složky; místo toho pro tento účel použijte složku <usr> .

- <sib_context>

Tato složka se používá pro vlastnosti systémových zpráv WAS/SIB, které nejsou vystaveny uživatelským aplikacím WAS/SIB nebo jako vlastnosti JMS . Patří mezi ně zabezpečení a transakční vlastnosti, které se používají pro webové služby a podobně.

Aplikace nesmí do této složky přidávat vlastnosti.

- `<mqema>`

  Tato složka byla použita produktem WAS/SIB namísto složky `<mqext>` .

Názvy složek MQRFH2 rozlišují velikost písmen.

Následující složky jsou vyhrazeny, v libovolné kombinaci malých a velkých písmen:

- Jakákoli složka s předponou `mq` nebo `wmq` ; vyhrazeno pro použití společností IBM MQ.
- Jakákoli složka s předponou `sib` ; vyhrazeno pro použití ve WAS/SIB.
- Složky `<Root>` a `<Body>` ; vyhrazeno, ale nepoužito.

Následující složky nejsou rozpoznány jako obsahující vlastnosti zprávy:

- `<psc>`

  Používá produkt IBM Integration Bus k předávání zpráv příkazů publikování/odběru zprostředkovateli.

- `<pscr>`

  Používá produkt IBM Integration Bus k tomu, aby obsahoval informace od zprostředkovatele v odezvě na zprávy příkazů publikování/odběru.

- Libovolná složka, která není definována pomocí IBM, není označena atributem `content='properties'` .

Neuvádějte `content='properties'` ve složkách `<psc>` nebo `<pscr>` . Pokud tak učiníte, budou tyto složky považovány za vlastnosti a produkt IBM Integration Bus pravděpodobně přestane fungovat podle očekávání.

Pokud vaše aplikace sestavuje zprávy s vlastnostmi, musí být záhlaví v záhlaví MQRFH2 rozpoznáno jako záhlaví MQRFH2 obsahující vlastnosti v seznamu záhlaví, která lze zřetězit v záhlaví zprávy.

MQRFH2 může předcházet libovolný počet standardních záhlaví MQH nebo MQCIH, MQDLH, MQIIH, MQTM, MQTMC2nebo MQXQH. Řetězec nebo MQCFH ukončí syntaktickou analýzu, protože je nelze zřetězit.

Je možné, aby zpráva obsahovala více záhlaví MQRFH2 , která obsahují všechny vlastnosti zprávy. Složky se stejným názvem mohou současně existovat v různých záhlavích, pokud není jinak omezeno, například WAS/SIB. Se složkami se zachází jako s jednou logickou složkou, pokud jsou všechny ve významných záhlavích.

Zatímco složky z důležitých záhlaví nelze sloučit s těmito složkami v nevýznamných záhlavích, složky se stejným názvem v důležitých záhlavích lze sloučit a odebrat všechny konfliktní vlastnosti. Vaše aplikace nesmí záviset na rozvržení vlastností v rámci své zprávy.

Skupiny MQRFH2 jsou analyzovány pro vlastnosti ve složkách definovaných uživatelem, tj. nikoli ve složkách `<wmq>`, `<jms>`, `<mcd>`, `<usr>`, `<mqext>`, `<sib>`, `<sib_usr>`, `<sib_context>`a `<mqema>` .

Skupiny ve složkách vlastností IBM-defined properties, s výjimkou složek `<wmq>` a `<mq>` , jsou analyzovány pro vlastnosti.

Složka MQRFH2 nesmí obsahovat smíšený obsah; složka nebo skupina mohou obsahovat buď skupiny, vlastnosti, nebo hodnotu, nikoli však obojí.

Segment zprávy, buď první, nebo následný segment, nemůže obsahovat jiné vlastnosti IBM MQ -defined, než jsou tyto vlastnosti v deskriptoru zprávy. Proto vložení zprávy obsahující tyto vlastnosti s nastavenou hodnotou MQMF_SEGMENT nebo MQMF_SEGMENTATION_ALLOWED způsobí selhání operace put s hodnotou MQRC_SEGMENTATION_NOT_ALLOWED.

Skupiny zpráv však mohou obsahovat IBM MQ -defined properties.

Generování záhlaví MQRFH2

Pokud produkt IBM MQ převede vlastnosti zprávy na jejich reprezentaci MQRFH2 , musí ke zprávě přidat MQRFH2 . Přidá MQRFH2 buď jako samostatné záhlaví, nebo jej sloučí s existujícím záhlavím.

Generování nových záhlaví MQRFH2 pomocí IBM MQ může narušit existující záhlaví ve zprávě. Aplikace, které analyzují vyrovnávací paměť zpráv pro záhlaví, si musí uvědomit, že počet a umístění záhlaví ve vyrovnávací paměti se za určitých okolností může změnit. Produkt IBM MQ se pokusí minimalizovat dopad přidání vlastností do zprávy sloučením vlastností zprávy do existujícího záhlaví MQRFH2 , kde je to možné. Pokouší se také minimalizovat dopad vložením generovaného souboru MQRFH2 do pevné pozice vzhledem k ostatním záhlavím ve vyrovnávací paměti zpráv.

Generované záhlaví MQRFH2 je umístěno za záhlaví MQMDa libovolný počet záhlaví MQXQH, MQRFHa MQDLH , bez ohledu na jejich pořadí. Generované záhlaví MQRFH2 je umístěno bezprostředně před první záhlaví, které není MQMD, MQXQH, MQDLHnebo MQRFH .

z/OS Na systémech z/OS se generované záhlaví MQRFH2 vytvoří v souboru CCSIDaplikace. To je definováno takto:

- Pro dávkové aplikace LE používající rozhraní DLL je CCSID hodnota CODESET přidružená k aktuálnímu národnímu prostředí v době vydání **MQCONN** (výchozí hodnota je 1047).
- Pro dávkové aplikace LE svázané s jedním ze stubů MQ je CCSID CODESET přidružený k aktuálnímu národnímu prostředí v době prvního volání MQI vydaného po **MQCONN** (výchozí hodnota je 1047).
- V případě dávkových aplikací, které nejsou typu LE a jsou spuštěny v podprocesu z/OS UNIX System Services (z/OS UNIX), je CCSID hodnota THLICCSID v době prvního volání MQI vydaného po **MQCONN** (výchozí hodnota je 1047).
- V případě jiných dávkových aplikací je CCSID CCSID správce front.

Pro aplikace LE lze národní prostředí změnit pomocí volatelné služby setlocale() / CEESETL LE . U aplikací, které nejsou typu LE a běží na podprocesech z/OS UNIX , lze hodnotu proměnné THLICCSID změnit pomocí z/OS UNIX mapovacího makra **BPXYTHLI**.

Pravidla pro slučování generovaných MQRFH2

Následující pravidla platí pro sloučení vygenerovaného souboru MQRFH2 s existujícím souborem MQRFH2. Generované záhlaví MQRFH2 se sloučí s existujícím záhlavím MQRFH2 , pokud:

1. Existující soubor MQRFH2 je na stejné pozici IBM MQ by umístil vygenerovaný soubor MQRFH2nebo dříve v řetězci záhlaví.
2. CCSID generovaných vlastností je stejný jako NameValueCCSID existujícího MQRFH2.

Jinak se generované záhlaví umístí odděleně do vyrovnávací paměti, na pozici popsanou dříve.

Pravidla pro slučování složek v existujícím produktu MQRFH2

Pokud jsou vlastnosti zprávy sloučeny do existujícího souboru MQRFH2, bude existující soubor MQRFH2 vyhledán pro složky, které odpovídají vlastnostem zprávy, a sloučí je. Pokud odpovídající složka neexistuje, přidá se nová složka na konec existujících složek. Pokud odpovídající složka existuje, prohledá se. Všechny odpovídající vlastnosti jsou přepsány. Všechny nové se přidají na konec složky.

Omezení složky MQRFH2

Přehled omezení složek v záhlavích MQRFH2

Omezení MQRFH2 platí pro následující složky:

- Názvy prvků ve složce <usr> nesmí začínat předponou JMS ; takové názvy vlastností jsou vyhrazeny pro použití produktem JMS a nejsou platné pro vlastnosti definované uživatelem.

  Takový název prvku nezpůsobí selhání syntaktické analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy IBM MQ .

- Názvy prvků ve složce <usr> nesmí být ve směsi malých nebo velkých písmen, NULL, TRUE, FALSE, NOT, AND, OR, BETWEEN, LIKE, IN, IS a ESCAPE. Tyto názvy odpovídají klíčovým slovům SQL a ztěžují syntaktickou analýzu selektorů, protože <usr> je výchozí složka používaná v případě, že pro konkrétní vlastnost v selektoru není určena žádná složka.

  Takový název prvku nezpůsobí selhání syntaktické analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy IBM MQ .
- Model obsahu složky <usr> je následující:
  - Jako název prvku lze použít libovolný platný název XML za předpokladu, že neobsahuje dvojtečku.
  - Povoleny jsou pouze jednoduché prvky, nikoli vnořené složky.
  - Všechny prvky mají výchozí typ řetězce, pokud nejsou upraveny atributem dt="xxx" .
  - Všechny prvky jsou volitelné, ale ve složce by se neměly vyskytovat více než jednou.
- Názvy prvků ve všech složkách, u které se má za to, že obsahují vlastnosti zprávy, nesmí obsahovat tečku (.) (Znak Unicode U+002E), protože se používá v názvech vlastností k označení hierarchie.

  Takový název prvku nezpůsobí selhání syntaktické analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy IBM MQ .

Obecně lze říci, že záhlaví MQRFH2 , která obsahují platná data ve stylu XML, může produkt IBM MQ analyzovat bez selhání, ačkoli určité prvky MQRFH2 nejsou přístupné prostřednictvím rozhraní API vlastností zprávy IBM MQ .

Konflikty názvů prvku MQRFH2

Přehled konfliktů v názvech prvků MQRFH2 .

K vlastnosti zprávy lze připojit pouze jednu hodnotu. Pokud pokus o přístup k vlastnosti vede ke konfliktu hodnot, jeden je vybrán přednostně před druhým.

Syntaxe IBM MQ pro přístup k prvkům MQRFH2 umožňuje jedinečnou identifikaci prvku, pokud složka neobsahuje žádné prvky se stejným názvem. Pokud složka obsahuje více než jeden prvek se stejným názvem, hodnota použité vlastnosti je nejbližší k záhlaví zprávy.

To platí, pokud jsou dvě nebo více složek se stejným názvem obsaženy v různých významných záhlavích MQRFH2 v rámci stejné zprávy.

Ke konfliktu může dojít při zpracování volání MQGET poté, co byla vlastnost deskriptoru, která není zprávou, nastavena dvakrát: jak prostřednictvím volání MQSETMP, tak přímo v prvotním záhlaví MQRFH2 .

Pokud k tomu dojde, má vlastnost přidružená ke zprávě voláním rozhraní API přednost před vlastností v datech zprávy, tj. před vlastností v prvotním záhlaví MQRFH2 . Dojde-li ke konfliktu, má se za to, že přichází logicky před daty zprávy.

Mapování z názvů vlastností na názvy složek a prvků MQRFH2

Přehled rozdílů mezi názvy vlastností a názvy prvků v záhlaví MQRFH2 .

Používáte-li některá z definovaných rozhraní API, která v konečném důsledku generují záhlaví MQRFH2 , není název vlastnosti nutně názvem prvku ve složce MQRFH2 , aby bylo možné určit vlastnosti zprávy (například MQ JMS).

Proto dojde k mapování z názvu vlastnosti na prvek MQRFH2 a opačným způsobem s přihlédnutím jak k názvu složky, která obsahuje prvek, tak k názvu prvku. Některé příklady z adresáře IBM MQ classes for JMS jsou již zdokumentovány v souboru Použití IBM MQ classes for Java.

Tabulka 636. Názvy vlastností mapované na složku MQRFH2 a názvy prvků

Název vlastnosti	Název složky MQRFH2	Název prvku MQRFH2
JMSDestination	jms	Dst
JMSType	mcd	Type, Set, Fmt

Tabulka 636. Názvy vlastností mapované na složku MQRFH2 a názvy prvků (pokračování)		
Název vlastnosti	**Název složky MQRFH2**	**Název prvku MQRFH2**
xxx (definovaný uživatelem, kde xxx nezačíná na JMS)	usr	xxx

Proto, když aplikace JMS přistupuje k vlastnosti JMSDestination , mapuje se na prvek Dst ve složce <jms> .

Při zadávání vlastností jako prvků MQRFH2 definuje IBM MQ své prvky takto:

Tabulka 637. Názvy vlastností mapované na názvy složek, skupin a prvků MQRFH2			
Název vlastnosti	**Název složky MQRFH2**	**Název skupiny MQRFH2**	**Název prvku MQRFH2**
<Property>	<usr>	Není k dispozici.	<Property>
<folder>. <Property>	<folder>	Není k dispozici.	<Property>
<folder>. <group>. <Property>	<folder>	<group>	<Property>

Když se například aplikace IBM MQ pokusí o přístup k vlastnosti Property1 , mapuje se na prvek Property1 ve složce <usr> . Vlastnost wmq.Property2 se mapuje na vlastnost Property2 ve složce <wmq> .

Pokud název vlastnosti obsahuje více než jeden název. , použitý název prvku MQRFH2 je ten, který následuje za posledním. a skupiny MQRFH2 se používají k vytvoření hierarchie; vnořené skupiny MQRFH2 jsou povoleny.

K vlastnostem záhlaví JMS a specifickým vlastnostem poskytovatele, které jsou obsaženy v souboru MQRFH2 ve složkách <mcd>, <jms>a <mqext> , přistupuje aplikace IBM MQ s použitím krátkých názvů definovaných v části Použití IBM MQ classes for Java .

K vlastnostem definovaným uživatelem JMS se přistupuje ze složky <usr> . Aplikace IBM MQ může použít složku <usr> pro své vlastnosti aplikace, pokud je přijatelné, aby se tato vlastnost zobrazovala aplikacím JMS jako jedna z vlastností definovaných uživatelem.

Pokud to není přijatelné, zvolte jinou složku; složka <wmq_usr> je poskytnuta jako standardní umístění pro takové vlastnosti, které nejsouJMS .

Vaše aplikace mohou určit a používat libovolnou složku MQRFH2 s dobře definovaným použitím, které nejsou zdokumentovány v souboru “Vlastnosti určené jako prvky MQRFH2” na stránce 928 , pokud si poznamenáte následující:

1. Složka již může být používána nebo může být v budoucnu používána jinou aplikací, která poskytuje nedefinovaný přístup k vlastnostem, jež jsou v ní obsaženy. Doporučené konvence pojmenování pro názvy vlastností naleznete v tématu Názvy vlastností .
2. Vlastnosti nejsou přístupné pro předchozí verze klienta IBM MQ classes for JMS nebo XMS , které mají přístup pouze ke složce <usr> pro vlastnosti definované uživatelem.
3. Složka musí být označena atributem content s hodnotou nastavenou na properties, například content='properties'.

   Produkt “MQSETMP-Nastavení vlastnosti zprávy” na stránce 782 automaticky přidá tento atribut podle potřeby. Tento atribut nesmí být přidán do žádné ze složek IBM-defined, například <jms> a <usr>. Pokud tak učiníte, klient IBM MQ classes for JMS před IBM WebSphere MQ 7.0zprávu odmítne. s MessageFormatException.

Vzhledem k tomu, že složka <usr> je výchozím umístěním pro vlastnosti syntaxe <Property> , aplikace IBM MQ a aplikace JMS pro přístup ke stejné hodnotě vlastnosti definované uživatelem se stejným názvem.

Vyhrazené názvy složek

Existuje několik vyhrazených názvů složek. Nemůžete použít takové názvy, jako jsou předpony složek; například `Root.Property1` nemá přístup k platné vlastnosti, protože `Root` je vyhrazeno. Následující seznam obsahuje vyhrazené názvy složek:

- Kořen
- Tělo
- Vlastnosti
- Prostředí
- LocalEnvironment
- DestinationList
- ExceptionList
- InputBody
- InputRoot
- InputProperties
- InputLocalprostředí
- Seznam InputDestination
- Seznam InputException
- OutputRoot
- Prostředí OutputLocal
- OutputDestinationSeznam
- Seznam OutputException

Mapování polí deskriptoru vlastností do záhlaví MQRFH2

Když je vlastnost přeložena do prvku MQRFH2 , následující atributy prvku se používají k určení významných polí deskriptoru vlastnosti: Popisuje, jak jsou pole MQPD přeložena do atributů prvku MQRFH2 .

Podpora

Pole deskriptoru vlastnosti podpory je rozděleno na tři atributy prvku

- Atribut prvku **sr** určuje hodnoty v bitové masce MQPD_REJECT_UNSUP_MASK.
- Atribut prvku **sa** určuje hodnoty v bitové masce MQPD_ACCEPT_UNSUP_MASK.
- Atribut prvku **sx** určuje hodnoty v bitové masce MQPD_ACCEPT_UNSUP_IF_XMIT_MASK.

Tyto atributy prvku jsou platné pouze ve složce < mq> a jsou ignorovány, pokud jsou nastaveny na prvky v ostatních složkách, které obsahují vlastnosti.

Tabulka 638. Pole MQPD mapovaná na atributy prvku MQRFH2

Hodnota podpory	Atribut prvku MQRFH2	Hodnota atributu MQRFH2
MQPD_SUPPORT_OPTIONAL	sa	volitelné Toto je výchozí hodnota.
MQPD_SUPPORT_REQUIRED	sr	povinné
MQPD_SUPPORT_REQUIRED_IF_LOCAL	sx	lokální

Kontext

Pomocí atributu prvku **context** můžete označit kontext zprávy, ke kterému vlastnost patří. Použijte pouze jednu hodnotu. Tento atribut prvku je platný pro vlastnost v libovolné složce obsahující vlastnosti.

Tabulka 639. Hodnoty kontextu mapované na hodnoty atributu MQRFH2

Hodnota kontextu	Hodnota atributu MQRFH2
MQPD_NO_CONTEXT	Není Toto je výchozí hodnota.
MQPD_USER _CONTEXT	uživatel

CopyOptions

Pomocí atributu prvku **copy** můžete označit zprávy, do kterých se má vlastnost zkopírovat. Přijatelná je více než jedna hodnota; více hodnot oddělte čárkou. Platné jsou například hodnoty **copy='reply'** a **copy='publish,report'** . Tento atribut prvku je platný pro vlastnost v libovolné složce obsahující vlastnosti.

Poznámka: V definici atributu jsou platné jednoduché uvozovky nebo dvojité uvozovky, například **copy='reply'** nebo **copy="report"** .

Tabulka 640. Hodnoty CopyOption mapované na hodnoty atributu MQRFH2

Hodnota CopyOption	Hodnota atributu MQRFH2
MQPD_COPY_FORWARD	objekt forward
MQPD_COPY_REPLY	reply
MQPD_COPY_REPORT	sestava
MQPD_COPY_PUBLISH	publikovat
MQPD_COPY_ALL	vše Neuvádějte tuto hodnotu s žádnou jinou hodnotou. Při použití s jinou hodnotou má tato hodnota přednost před libovolnou hodnotou kromě hodnoty **none**.
MQPD_COPY_DEFAULT	default Toto je výchozí hodnota. Jde o ekvivalent k zadání tří hodnot MQCOPY_FORWARD, MQCOPY_REPORT a MQCOPY_PUBLISH. Neuvádějte tuto hodnotu s žádnou jinou hodnotou.
MQPD_COPY_NONE	Není Neuvádějte tuto hodnotu s žádnou jinou hodnotou. Při použití s jinou hodnotou má tato hodnota přednost.

Omezení složky < mq> MQRFH2

Je-li zpráva vložena do fronty, je vyhledána složka < mq>, aby mohla být zpracována v souladu s vlastnostmi definovanými pro MQ. Chcete-li povolit efektivní syntaktickou analýzu vlastností definovaných v produktu MQ, platí pro složku následující omezení:

- Pouze vlastnosti v první významné složce < mq> ve zprávě jsou produktem MQovlivněny; vlastnosti v jakékoli jiné složce < mq> ve zprávě jsou ignorovány.
- Pokud je složka v UTF-8, jsou ve složce povoleny pouze jednobajtové znaky UTF-8 . Vícebajtový znak ve složce může způsobit selhání syntaktické analýzy a odmítnutí zprávy.

- Nezahrnujte skupiny MQRFH2 do složky < mq>. Přítomnost znaku Unicode U+003C v hodnotě vlastnosti způsobí odmítnutí zprávy.
- Ve složce nepoužívejte únikové řetězce. Řídicí řetězec je považován za skutečnou hodnotu prvku.
- Pouze znak Unicode U+0020 je ve složce považován za mezerový znak. Všechny ostatní znaky jsou považovány za významné a mohou způsobit selhání syntaktické analýzy složky a odmítnutí zprávy.

Pokud se syntaktická analýza složky < mq> nezdaří nebo pokud složka nedodržuje tato omezení, bude zpráva odmítnuta s kódem CompCode **MQCC_FAILED** a kódem příčiny **MQRC_RFH_RESTRICTED_FORMAT_ERR**.

Záhlaví MQRFH2 , která nejsou platná

V době zpracování procesů volání MQPUT, MQPUT1nebo MQGET může dojít k částečné syntaktické analýze všech záhlaví MQRFH2 ve zprávě, aby bylo možné zkontrolovat, které složky jsou zahrnuty, a určit, zda složky obsahují vlastnosti. Přehled záhlaví MQRFH2 , která nejsou platná.

Pokud nelze částečnou analýzu zprávy úspěšně dokončit, protože struktura není platná, například pole `StrucLength` je příliš malé, pak:

- Volání MQPUT nebo MQPUT1 se nezdaří s kódem příčiny MQRC_RFH_ERROR, pokud lze určit, že aplikace obsahuje nějakou volbu IBM WebSphere MQ 7 , aby existující aplikace neselhaly.
- Volání MQGET se úspěšně vrátí a ve vyrovnávací paměti, kterou jste zadali, se vrátí MQRFH2 obsahující chybu.

Pokud dílčí syntaktická analýza selže, protože nelze zjistit, zda konkrétní složka obsahuje vlastnosti, či nikoli, například složka začne `<<jms`, takže syntaktická analýza selže před určením názvu složky, pak:

- Volání MQPUT nebo MQPUT1 se nezdaří s kódem příčiny MQRC_RFH_FORMAT_ERROR, pokud lze určit, že aplikace obsahuje nějakou volbu IBM WebSphere MQ 7 , takže existující aplikace neselžou.
- Volání MQGET se úspěšně vrátí a ve vyrovnávací paměti, kterou jste zadali, se vrátí MQRFH2 obsahující chybu.
- V rámci správce front není zpráva odmítnuta kvůli nesprávně formátované složce, ale se složkou je vždy zacházeno, jako by v ní nebyly obsaženy žádné vlastnosti.

Zpráva může procházet sítí správců front se složkou obsahující takovou chybu syntaxe, ale nikdy nebyla analyzována a zjištěna, zatímco jedna nebo více složek ve zprávě jsou:

- Platný
- Úspěšně analyzováno
- Používá se při zpracování zprávy

Proto není detekce zaručena.

Pokud některá z vašich aplikací používá pro přístup k vlastnosti produkt “MQSETMP-Nastavení vlastnosti zprávy” na stránce 782nebo MQINQMP, což způsobí úplnou analýzu složky MQRFH2 , a zjistí tak chybu, kterou nelze dokončit syntaktickou analýzu, je to označeno příslušným návratovým kódem pro volání rozhraní API. Aplikaci nejsou zpřístupněny žádné vlastnosti ve složce.

Dojde-li k pokusu o úplnou analýzu složky MQRFH2 a syntaktický analyzátor nalezne nerozpoznané atributy prvku nebo nerozpoznaný datový typ, syntaktická analýza bude pokračovat a úspěšně dokončena bez vydání varování; nejedná se o chybu syntaktické analýzy.

Převod kódové stránky

Tato část popisuje názvy kódových sad a CCSID, národní jazyk, převod z/OS , IBM i převod a podporu převodu Unicode.

V každé sekci národního jazyka jsou uvedeny následující informace:

- Podporované nativní CCSID
- Převody kódových stránek, které nejsou podporovány

V informacích se používají následující výrazy:


AIX
Označuje IBM MQ for AIX.


Linux
Označuje IBM MQ pro Linux pro Intel a IBM MQ pro Linux pro zSeries.


OS/400
Označuje IBM MQ for IBM i.


Windows
Označuje IBM MQ for Windows.


z/OS
Označuje IBM MQ for z/OS.

Předvolba pro převod dat je pro převod, který se má provést na cílovém (přijímacím) systému.

Pokud zdrojový produkt podporuje převod, lze kanál nastavit a vyměňovat data nastavením atributu kanálu CONVERT na hodnotu YES ve zdroji.

Poznámka:

1. Převod pro informace IBM MQ MQI client probíhá na serveru, takže server musí podporovat převod z CCSID klienta na CCSID serveru.
2. Převod může zahrnovat podporu přidanou pomocí CSD/PTF k nejnovější verzi produktu IBM MQ. Zkontrolujte obsah nejnovější úrovně služeb, abyste zjistili, zda potřebujete nainstalovat CSD/PTF, abyste tento převod povolili.
3. CCSID správce front IBM MQ musí být Smíšený nebo SBCS.
4. Některé identifikátory CCSID, například 850 v systému AIX, které nejsou podporovány operačním systémem, mohou být i nadále používány aplikací a také mohou být nastaveny jako identifikátor CCSID správce front IBM MQ . To je povoleno pouze pro účely zpětné kompatibility a převod selže, pokud nejsou nainstalovány příslušné převodní tabulky.

Viz Tabulka 641 na stránce 937 pro křížový odkaz mezi některými čísly CCSID a některými názvy průmyslových kódových sad.

Související odkazy

“Národní jazyky” na stránce 938
Tyto informace obsahují jazyky podporované produktem IBM MQ.

Názvy kódových sad a CCSID

Názvy kódových sad a odpovídající CCSID pro každý název kódové sady.


Produkt IBM MQ for z/OS poskytuje více převodů, než je uvedeno v tabulkách specifických pro daný jazyk. Úplný seznam převodů naleznete v části Tabulka 674 na stránce 963.

Tabulka 641. Názvy kódových sad a CCSID

Názvy kódové sady	**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

Tabulka 641. Názvy kódových sad a CCSID (pokračování)	
Názvy kódové sady	**CCSID**
ISO 8859-8	916
ISO 8859-9	920
ISO 8859-13	921
ISO 8859-15 (v eurech)	923
big5	950
eucJP	954 5050 33722
eucKR	970
eucTW	964
eucCN	1383
PCK	943
GBK	1386
koi8-r	878

Národní jazyky

Tyto informace obsahují jazyky podporované produktem IBM MQ.

Jazyky podporované produktem IBM MQ jsou:

- Americká angličtina-viz téma “Česká republika” na stránce 939
- Němčina-viz téma “Němčina” na stránce 939
- Dánština a norština-viz téma “Dánština a norština” na stránce 940
- Finština a švédština-viz téma “Finština a švédština” na stránce 941
- Italština-viz téma “Italština” na stránce 942
- Španělština-viz téma “Španělština” na stránce 942
- Britská angličtina/gaelština-viz téma “Angličtina (UK) /gaelština” na stránce 943
- Francouzština-viz téma “Francouzština” na stránce 944
- Vícejazyčné-viz téma “Vícejazyčné” na stránce 944
- Portugalština-viz téma “Portugalština” na stránce 945
- Islandština-viz téma “Islandština” na stránce 946
- Východoevropské jazyky-viz téma “Východoevropské jazyky” na stránce 946
- Cyrilice-viz téma “Cyrilice” na stránce 948
- Estonština-viz téma “Estonština” na stránce 948
- Lotyština a litevština-viz téma “Lotyština a litevština” na stránce 950
- Ukranian-viz téma “Ukrajinština” na stránce 951
- Řečtina-viz téma “Řečtina” na stránce 951
- Turečtina-viz téma “Turečtina” na stránce 952
- Hebrejština-viz téma “Hebrejština” na stránce 953
- Farsi-viz téma “Farsi” na stránce 955
- Urdu-viz téma “Urdština” na stránce 955
- Thajsky-viz téma “Thajština” na stránce 955

- Lao-viz téma “Laoština” na stránce 956
- Vietnamština-viz téma “Vietnamština” na stránce 956
- Japonská latina SBCS-viz téma “Japonská latina SBCS” na stránce 957
- Japonská Katakana SBCS-viz téma “Japonská Katakana SBCS” na stránce 958
- Japonský Kanji/Latin Mixed-viz téma “Smíšené červené, fialové a růžové květy” na stránce 959
- Japonský Kanji/Katakana Mixed-viz téma “Smíšené červené, fialové a růžové květy” na stránce 960
- Korejština-viz téma “Korejština” na stránce 961
- Zjednodušená čínština-viz téma “Zjednodušená čínština” na stránce 962
- Tradiční čínština-viz téma “Tradiční čínština” na stránce 962

Česká republika

Podrobnosti o CCSID a konverzi CCSID pro americkou angličtinu.

Tabulka 642. Nativní CCSID pro americkou angličtinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	37, 924, 1140
AIX AIX	819, 923, 5348
Windows Windows	437, 850, 1252, 5348, 858
Linux Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

37
Nepřevádí se na kódové stránky 923, 858

924
Nepřevádí se na kódové stránky 437, 858, 1051, 1140, 1252, 1275, 5348

1140
Nepřevádí na kódové stránky 924, 1051, 1275

Němčina

Podrobnosti o CCSID a konverzi CCSID pro němčinu.

Tabulka 643. Nativní identifikátory CCSID pro němčinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	273, 924, 1141

Tabulka 643. Nativní identifikátory CCSID pro němčinu na podporovaných platformách (pokračování)	
Platforma	**Nativní CCSID**
AIX	819, 923, 5348
Windows	437, 850, 858, 1252, 5348
Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

273
Nepřevádí na kódové stránky 858, 923, 924, 1275

924
Nepřevádí se na kódové stránky 273, 437, 858, 1051, 1141, 1252, 1275, 5348

1141
Nepřevádí na kódové stránky 924, 1051, 1275

Dánština a norština

Podrobnosti o CCSID a konverzi CCSID pro dánštinu a norštinu.

Tabulka 644. nativní CCSID pro dánštinu a norštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	277, 924, 1142
AIX	819, 923, 5348
Windows	850, 858, 865, 1252, 5348
Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

277
Nepřevádí na kódové stránky 858, 923, 924, 1275

924
Nepřevádí se na kódové stránky 277, 858, 865, 1051, 1142, 1252, 1275, 5348

1142
Nepřevede na kódové stránky 924, 865, 1051, 1275

AIX

Kódová stránka:

819
Nepřevede na kódovou stránku 865

Windows

Kódová stránka:

865
Nepřevádí se na kódové stránky 1051, 1275

Finština a švédština

Podrobnosti o CCSID a konverzi CCSID pro finštinu a švédštinu.

Tabulka 645. Nativní CCSID pro finštinu a švédštinu na podporovaných platformách

Platforma	Nativní CCSID
 IBM i  z/OS	278, 924, 1143
 AIX	819, 923, 5348
 Windows	437, 850, 858, 865, 1252, 5348
 Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

Kódová stránka:

278
Nepřevádí na kódové stránky 858, 923, 924, 1275

924
Nepřevádí se na kódové stránky 278, 437, 858, 865, 1051, 1143, 1252, 1275, 5348

1143
Nepřevádí se na kódové stránky 865, 924, 1051, 1275

AIX

AIX

Kódová stránka:

819
Nepřevede na kódovou stránku 865

850
Nepřevede na kódovou stránku 865

Windows

Windows

Kódová stránka:

865
Nepřevádí se na kódové stránky 1051, 1275

Italština

Podrobnosti o CCSID a konverzi CCSID pro italštinu.

Tabulka 646. Nativní identifikátory CCSID pro italštinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	280, 924, 1144
AIX AIX	819, 923, 5348
Windows Windows	437, 850, 858, 1252, 5348
Linux Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

280
Nepřevádí na kódové stránky 858, 923, 924, 1275

924
Nepřevádí se na kódové stránky 280, 437, 858, 1051, 1144, 1252, 1275, 5348

1144
Nepřevádí na kódové stránky 924, 1051, 1275

Španělština

Podrobnosti o CCSID a konverzi CCSID pro španělštinu.

Tabulka 647. Nativní CCSID pro španělštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	284, 924, 1145
AIX	819, 923, 5348
Windows	437, 850, 858, 1252, 5348
Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

284
Nepřevádí na kódové stránky 858, 923, 924, 1275

924
Nepřevádí se na kódové stránky 284, 437, 858, 1051, 1145, 1252, 1275, 5348

1145
Nepřevádí na kódové stránky 924, 1051, 1275

Angličtina (UK) /gaelština

Podrobnosti o CCSID a konverzi CCSID pro britskou angličtinu/gaelštinu.

Tabulka 648. Nativní CCSID pro britskou angličtinu/gaelštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	285, 924, 1146
AIX	819, 923, 5348
Windows	437, 850, 858, 1252, 5348
Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

285
Nepřevádí na kódové stránky 858, 923, 924, 1275

924
Nepřevádí se na kódové stránky 285, 437, 858, 1051, 1146, 1252, 1275, 5348

1146
Nepřevádí na kódové stránky 924, 1051, 1275

Francouzština

Podrobnosti o CCSID a konverzi CCSID pro francouzštinu.

Tabulka 649. Nativní identifikátory CCSID pro francouzštinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	297, 924, 1147
AIX AIX	819, 923, 5348
Windows Windows	437, 850, 858, 1252, 5348
Linux Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

297
Nepřevádí se na kódové stránky 858, 923, 924, 1275, 5348

924
Nepřevádí se na kódové stránky 297, 437, 858, 1051, 1147, 1252, 1275, 5348

1147
Nepřevádí na kódové stránky 924, 1051, 1275

Vícejazyčné

Podrobnosti o CCSID a konverzi CCSID pro Multilingual.

Tabulka 650. Nativní identifikátory CCSID pro vícejazyčný převod na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	500, 924, 1148
AIX AIX	819, 923, 5348
Windows Windows	437, 850, 858, 1252, 5348

Tabulka 650. Nativní identifikátory CCSID pro vícejazyčný převod na podporovaných platformách (pokračování)	
Platforma	**Nativní CCSID**
Linux Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

500
Nepřevádí se na kódové stránky 858, 923

924
Nepřevádí se na kódové stránky 437, 858, 1051, 1148, 1252, 1275, 5348

1148
Nepřevádí na kódové stránky 924, 1051, 1275

Portugalština

Podrobnosti o CCSID a konverzi CCSID pro portugalštinu.

Tabulka 651. Nativní identifikátory CCSID pro portugalštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i	37, 500, 924, 1140
z/OS IBM i	500, 924, 1140
AIX AIX	819, 923, 5348
Windows Windows	850, 858, 860, 1252, 5348
Linux Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

37
Nepřevádí na kódové stránky 858, 923, 1275

500
Nepřevádí na kódové stránky 858, 923, 1275

924
Nepřevádí se na kódové stránky 858, 860, 1051, 1140, 1252, 1275, 5348

1140
Nepřevádí se na kódové stránky 860, 924, 1051, 1275

Windows

Windows

Kódová stránka:

860
Nepřevádí se na kódové stránky 1051, 1275

Islandština

Podrobnosti o CCSID a konverzi CCSID pro islandštinu.

Tabulka 652. Nativní CCSID pro islandštinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	871, 924, 1149
AIX AIX	819, 923, 5348
Windows Windows	850, 858, 861, 1252, 5348
Linux Linux	819, 923
Klient Apple	1275

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

871
Nepřevádí se na kódové stránky 858, 923, 924, 1275, 5348

924
Nepřevádí se na kódové stránky 858, 861, 871, 1051, 1149, 1252, 1275, 5348

1149
Nepřevádí na kódové stránky 924, 1051, 1275

Windows

Windows

Kódová stránka:

861
Nepřevádí se na kódové stránky 1051, 1275

Východoevropské jazyky

Podrobnosti o CCSID a konverzi CCSID pro východoevropské jazyky. Mezi typické jazyky, které používají tyto CCSID, patří albánština, chorvatština, čeština, maďarština, polština, rumunština, srbština, slovenština a slovinština.

Tabulka 653. Nativní CCSID pro východoevropské jazyky na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	870, 1153
Windows Windows	852, 1250, 5346, 9044
AIX AIX Linux Linux	912
Východoevropský klient Apple	1282
Rumunský klient Apple	1285
Chorvatský klient Apple	1284

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

870
Nepřevádí na kódové stránky 1284, 1285

1153
Nepřevádí na kódové stránky 1250, 1284, 1285

IBM i

IBM i

Kódová stránka:

870
Nepřevádí se na kódové stránky 1284, 1285, 5346, 9044

1153
Nepřevádí se na kódové stránky 1282, 1284, 1285, 5346, 9044

, Linux

Linux

Kódová stránka:

912
Nepřevádí na kódové stránky 1284, 1285

Windows

Windows

Kódová stránka:

852
Nepřevádí na kódové stránky 1284, 1285

1250
Nepřevádí na kódové stránky 1284, 1285

9044
Nepřevádí se na kódové stránky 912, 1282, 1284, 1285

Cyrilice

Podrobnosti o CCSID a konverzi CCSID pro cyrilici. Mezi typické jazyky, které používají tyto CCSID, patří běloruština, bulharština, makedonština, ruština a srbština.

Tabulka 654. Nativní identifikátory CCSID pro cyrilici na podporovaných platformách

Platforma	**Nativní CCSID**
z/OS z/OS	1025
IBM i IBM i	880, 1025
Windows Windows	855, 866, 1131, 1251, 5347
AIX AIX Linux Linux	915
Klient Apple	1283

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

880
Nepřevádí se na kódové stránky 855, 866, 878, 1131, 5347

1025
Nepřevede na kódové stránky 878, 5347

Windows

Windows

Kódová stránka:

855
Nepřevede na kódovou stránku 1131

866
Nepřevede na kódovou stránku 1131

1131
Nepřevádí se na kódové stránky 855, 866, 880, 1283

Estonština

Podrobnosti o CCSID a konverzi CCSID pro estonštinu.

Tabulka 655. Nativní identifikátory CCSID pro estonštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	1122, 1157
Windows Windows	902, 922, 1257, 5353, 9449
AIX AIX Linux Linux	902, 922

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

1122
Neprovádí převod na kódové stránky 902, 1157, 9449

1157
Nepřevádí na kódové stránky 922, 1122, 1257, 9449

IBM i

IBM i

Kódová stránka:

1122
Nepřevádí se na kódové stránky 902, 5353, 9449

1157
Nepřevádí na kódové stránky 922, 5353, 9449

Linux

Linux

Kódová stránka:

902
Nepřevádí na kódové stránky 922, 1122, 9449

922
Neprovádí převod na kódové stránky 902, 1157, 9449

Windows

Windows

Kódová stránka:

5353
Nepřevede na kódovou stránku 9449

9449
Nepřevádí se na kódové stránky 902, 922, 1122, 1157, 1257, 5353

902
Nepřevádí na kódové stránky 922, 1122, 9449

Lotyština a litevština

Podrobnosti o CCSID a konverzi CCSID pro lotyšskou a litevskou.

Tabulka 656. Nativní CCSID pro lotyšský a litevský jazyk na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	1112, 1156
Windows	901, 921, 1257, 5353, 9449
AIX Linux	901, 921

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

1112
Nepřevádí na kódové stránky 901, 1156, 9449

1156
Nepřevádí na kódové stránky 901, 1156, 9449

IBM i

IBM i

Kódová stránka:

1112
Nepřevede na kódovou stránku 5353

1153
Nepřevádí se na kódové stránky 921, 5353, 9449

Linux

Linux

Kódová stránka:

902
Nepřevádí na kódové stránky 921, 1112, 1257, 9449

921
Nepřevádí na kódové stránky 901, 1156, 9449

Windows

Windows

Kódová stránka:

901
Nepřevádí na kódové stránky 921, 1112, 1257, 9449

5355
Nepřevede na kódovou stránku 9449

9449
Nepřevádí se na kódové stránky 901, 921, 1112, 1156, 1257

Ukrajinština
Podrobnosti o CCSID a konverzi CCSID pro ukrajinskou.

Tabulka 657. Nativní CCSID pro Ukranian na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	1123
Windows Windows	1124, 1125, 1251, 5347
AIX AIX Linux Linux	1124

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i
IBM i

Kódová stránka:

1123
Neprovádí převod na kódovou stránku 5347

Windows
Windows

Kódová stránka:

1125
Nepřevede na kódovou stránku 1123

Řečtina
Podrobnosti o CCSID a konverzi CCSID pro řečtinu.

Tabulka 658. Nativní CCSID pro řečtinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	875
Windows Windows	869, 1253, 5349

Tabulka 658. Nativní CCSID pro řečtinu na podporovaných platformách (pokračování)	
Platforma	**Nativní CCSID**
AIX AIX Linux Linux NCR	813
Klient Apple	1280
Klient systému DOS	737

Všechny neklientské platformy podporují převod mezi svými nativními identifikátory CCSID, nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

875
Nepřevede na kódovou stránku 5349

Windows

Windows

Kódová stránka:

1253
Nepřevede na kódovou stránku 737

5349
Nepřevede na kódovou stránku 737

Turečtina

Podrobnosti o CCSID a konverzi CCSID pro turečtinu.

Tabulka 659. Nativní identifikátory CCSID pro turečtinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	1026
Windows Windows	857, 1254, 5350
AIX AIX Linux Linux	920
Klient Apple	1281

Všechny neklientské platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID ostatních platforem s následujícími výjimkami.

IBM i

Kódová stránka:

1026
Nepřevede na kódovou stránku 5350

Hebrejština

Podrobnosti o CCSID a konverzi CCSID pro hebrejštinu.

Tabulka 660. Nativní CCSID pro hebrejštinu na podporovaných platformách

Platforma	Nativní CCSID
 z/OS	424, 803, 4899, 12712
 IBM i	424
 AIX	916, 9048
 Windows	1255, 5351
 Linux	916

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

424
Nepřevádí na kódové stránky 867, 4899, 9048, 12712

803
Nepřevádí se na kódové stránky 867, 4899, 5351, 9048, 12712

4899
Nepřevádí se na kódové stránky 424, 803, 856, 862, 916, 1255

12712
Nepřevádí na kódové stránky 424, 803, 856, 916, 1255

IBM i

Kódová stránka:

424
Nepřevádí se na kódové stránky 803, 867, 4899, 5351, 9048, 12712

Kódová stránka 424 také převádí na a z CCSID 4952, což je varianta 856.

AIX

Kódová stránka:

916
Nepřevádí na kódové stránky 867, 4899, 9048, 12712

9048
Nepřevádí se na kódové stránky 424, 803, 856, 862, 916, 1255

Windows

Windows

Kódová stránka:

1255
Nepřevádí na kódové stránky 867, 4899, 9048, 12712

5351
Nepřevede na kódovou stránku 803

Arabština

Podrobnosti o CCSID a konverzi CCSID pro arabštinu

Tabulka 661. Nativní CCSID pro arabštinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	420
AIX AIX	1046, 1089
	1089 (viz poznámka)
Windows Windows	720, 864, 1256, 5352
Linux Linux	1089

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

420
Nepřevede na kódovou stránku 5352

Linux, Tru64

Linux

Kódová stránka:

1089
Nepřevede na kódovou stránku 720

Windows

Windows

Kódová stránka:

720
Nepřevádí se na kódové stránky 1089, 5352

5352
Nepřevede na kódovou stránku 720

Farsi
Podrobnosti o CCSID a konverzi CCSID pro Farsi.

Tabulka 662. Nativní CCSID pro perštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	1097
AIX Linux Windows	1098 (viz poznámka)

Poznámka: Nativní CCSID pro tyto platformy nebyl standardizován a může se změnit.

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem.

Urdština
Podrobnosti o CCSID a konverzi CCSID pro urdštinu.

Tabulka 663. Nativní CCSID pro urdštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	918
Windows	868
AIX Linux	1006

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i
IBM i

Kódová stránka:

918
Nepřevede na kódovou stránku 1006

Thajština
Podrobnosti o CCSID a konverzi CCSID pro thajštinu.

Tabulka 664. Nativní CCSID pro thajštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	838
AIX Linux Windows	874 (viz poznámka)

Poznámka: Nativní CCSID pro tyto platformy nebyl standardizován a může se změnit.

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem.

Laoština

Podrobnosti o CCSID a konverzi CCSID pro Lao.

Tabulka 665. Nativní identifikátory CCSID pro Lao na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	1132
AIX Linux Windows	1133

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem.

Vietnamština

Podrobnosti o CCSID a konverzi CCSID pro vietnamštinu.

Tabulka 666. Nativní identifikátory CCSID pro vietnamštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	1130
Windows	1258, 5354
AIX Linux	1129

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

1130
Nepřevádí se na kódové stránky 1129, 5354

Japonská latina SBCS

Podrobnosti o CCSID a konverzi CCSID pro japonskou latinku SBCS.

Tabulka 667. Nativní identifikátory CCSID pro japonskou latinku SBCS na podporovaných platformách

Platforma	Nativní CCSID
 IBM i  z/OS	1027
 AIX	932, 5050, 33722 (viz poznámka 1)
 Windows	932, 943 (viz poznámka 2)
 Linux	943, 5050

Poznámka:

1.  5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na webu AIX. CCSID ohlášený operačním systémem je 33722.
2.  Produkt Windows NT používá kódovou stránku 932, ale toto je nejlépe reprezentováno CCSID 943. Avšak ne všechny platformy produktu IBM MQ podporují tento CCSID.

   V systému IBM MQ for Windows se používá CCSID 932 ke znázornění kódové stránky 932, ale lze provést změnu souboru `../conv/table/ccsid.tbl` , která změní CCSID použitý na 943.

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

1027
Nepřevádí se na kódové stránky 932, 942, 943, 954, 5050, 33722

IBM i

Kódová stránka:

1027
Nepřevede na kódovou stránku 932

AIX

Kódová stránka:

932
Nepřevede na kódovou stránku 1027

5050
Nepřevede na kódovou stránku 1027

33722
Nepřevede na kódovou stránku 1027

Linux

Linux

Kódová stránka:

943
Nepřevede na kódovou stránku 1027

5050
Nepřevede na kódovou stránku 1027

Japonská Katakana SBCS

Podrobnosti o CCSID a konverzi CCSID pro japonskou Katakanu SBCS.

Tabulka 668. Nativní CCSID pro japonštinu Katakana SBCS na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	290
AIX AIX	932, 5050, 33722 (viz poznámka 1)
Windows Windows	932, 943 (viz poznámka 2)
Linux Linux	943, 5050

Poznámka:

1. AIX 5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na webu AIX. CCSID ohlášený operačním systémem je 33722.
2. Windows Produkt Windows NT používá kódovou stránku 932, ale toto je nejlépe reprezentováno CCSID 943. Avšak ne všechny platformy produktu IBM MQ podporují tento CCSID.

   V systému IBM MQ for Windows se používá CCSID 932 ke znázornění kódové stránky 932, ale lze provést změnu souboru `../conv/table/ccsid.tbl` , která změní CCSID použitý na 943.
3. Kromě předchozích převodů produkt IBM MQ podporuje převod z CCSID 897 na CCSID 37, 273, 277, 278, 280, 284, 285, 290, 297, 437, 500, 819, 850, 1027 a 1252 na následujících platformách:
   - AIX AIX
   - Linux Linux

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

290
Nepřevádí se na kódové stránky 932, 943, 954, 5050, 33722

IBM i

Kódová stránka:

290
Nepřevede na kódovou stránku 932

AIX

Kódová stránka:

932
Nepřevádí na kódové stránky 290, 897

5050
Nepřevádí na kódové stránky 290, 897

33722
Nepřevádí na kódové stránky 290, 897

Linux

Kódová stránka:

943
Nepřevádí na kódové stránky 290, 897

5050
Nepřevádí na kódové stránky 290, 897

Smíšené červené, fialové a růžové květy

Podrobnosti o CCSID a konverzi CCSID pro japonský smíšený kód Kanji/latinku.

Tabulka 669. Nativní CCSID pro japonštinu Kanji/latinku Smíšené na podporovaných platformách

Platforma	Nativní CCSID
 IBM i  z/OS	1399, 5035 (viz poznámka 1)
 AIX	932, 5050, 33722 (viz poznámka 2)
 Windows	932, 943 (viz poznámka 4)
 Linux	943, 5050

Poznámka:

1.  5035 je CCSID související s kódovou stránkou 939
2.  5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na webu AIX. CCSID ohlášený operačním systémem je 33722.

3. Windows Produkt Windows NT používá kódovou stránku 932, ale toto je nejlépe reprezentováno CCSID 943. Avšak ne všechny platformy produktu IBM MQ podporují tento CCSID.

   V systému IBM MQ for Windows se používá CCSID 932 ke znázornění kódové stránky 932, ale lze provést změnu souboru `../conv/table/ccsid.tbl` , která změní CCSID použitý na 943.

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

1399
Nepřevádí se na kódové stránky 954, 5035, 5050, 33722

5035
Nepřevádí se na kódové stránky 954, 1399, 5050, 33722

IBM i

IBM i

Kódová stránka:

1399
Nepřevede na kódovou stránku 5039

5035
Nepřevede na kódovou stránku 5039

Smíšené červené, fialové a růžové květy

Podrobnosti o CCSID a konverzi CCSID pro japonskou směs Kanji/Katakana.

Tabulka 670. Nativní CCSID pro japonštinu Kanji/Katakana Smíšený na podporovaných platformách

Platforma	Nativní CCSID
z/OS z/OS	1390, 5026 (viz poznámka “1” na stránce 960)
IBM i IBM i	5026 (viz poznámka “1” na stránce 960)
AIX AIX	932, 5050, 33722 (viz poznámka “2” na stránce 961)
Windows Windows	932, 943 (viz poznámka “3” na stránce 961)
Linux Linux	943, 5050

Poznámka:

1. IBM i z/OS Jednobajtový režim CCSID 1390 a 5026 v EBCDIC obsahuje malá písmena na různých místech typického nebo invariantního rozvržení pro základní latinku. Proto je třeba dbát na to, aby při převodu dat zpráv na jiné identifikátory CCSID nedošlo ke ztrátě dat. Použití těchto identifikátorů CCSID jako výchozího identifikátoru CCSID správce front může také způsobit problémy při komunikaci s jinými správci front. Například názvy kanálů používající malá písmena nemusí být na vzdáleném systému správně interpretovány. 5026 je CCSID související s kódovou stránkou 930. CCSID 5026 je CCSID ohlášený v systému IBM i , když je vybrána funkce japonské Katakany (DBCS).

2. AIX 5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na webu AIX. CCSID ohlášený operačním systémem je 33722.
3. Windows Produkt Windows NT používá kódovou stránku 932, ale toto je nejlépe reprezentováno CCSID 943. Avšak ne všechny platformy produktu IBM MQ podporují tento CCSID.

   V systému IBM MQ for Windowsse k reprezentaci kódové stránky 932 používá CCSID 932, ale lze provést změnu souboru `../conv/table/ccsid.tbl` , která změní CCSID použitý na 943.

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

1390
Nepřevádí se na kódové stránky 954, 5026, 5050, 33722

Nepřijímá malá písmena.

5026
Nepřevádí se na kódové stránky 954, 1390, 5050, 33722

IBM i

IBM i

Kódová stránka:

5026
Nepřevádí na kódové stránky 1390, 5039

Korejština

Podrobnosti o CCSID a konverzi CCSID pro korejštinu.

Tabulka 671. Nativní identifikátory CCSID pro korejštinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i IBM i z/OS z/OS	933, 1364
AIX AIX Linux Linux	970
Windows Windows	949, 1363

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

933
Nepřevede na kódovou stránku 970

1364
Nepřevede na kódovou stránku 970

Zjednodušená čínština
Podrobnosti o CCSID a konverzi CCSID pro zjednodušenou čínštinu.

Tabulka 672. Nativní CCSID pro zjednodušenou čínštinu na podporovaných platformách

Platforma	Nativní CCSID
z/OS z/OS	935, 1388
IBM i IBM i	935, 1388
AIX AIX	1383, 1386
Windows Windows	1381, 1386 (viz poznámka 2)
Linux Linux	1383

Poznámka:

1. Windows Windows používá kódovou stránku 936, ale toto je nejlépe reprezentováno CCSID 1386. Avšak ne všechny platformy produktu IBM MQ podporují tento CCSID.

   Na IBM MQ for Windows CCSID 1381 se používá ke znázornění kódové stránky 936, ale lze provést změnu souboru `../conv/table/ccsid.tbl` , která změní CCSID použitý na 1386.

2. Produkt IBM MQ podporuje čínský standard GB18030 .

   Windows Linux z/OS V systémech z/OS, Windows a Linuxje podpora převodu poskytována mezi kódováním Unicode (UTF-8 a UTF-16) a CCSID 1388 (EBCDIC s rozšířeními GB18030), Unicode (UTF-8 a UTF-16) a CCSID 5488 (GB18030) a mezi CCSID 1388 a CCSID 5488.

   Poznámka: Aby bylo možné používat znaky GB18030 , musí být CCSID nastaven na 5488. Není však možné nastavit CCSID ve správci front vytvořeném pomocí IBM MQ Explorer nebo IBM MQ Console. Místo toho musíte buď vytvořit správce front pomocí rozhraní příkazového řádku s identifikátorem CCSID 5488, nebo použít příkazový řádek rozhraní CLI ke změně identifikátoru CCSID po vytvoření správce front.

   IBM i V systému IBM iposkytuje operační systém podporu pro převod mezi kódováním Unicode (UTF-8 a UTF-16) a CCSID 1388 (EBCDIC s příponami GB18030).

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS
z/OS

Kódová stránka:

935
Neprovádí převod na kódovou stránku 1383

1388
Neprovádí převod na kódovou stránku 1383

Tradiční čínština
Podrobnosti o CCSID a konverzi CCSID pro tradiční čínštinu.

Tabulka 673. Nativní identifikátory CCSID pro tradiční čínštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	937
Windows Windows	950
AIX AIX Linux Linux	950, 964

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

937
Nepřevede na kódovou stránku 964

1388
Neprovádí převod na kódovou stránku 1383

Linux

Linux

Kódová stránka:

964
Nepřevede na kódovou stránku 938

z/OS z/OS podpora převodu

Seznam podporovaných konverzí CCSID.

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID	
CCSID	**Převádí do a z CCSID**
37	256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 897, 903-905, 912, 920-924, 1009, 1025-1027, 1043, 1040, 8402, 997, 997, 92097284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 455, 455, 455, 455, 455, 455, 455, 455, 455, 455, 455, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 855, 8, 8, 855, 8, 8, 8, 8, 8, 855, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8,, 8, 8, 8, 8,,,, 8, 8, 8, 8,,,, 8,,, 8, 8, č, č, 8, č,,,,, č, č, č, č, č, č, č, č, č, č, č, č, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664, 28709

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
256	37, 273, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 819, 833, 836, 838, 850, 852, 857, 860-866, 869-871, 875, 880, 905, 1025-1027, 1112, 1122, 1200, 1208, 1251-1252, 1275, 4386, 4929, 4932, 9034, 4946, 4948, 4953
259	437, 808, 850-852, 855-858, 860-865, 867, 869, 872, 874, 899, 901-902, 915, 1098, 1161-1162, 1200, 1208, 1250-1258, 4946, 4948, 4951-4953, 4960, 4970, 5346, 5348, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584
273	37, 256, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 836, 838, 850, 852, 855-858, 860-865, 869-871, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025, 1040-1043, 1047, 1047, 1047, 1088, 449590, 490, 490, 4402, 427960, 427427960, 427960, 427425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 417, 417, 625, 425, 625, 625, 4, 425, 4, 4, 4, 4, 425, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 8, 4,,
274	500, 1047
275	37, 437, 500, 819, 850, 1047, 1200, 1208, 1252, 4946, 5348, 8229, 13488, 17584, 28709
277	37, 256, 273, 278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 916, 920, 923-924, 1009, 1025-1043, 1047, 1051, 1051, 951, 920, 9602, 4402, 4402, 4282, 4284, 4284, 428417, 417, 416, 417, 416, 416, 416, 416, 416, 416, 425, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 4, 416, 4, 416, 4, 4, 455, 455, 455, 8, 4,
278	37, 256, 273, 277, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-875, 880, 897, 903, 916, 920, 923-924, 1009, 1025-1052, 4940-1043, 1047, 1051, 951, 920, 923-924, 1009
280	37, 256, 273, 277-278, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 903, 916, 920, 923-924, 1025-1027, 104940-1043, 1047, 1051, 1051, 920, 920, 923, 4402
281	1047
282	500, 1047, 1200, 1208, 13488, 17584
284	37, 256, 273, 277-278, 280, 285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 916, 920, 923-924, 1009, 10486, 1040-1043, 1047, 1051, 1051, 4402, 449490, 490, 490, 916

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
285	37, 256, 273, 277-278, 280, 284, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 880, 897, 903, 912, 920, 923-924, 1025-1027, 10114109052, 4990490449625, 494259, 494259, 4259602, 9.
290	37, 256, 273, 277-278, 280, 284-285, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4929, 4932, 494949453, 4948, 4948, 49417, 490
293	1200, 1208, 13488, 17584
297	, 737, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 875, 880, 897, 903, 912, 916, 920, 923-924, 1025-27, 1040-1043, 1049490, 1049449470, 880, 897, 903, 920, 4923-924, 1025, 4944944949, 1051, 49449449425, 449425, 449449760, 9., 449449425, 449425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 425, 4ve, 4ve, 4ve, 4ve, 4ve, 4ve, 4ve, 460, 460, 4ve, 460, 460, 460, 460, 460, 460, 460, 4, 460, 6, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 460, 760, 460, 4, 760, 760, 760, 760, 760, 760, 760, 760, 760, 760, 760, 760, 760, 760, 4, 4, 760, 760, 760, 760, 760, 4, 4, 4, 4, 4, 4, 4, 760, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
300	301, 941, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
301	300, 941, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
367	37, 256, 273, 277-278, 280, 284, 290, 297, 500, 819, 833, 836, 850, 871, 875, 1009, 1026-1027, 1041, 1088, 1115, 1126, 1200, 1208, 4386, 4929, 4932, 4946, 4971, 5123, 5211, 8229, 8482, 9025, 13121, 13488, 17584, 25617, 25664, 28709
420	37, 256, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 8612, 9044, 9049, 9056, 9238, 13488, 16804, 17248, 17584,
423	37, 256, 273, 277-278, 280, 284-285, 297, 437, 500, 737, 775, 813, 819, 838, 850-852, 857, 860-865, 869-871, 874-875, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 49909, 4946, 4946
424	37, 256, 420, 437, 500, 737, 775, 803, 819, 836, 850, 852, 856-857, 860-865, 916, 1112, 1122, 1200, 1208, 1252, 1255, 4932, 4946, 4948, 4952-4953, 4960, 5012, 5351, 8229, 8612, 9044, 9049, 9056, 13488, 16804, 17248, 17584, 28709
437	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-863, 865-866, 874-875, 880, 897, 903, 905, 912, 912, 920-924, 1025-1043, 1043 25473, 25479, 25617, 25619, 28709

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
500	737, 273-275, 277-278, 280, 282, 284-285, 290, 297, 367, 420, 423-424, 737, 775, 813, 819, 833, 836, 838, 850-852, 855-858, 860-866, 869-871, 874-875, 880, 891, 897, 903-905, 912, 920-924, 1004, 1004, 1009-1021, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 9238, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664, 28709
720	37, 420, 864, 1200, 1208, 1256, 4960, 8229, 8612, 9056, 13488, 16804, 17248, 17584, 28709
737	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 833, 836, 838, 850, 869-871, 875, 880, 905, 1025-1027, 1097, 1200, 1208, 1252-1253, 1280, 4386, 4909, 4929, 4932, 4934, 4946, 4971, 5123, 8029, 8482, 8612, 9025, 9030
775	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 833, 836, 838, 850, 870-871, 875, 880, 905, 1025-1027, 1097, 1112, 1122, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 8229, 8482, 8612, 9025, 90430, 13121
803	424, 819, 850, 856, 862, 916, 1200, 1208, 1252, 1255, 4946, 4952, 5012, 13488, 17584
806	1200, 1208, 13488, 17584
808	259, 858-859, 872, 923-924, 1140, 1148, 1153-1154, 1200, 1208, 5347, 5348, 13488, 17584
813	28284-285, 297, 423, 437, 500, 737, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1253, 1280, 4909, 4934, 4946, 4948, 4953, 4970790, 490, 490, 490, 490, 590, 590, 490, 460, 12960, 1290, 129, 129, 12, 129, 412, 412, 412, 12, 12, 412, 4, 4, 4, 129, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
819	1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 2, 2, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 2, 3, 2, 2, 3, 2, 2, 2, 2, 2, 2, 2, 2, 3, 2, 2, 2, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 2 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
833	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1200, 1208, 1252, 4386, 49121, 4946, 4948, 4951, 49555, 49553, 4960
834	926, 951, 1200, 1208, 1362, 4930, 9026, 13488, 17584
835	927, 947, 1200, 1208, 4931, 9027, 13488, 17584, 21427

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
836	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 424, 437, 500, 737, 775, 819, 833, 850, 852, 855, 857, 870-871, 875, 903, 1009, 1025-1027, 1040-1043, 1088, 1112, 1114-1115, 1122, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4949, 4951, 4953, 4953
837	928, 1200, 1208, 1380, 1385, 4933, 13488, 17584
838	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252, 4909, 4934, 4946, 499048, 4953, 4944, 4960, 4960
848	924, 1148, 1158, 1200, 1208, 5347, 13488, 17584
849	924, 1148, 1154, 1200, 1208, 5347, 13488, 17584
850	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 852, 855-858, 860-866, 869-871, 874-875, 897, 903, 905, 912, 914, 924, 1004, 1025-1027, 880, 880, 903, 916, 49916, 49449925, 49925, 49525, 10925, 10925, 10916, 10960, 10960, 10960, 10, 10960, 10960, 10916, 10916, 10, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 416, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 416, 4, 4, 4, 4, 4, 4, 416, 4, 416, 416, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 416, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
851	259, 423, 500, 875, 1200, 1208, 4971, 13488, 17584
852	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 855, 857, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1043, 1047, 1088, 1097, 49449449490, 4904256, 905, 912
855	37, 259, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 819, 833, 836, 850, 852, 857, 866, 870-871, 878, 880, 912, 915, 1025-1027, 1040-1043, 1088, 1200, 1208, 1250-1252, 1283, 4386, 4929, 4932, 4946, 49121, 4948, 4953, 5123, 5123, 59029, 5347
856	259, 273, 424, 500, 803, 850, 862, 916, 1200, 1208, 1255, 4946, 4952, 5012, 5351, 13488, 17584
857	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1088, 10350, 4997, 1200, 1208, 49552, 1252, 1254
858	37, 259, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 860-861, 865, 871-872, 901-902, 923-924, 1047, 1051, 1140-1149, 1153-1157, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
859	808, 872, 901-902, 1153-1157, 1160-1162, 1164, 1200, 1208, 13488, 17584

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
860	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857-858, 861, 863, 865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 923-924, 1025, 1043, 1097, 1140, 11451146
861	, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857-858, 860, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 923-924, 1025-1027, 1041-1043, 1097, 9049044949, 49497, 49385, 412, 916
862	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 803, 833, 838, 850, 856, 870-871, 875, 880, 905, 916, 1025-1027, 1097, 1200, 1208, 1252, 1255, 4386, 4929, 4934, 4946, 4952, 4971, 5012, 5123, 5351, 8229, 8482,
863	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857, 860-861, 865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1041-1043, 1097, 1140-1149, 1200, 1208, 1252
864	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4960, 4934, 4846, 4971, 5104, 5123, 5123
865	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 819, 833, 838, 850, 858, 860, 863, 870-871, 875, 880, 905, 923-924, 1025-1027, 1097, 1142-1143, 1148, 1200, 1208, 1252, 4386, 4929, 8034, 4946, 4971, 5123, 59029, 5342
866	37, 256, 437, 500, 819, 850, 855, 870, 878, 880, 915, 1025, 1200, 1208, 1251-1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
867	259, 1153-1155, 1160, 1200, 1208, 4899, 5351, 9048, 12712, 13488, 17584
868	918, 1006, 1200, 1208, 13488, 17584
869	37, 256, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 870-871, 874-875, 880, 897, 903, 912, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1254, 1280, 4909, 4934, 4934, 4946, 497048,
870	37, 256, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 855, 857, 860-866, 869, 871, 874-875, 880, 897, 903, 912, 920, 1009, 1025-1027, 1040-1043, 1047, 10494906, 494906, 4980, 915, 915, 915, 916, 920, 920, 960, 960, 960, 922, 960, 960, 960, 960, 960, 960, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 460, 4, 460, 460, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 460, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4-, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
871	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869, 870, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1027, 1040-1043, 1047, 1047, 10905, 490490490, 4904274259.
872	259, 808, 858-859, 923-924, 1140-1149, 1153-1155, 1200, 1208, 5347, 5348, 13488, 17584
874	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4970-498271, 5012, 9090490490, 54123,
875	37, 256, 273, 277-278, 280, 284-285, 297, 367, 423, 437, 500, 737, 775, 813, 819, 836, 838, 850-852, 857, 860-865, 869-871, 874, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1047, 1088, 1112, 1122, 1208, 4990490590, 1280,
878	855, 866, 880, 915, 1025, 1131, 1200, 1208, 1251, 1283, 4951, 5347, 13488, 17584
880	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 855, 857, 860-866, 869-871, 874-875, 878, 897, 903, 912, 915 -916, 920, 1009, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 90341-1252, 1283, 49090, 446
891	500, 833, 1088, 1200, 1208, 4929, 9025, 13121, 13488, 17584, 25664
895	290, 500, 1027, 1041, 1200, 1208, 4386, 5123, 8482, 13488, 17584, 25617
896	290, 1027, 1041, 1200, 1208, 4386, 4992, 5123, 8482, 13488, 17584, 25617
897	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4386, 4909, 494866, 4948, 4953, 4970-4950, 5012
899	259
901	259, 858-859, 902, 923-924, 1140, 1148, 1156-1157, 1200, 1208, 5348, 5353, 13488, 17584
902	259, 858-859, 901, 923-924, 1140, 1148, 1156-1157, 1200, 1208, 5348, 5353, 13488, 17584
903	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 836, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 912, 916, 920, 1025-1027, 1041-1043, 1115, 1200, 1208, 1252, 4909, 4932, 4934, 4946, 4948, 4953, 4912 70-4971,
904	37, 500, 1114, 1200, 1208, 5210, 8229, 13488, 17584, 25480, 28709
905	37, 256, 437, 500, 737, 775, 819, 850, 852, 857, 860-865, 920, 1026, 1112, 1122, 1200, 1208, 1252, 1254, 1281, 4946, 4948, 4953, 4960, 8229, 9044, 9049, 9056, 13488, 17248, 17584, 28709

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
912	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 916, 920, 1025-1027, 1041-1043, 1047, 1200, 1208, 1250, 1252, 1282, 4909, 4934, 4946, 4948, 4951, 9059053, 4953,
914	37, 437, 500, 819, 850, 1200, 1208, 1252, 1257, 4946, 8229, 13488, 17584, 28709
915	37, 259, 437, 500, 819, 850, 855, 866, 870, 878, 880, 1025, 1131, 1200, 1208, 1251-1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
916	37, 273, 277-278, 280, 284-285, 297, 423-424, 437, 500, 803, 813, 819, 838, 850, 852, 856-857, 860-863, 869-871, 874-875, 880, 897, 903, 912, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 1255, 4909, 2534, 4946, 4948, 49905905, 4952-49770-71
918	864, 868, 1006, 1200, 1208, 4960, 9056, 13488, 17248, 17584
920	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 1025-1026, 1200, 1208, 1252, 1254, 1281, 4909, 4934, 4946, 4948, 4753, 4970-498270, 9049029, 5350, 5350,
921	37, 437, 500, 819, 850, 922, 1112, 1122, 1200, 1208, 1252, 1257, 4946, 5353, 8229, 13488, 17584, 28709
922	37, 437, 500, 819, 850, 921, 1112, 1122, 1200, 1208, 1252, 1257, 4946, 5353, 8229, 13488, 17584, 28709
923	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860-861, 865, 871-872, 901-902, 924, 1047, 1051, 1140-1149, 1153-1158, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
924	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 848-850, 858, 860-861, 865, 871-872, 901-902, 923, 1047, 1051, 1140-1149, 1153-1157, 1160-1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
926	834, 951, 9026
927	835, 947, 1200, 1208, 4931, 9027, 13488, 17584, 21427
928	837, 1200, 1208, 1380, 13488, 17584
930	931-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
931	930, 932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
932	930-931, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
933	934, 944, 949, 1200, 1208, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25510, 25520, 25525, 29616, 29621, 33717, 37813

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
934	933, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25510, 25525, 29621, 33717, 37813
935	936, 946, 1200, 1208, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584, 25512
936	935, 946, 1381, 5031, 5477, 5484, 9127, 13223, 25512
937	938, 948, 950, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
938	937, 950, 1370, 5033, 5046, 9142, 25514
939	930-932, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
941	300-301, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
942	930-932, 939, 943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
943	930-932, 939, 942, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
944	933, 949, 1200, 1208, 5029, 5045, 5460, 9125, 13221, 13488, 17317, 17584, 25520, 25525, 29616, 29621, 33717, 37813
946	935-936, 1200, 1208, 5031, 5484, 9127, 13223, 13488, 17584, 25512
947	835, 927, 1200, 1208, 4931, 9027, 13488, 17584, 21427
948	937, 950, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25524, 29620
949	933-934, 944, 1200, 1208, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25510, 25520, 25525, 29616, 29621, 33717, 37813
950	937-938, 948, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
951	834, 926, 1200, 1208, 1362, 4930, 9026, 13488, 17584
1004	500, 819, 850, 1200, 1208, 4946, 13488, 17584
1006	868, 918, 1200, 1208, 13488, 17584
1008	420, 864, 1200, 1208, 4960, 5104, 8612, 9056, 13488, 16804, 17248, 17584
1009	37, 273, 277-278, 280, 284, 290, 297, 367, 423, 500, 833, 836, 870-871, 875, 880, 1025-1026, 1200, 1208, 4386, 4929, 4932, 4971, 8229, 8482, 9025, 13121, 13488, 17584, 28709
1010	500, 1200, 1208, 13488, 17584
1011	500, 1200, 1208, 13488, 17584
1012	500, 1200, 1208, 13488, 17584

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
1013	500, 1140, 1200, 1208, 13488, 17584
1014	500, 1200, 1208, 13488, 17584
1015	500, 1200, 1208, 13488, 17584
1016	500, 1200, 1208, 13488, 17584
1017	500, 1200, 1208, 13488, 17584
1018	500, 1200, 1208, 13488, 17584
1019	500, 1200, 1208, 13488, 17584
1020	500
1021	500
1023	500
1025	737, 737, 775, 813, 819, 836, 838, 850, 852, 855, 857, 860, 875, 878, 880, 897, 903, 915 -916, 920, 1009, 1026-1027, 10491043, 1051, 102449449490, 4944944949, 4944949, 4944949., 49449449449, 44944949., 4909., 4259.
1026	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-865, 869-875, 880, 897, 903, 905, 912, 916, 920, 1009, 1025, 1027, 10491043, 1047, 1088, 1111, 490, 490, 4276
1027	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-865, 869-875, 880, 895-897, 903, 912, 916, 1025-1043, 1047, 104988, 1112, 90449490, 49490, 490490, 4129200
1040	37, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 833, 836, 850, 852, 855, 857, 870-871, 1025-1027, 1041-1043, 1088, 1200, 1208, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25617, 25619
1041	37, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1027, 1040, 1042-1043, 1088, 1200, 1208, 1252, 4386, 49029
1042	, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1041, 1043, 1088, 1208, 4386, 4909, 4929, 494934
1043	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 1025-1027, 1040, 1041, 1042, 1088, 1114, 1200, 1208, 4386, 4904909, 4929, 4946
1046	420, 500, 864, 1089, 1127, 1200, 1208, 1256, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
1047	37, 273-275, 277-278, 280, 281, 282, 284-285, 290, 297, 437, 500, 819, 850, 852, 858, 870-871, 875, 912, 923-924, 1026-1027, 1140-1149, 1200, 1208, 1252, 1254, 4946, 4948, 5123, 8229, 8482, 13488, 17584, 28709
1051	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871, 923-924, 1025, 1097, 1140-1149, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1088	37, 273, 277-278, 280, 284-285, 290, 297, 367, 500, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 891, 1025-1027, 1040-1043, 1126, 1200, 1208, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 13464
1089	420, 500, 819, 850, 864, 1046, 1127, 1200, 1208, 1256, 4946, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1097	37, 437, 500, 737, 775, 819, 850, 852, 857, 860-865, 1051, 1098, 1112, 1122, 1200, 1208, 1252, 4946, 4948, 4953, 4960, 8229, 9044, 9049, 9056, 13488, 17248, 17584, 28709
1098	259, 420, 437, 819, 850, 1097, 1200, 1208, 1252, 4946, 8612, 13488, 16804, 17584
1100	37, 273, 277-278, 280, 284-285, 297, 500, 850, 4946, 8229, 28709
1101	500
1102	500
1103	500
1104	500
1105	500
1106	500
1107	500
1112	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 775, 819, 833, 836, 838, 850, 870-871, 875, 880, 905, 921-922, 1025-1027, 1097, 1122, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 5353, 8229, 848862, 12
1114	37, 437, 500, 819, 836, 850, 904, 1043, 1115, 1200, 1208, 4932, 4946, 5210-5211, 8229, 13488, 17584, 25480, 25619, 28709
1115	37, 367, 437, 500, 836, 903, 1114, 1200, 1208, 4932, 5210-5211, 8229, 13488, 17584, 25479, 28709
1122	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 775, 819, 833, 836, 838, 850, 870-871, 875, 880, 905, 921-922, 1025-1027, 1097, 1112, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 5353, 8229, 848862
1123	819, 1124-1125, 1148, 1200, 1208, 1251-1252, 1283, 5347, 13488, 17584
1124	37, 500, 1123, 1125, 1200, 1208, 1251, 1283, 5347, 8229, 13488, 17584, 28709

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
1125	500, 1123, 1124, 1200, 1208, 1251, 1283, 5347, 13488, 17584
1126	37, 367, 437, 500, 819, 833, 850, 1088, 1200, 1208, 1252, 4929, 4946, 8229, 9025, 13121, 13488, 17584, 25664, 28709
1127	420, 864, 1046, 1089, 1256, 4960, 5142, 8612, 9056, 9238, 16804, 17248
1129	500, 1130, 1200, 1208, 1258, 5354, 13488, 17584
1130	37, 500, 819, 850, 1129, 1200, 1208, 1252, 1258, 4946, 5354, 8229, 13488, 17584, 28709
1131	37, 500, 878, 915, 1025, 1200, 1208, 1251, 1283, 5347, 8229, 13488, 17584, 28709
1132	37, 500, 819, 850, 1133, 1200, 1208, 1252, 4946, 8229, 13488, 17584, 28709
1133	500, 1132, 1200, 1208, 13488, 17584
1137	37, 500, 819, 1200, 1208, 8229, 13488, 17584, 28709
1139	290, 1027, 4386, 5123, 8482
1140	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860, 863, 871-872, 901-902, 923-924, 1013, 1047, 1051, 1141-1149, 1153-1157, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1141	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140, 1142-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1142	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 865, 871-872, 923-924, 1047, 1051, 1140-1141, 1143-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1143	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 865, 871-872, 923-924, 1047, 1051, 1140-1142, 1144-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1144	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140-1143, 1145-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1145	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 860, 863, 871-872, 923-924, 1047, 1051, 1140-1144, 1146-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1146	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 860, 863, 871-872, 923-924, 1047, 1051, 1140-1145, 1147-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
1147	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140-1146, 1148-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1148	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 848-850, 858, 860-861, 863, 865, 871-872, 901-902, 923-924, 1047, 1051, 1123, 1140-1147, 1149, 1153-1164, 1200, 1208, 1252, 1275, 4899, 4946, 5348, 5349, 8229, 12712, 13488, 17584
1149	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 861, 863, 871-872, 923-924, 1047, 1051, 1140-1148, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1153	808, 858-859, 867, 872, 923-924, 1140-1149, 1154-1157, 1160-1162, 1200, 1208, 5348, 9044, 13488, 17584
1154	808, 849, 858-859, 867, 872, 923-924, 1140-1149, 1153, 1155-1157, 1160-1162, 1200, 1208, 5347, 5348, 13488, 17584
1155	858-859, 867, 872, 923-924, 1140-1149, 1153-1154, 1156-1157, 1160-1162, 1200, 1208, 5348, 5350, 9049, 13488, 17584
1156	858-859, 901-902, 923-924, 1140-1149, 1153-1155, 1157, 1160, 1200, 1208, 5348, 5353, 12712, 13488, 17584
1157	858-859, 901-902, 923-924, 1140-1149, 1153-1156, 1160, 1200, 1208, 5348, 5353, 12712, 13488, 17584
1158	848, 923, 1148, 1200, 1208, 5347, 5348, 13488, 17584
1159	1148, 1200, 1208, 13488, 17584
1160	858-859, 867, 923-924, 1140-1149, 1153-1157, 1161-1162, 1200, 1208, 5348, 13488, 17584
1161	259, 858-859, 923-924, 1140-1149, 1153-1155, 1160, 5348, 17584
1162	259, 858-859, 923-924, 1140-1149, 1153-1155, 1160, 5348, 17584
1163	924, 1148, 1164, 5354, 17584
1164	858-859, 923-924, 1140, 1148, 1163, 1200, 1208, 5348, 5354, 13488, 17584
1166	1200,1208,13488,17584
1200	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 891, 895-897, 901-905, 912, 915, 915, 916, 915, 925, 925, 925, 925 1362-1364, 1370-1371, 1374-1379, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
1208	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 90925, 918, 915, 916 1362-1364, 1370-1371, 1374-1379, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5026, 5035, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9012, 9056, 904
1250	37, 259, 273, 500, 819, 850, 852, 855, 870, 912, 1200, 1208, 1252, 1282, 4946, 4948, 4951, 5346, 8229, 9044, 13488, 17584, 28709
1251	37, 256, 259, 500, 819, 850, 855, 866, 878, 880, 915, 1025, 1123-1125, 1131, 1200, 1208, 1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
1252	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 737, 775, 803, 813, 819, 833, 836, 838, 850, 852, 855, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 920-924, 104925, 1025-10272, 925, 412690, 412690, 490, 490, 4129490, 490, 41298960, 412960, 412960, 412960, 412960, 412960, 412, 4, 412960, 4129, 412, 4, 4, 4, 4, 4, 4129, 4,,,, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 28709
1253	37, 259, 423, 500, 737, 813, 819, 850, 869, 875, 1200, 1208, 1280, 4909, 4946, 4971, 5349, 8229, 9061, 13488, 17584, 28709
1254	37, 259, 500, 819, 850, 857, 869, 905, 920, 1026, 1047, 1200, 1208, 1252, 1281, 4946, 4953, 5350, 8229, 9049, 9061, 13488, 17584, 28709
1255	37, 259, 424, 500, 803, 819, 850, 856, 862, 916, 1200, 1208, 1252, 1281, 4946, 4952, 5012, 5351, 8229, 13488, 17584, 28709
1256	259, 420, 500, 720, 850, 864, 1046, 1089, 1127, 1200, 1208, 4946, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1257	37, 259, 437, 500, 775, 819, 850, 914, 921-922, 1112, 1122, 1200, 1208, 1252, 4946, 5353, 8229, 13488, 17584, 28709
1258	37, 259, 500, 819, 1129-1130, 1200, 1208, 5354, 8229, 13488, 17584, 28709
1275	37, 256, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871, 923-924, 1051, 1140-1149, 1200, 1208, 1252, 4946, 5348, 8229, 13488, 17584, 28709
1276	1200, 1208, 13488, 17584
1277	1200, 1208, 13488, 17584
1280	37, 423, 437, 500, 737, 813, 819, 850, 869, 875, 1200, 1208, 1252-1253, 4909, 4946, 4971, 5349, 8229, 9061, 13488, 17584, 28709
1281	37, 437, 500, 819, 850, 857, 905, 920, 1026, 1200, 1208, 1252, 1254-1255, 4946, 4953, 5350, 8229, 9049, 13488, 17584, 28709

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
1282	500, 852, 870, 912, 1200, 1208, 1250, 4948, 5346, 9044, 13488, 17584
1283	37, 437, 500, 819, 850, 855, 866, 878, 880, 915, 1025, 1123-1125, 1131, 1200, 1208, 1251-1252, 4946, 4951, 5347, 8229, 13488, 17584, 28709
1284	1200, 1208, 13488, 17584
1285	1200, 1208, 13488, 17584
1351	300-301, 941, 1200, 1208, 4396, 8492, 13488, 16684, 17584
1362	834, 951, 1200, 1208, 4930, 9026, 13488, 17584
1363	933, 949, 1200, 1208, 1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25525, 29621, 33717, 37813
1364	933, 949, 1200, 1208, 1363, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25525, 29621, 33717, 37813
1370	937-938, 948, 950, 1200, 1208, 1371, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
1371	1200, 1208, 1370, 13488, 17584
1374	1200, 1208
1375	1200, 1208
1376	1200, 1208
1377	1200, 1208
1378	1200, 1208
1379	1200, 1208
1380	837, 928, 1200, 1208, 1385, 4933, 13488, 17584
1381	935-936, 1200, 1208, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584, 25512
1385	837, 1200, 1208, 1380, 4933, 13488, 17584
1386	935, 1200, 1208, 1381, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584
1388	935, 1200, 1208, 1381, 1386, 5031, 5477, 5482, 5484, 5488, 9127, 13223, 13488, 17584
1390	930-932, 939, 942-943, 1200, 1208, 1399, 5026, 5028, 5035, 5038-5039, 5055, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
1399	930-932, 939, 942-943, 1200, 1208, 1390, 5026, 5028, 5035, 5038-5039, 5050, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
4386	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1139, 1252, 4929, 49121, 4948, 4951, 4953, 4953, 4960, 4992
4396	300-301, 941, 1351, 8492, 16684

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
4899	867, 1148, 1200, 1208, 5351, 9048, 12712, 13488, 17584
4909	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1253, 1280, 4934, 4946, 4948, 493453, 4970770-4971, 5012
4929	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1252, 4386, 4932, 4946, 4948, 4951, 499053, 4968260, 24906, 590, 29
4930	834, 951, 1200, 1208, 1362, 9026, 13488, 17584
4931	835, 927, 947, 9027, 21427
4932	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 424, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 903, 1009, 1025-1027, 1040-1043, 1088, 1112, 1114-1115, 1122, 1252, 4386, 4829, 4946, 4948, 4951, 4953, 10-71,
4933	837, 1200, 1208, 1380, 1385, 13488, 17584
4934	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1252, 4909, 4948, 4953, 499060, 49660, 5070-4971,
4946	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 850, 890908, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 920-924, 9904494, 1025-1027, 6284284, 428928928, 490, 490, 490, 490, 490, 490, 490, 490, 490, 490, 490, 490, 490, 490, 490, 490, 4606, 490, 460, 460, 460, 4606, 4606, 4, 4, 4, 460, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,, 7, 7,, 7, 7, 7, 7, 7,,,,,,,, 7, 7,,,,,,,,,, 7,,, 7, 7,,,,,,,,,,,,,,,,,,,,,,,, 7, 7, 7, 7,,,,,,,,,,,,,,,,, 9066, 13121, 16804, 17248, 25473, 25479, 25617, 25619, 25664, 28709
4948	, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-875, 880, 897, 903, 912, 925, 1040-1043, 1047, 104990449449, 49449449, 875, 897, 903, 905, 912, 416, 920, 1025-1043, 1047, 10494906, 4494497606, 449875, 897, 90960
4951	37, 259, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 819, 833, 836, 850, 852, 855, 857, 866, 870-871, 878, 880, 912, 915, 1025-1027, 1040-1043, 1088, 1200, 1208, 1250-1252, 1283, 4386, 4929, 4932, 49121, 4846, 4948, 4953, 5123, 59044, 5346, 5347
4952	259, 273, 424, 500, 803, 850, 856, 862, 916, 1200, 1208, 1255, 4946, 5012, 5351, 13488, 17584
4953	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-104843, 1088, 1097, 4990449525, 49449575, 1254,

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
4960	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4988634, 4946, 4971, 5104, 5123, 5123, 552, 552, 552, 552, 4142,
4970	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1252, 4909, 4934, 4946, 4948, 4953, 4771, 509012, 5123, 8290, 9029, 9030
4971	37, 256, 273, 277-278, 280, 284-285, 297, 367, 423, 437, 500, 737, 775, 813, 819, 836, 838, 850-852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1047, 1088, 1112, 499049049, 49044954953, 44970, 9602, 960, 925, 916, 910, 910, 1025, 1027, 1027, 1025, 1025, 1025, 1025, 1025, 1043, 1047, 1043, 1043, 1043, 1043, 1043, 1043, 1043, 1043, 443, 1043, 443, 443, 443, 443, 443, 443, 443, 443, 443, 443, 443, 443, 443, 443, 443, 443, 4, 4, 4, 4, 4, 447, 4, 443, 443, 443, 443, 4, 443, 443, 4, 4, 4, 4, 443, 4, 4, 443, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4
4992	290, 896, 1027, 1041, 4386, 5123, 8482, 25617
5012	287, 287, 285, 297, 423-424, 437, 500, 803, 813, 819, 838, 850, 852, 856-857, 860-863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 1255, 4909, 4934, 4946, 4948, 49590, 4952-4953, 49790
5026	930-932, 939, 942-943, 1390, 1399, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 1208, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5028	930-932, 939, 942-943, 1390, 1399, 5026, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5029	933-934, 944, 949, 1363-1364, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5031	935-936, 946, 1381, 1386, 1388, 5477, 5482, 5484, 9127, 13223, 25512
5033	937-938, 948, 950, 1370, 5046, 9142, 25514, 25524, 29620
5035	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5038-5039, 9122, 9124, 9131, 9135, 1208, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5038	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5039	930-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
5045	933-934, 944, 949, 1363-1364, 5029, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5046	937-938, 948, 950, 1370, 5033, 9142, 25514, 25524, 29620

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
5104	420, 864, 1008, 1200, 1208, 4960, 8612, 9056, 13488, 16804, 17248, 17584
5123	290, 367, 423, 437, 819, 1027, 1041, 1047, 1140-1149, 1156, 1157, 1160, 1200, 1208, 1252, 4948, 5348, 8482, 13488
5142	420, 500, 864, 1046, 1089, 1127, 1200, 1208, 1256, 4960, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
5210	37, 437, 500, 819, 836, 850, 904, 1043, 1114-1115, 1200, 1208, 4932, 4946, 5211, 8229, 13488, 17584, 25480, 25619, 28709
5211	37, 367, 437, 500, 836, 903, 1114-1115, 4932, 5210, 8229, 25479, 28709
5346	37, 259, 273, 500, 819, 850, 852, 855, 870, 912, 1200, 1208, 1250, 1252, 1282, 4946, 4948, 4951, 8229, 9044, 13488, 17584, 28709
5347	808, 848-849, 855, 866, 872, 878, 880, 915, 1025, 1123-1125, 1131, 1154, 1158, 1200, 1208, 1251, 1283, 4951, 13488, 17584
5348	37, 259, 273, 275, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860-861, 863, 865, 871-872, 901-902, 923-924, 1051, 1140-1149, 1153-1158, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 8229, 13488, 17584, 28709
5349	813, 869, 875, 1148, 1200, 1208, 1253, 1280, 4909, 4971, 9061, 13488, 17584
5350	857, 920, 1026, 1155, 1200, 1208, 1254, 1281, 4953, 9049, 13488, 17584
5351	424, 856, 862, 867, 916, 1200, 1208, 1255, 4899, 4952, 5012, 9048, 12712, 13488, 17584
5352	420, 864, 1046, 1089, 1200, 1208, 1256, 4960, 5142, 8612, 9056, 9238, 13488, 16804, 17248, 17584
5353	901-902, 921-922, 1112, 1122, 1156-1157, 1200, 1208, 1257, 13488, 17584
5354	1129-1130, 1163, 1164, 1200, 1208, 1258, 13488, 17584
5460	933-934, 944, 949, 1363-1364, 5029, 5045, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5477	935-936, 1381, 1386, 1388, 5031, 5482, 5484, 9127, 13223, 25512
5482	935, 1381, 1386, 1388, 5031, 5477, 5484, 9127, 13223
5484	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 9127, 13223, 25512
5488	1388

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
8229	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 775, 813, 819, 833, 836, 838, 852, 855, 857-857-858, 89-871, 874-875, 880, 897, 903-905, 912, 914, 920-924, 1009, 1025, 1025, 447, 875, 880, 897, 897, 897, 897, 897, 912, 912, 912, 912, 912, 912, 414, 418418418, 418, 418, 418, 418, 418, 418, 418, 418, 418, 418, 418, 418, 418, 9, 418, 418, 418, 418, 9, 4, 4, 418, 6, 4, 4, 4, 4, 4, 4, 4, 418, 6, 4, 4, 4, 4, 418, 4, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 8, 4, 4, 87, 4, 4, 87, 8,, 8, 8, 9044, 9049, 9056, 9061, 9066, 13121, 16804, 17248, 25473, 25479, 25480, 25617, 25619, 25664, 28709
8482	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4946, 4948
8492	300-301, 941, 1351, 4396, 16684
8612	37, 256, 420, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 9044, 9049, 9056, 9238, 13488, 16804, 17248, 17584,
9025	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1252, 4386, 4929, 4832, 4946, 4948, 4851, 4982249019, 4960, 5123, 5123
9026	834, 926, 951, 1362, 4930
9027	835, 927, 947, 1200, 1208, 4931, 13488, 17584, 21427
9030	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 124752, 4909, 4934, 499046, 4948, 4957053, 4953
9044	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1043, 1047, 109044952, 1097, 1153,
9048	867, 1200, 1208, 4899, 5351, 12712, 13488, 17584
9049	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1043, 1088, 1097, 49549590, 4904905, 1208
9056	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 494829, 4946, 499060, 4971, 5104, 5123, 5123, 552

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
9061	37, 256, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-43, 1200, 1208, 1252-1254, 1280, 4909, 4934, 4946, 4948,
9066	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 499053, 4970-4930, 5012
9122	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9124	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9125	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
9127	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 13223, 25512
9131	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9135	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9142	937-938, 948, 950, 1370, 5033, 5046, 25514, 25524, 29620
9238	420, 500, 864, 1046, 1089, 1127, 1200, 1208, 1256, 4960, 5142, 5352, 8612, 9056, 13488, 16804, 17248, 17584
9555	933, 949, 1363-1364, 5029, 5045, 5460, 9125, 13221, 13651, 17317, 25525, 29621, 33717, 37813
12712	862, 867, 1148, 1156-1157, 1200, 1208, 4899, 5351, 9048, 13488, 17584
13121	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1200, 1208, 1252, 4386, 4829, 4932, 4946, 4949494949, 560, 4951, 4953
13218	930-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
13219	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
13221	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
13223	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 25512

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
13231	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 17314, 25508, 25518, 29614, 33698-33700, 37796
13488	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 891, 895-810097, 901-905, 912, 912, 916, 916 1362-1364, 1370-1371, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 1274, 1374, 13712
13651	933, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 17317, 25525, 29621, 33717, 37813
16684	300-301, 941, 1200, 1208, 1351, 4396, 8492, 13488, 17584
16804	37, 256, 420, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 8612, 9044, 9049, 9056, 9238, 13488, 17248, 17584, 28709
17248	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4946, 499060, 4971, 5104, 123, 5123, 552
17314	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 25508, 25518, 29614, 33698-33700, 37796
17317	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 25510, 25520, 25525, 29616, 29621, 33717, 37813
17584	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 891, 895-810097, 901-905, 912, 912, 916, 916 1362-1364, 1370-1371, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 1274, 13427, 134121
21427	835, 927, 947, 1200, 1208, 4931, 9027, 13488, 17584
25473	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1027, 1041-1043, 1252, 4386, 4909, 4934, 494846, 4948, 4953, 4970-499071, 5012 44, 544, 544,
25479	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 836, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1115, 1252, 4909, 4932, 4934, 4946, 4948, 499053, 4970-59030, 5012
25480	37, 500, 904, 1114, 5210, 8229, 28709
25508	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25518, 29614, 33698-33700, 37796

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
25510	933-934, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25525, 29621, 33717, 37813
25512	935-936, 946, 1381, 5031, 5477, 5484, 9127, 13223
25514	937-938, 950, 1370, 5033, 5046, 9142
25518	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 29614, 33698-33700, 37796
25520	933, 944, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25525, 29616, 29621, 33717, 37813
25524	937, 948, 950, 1370, 5033, 5046, 9142, 29620
25525	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 29616, 29621, 33717, 37813
25617	37, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1027, 1040-1043, 1088, 1252, 4386, 4909, 4949, 4946, 4934,
25619	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 1025-1027, 1043, 1088, 1114, 4386, 4909, 4929, 4932, 4934, 4946, 4946
25664	37, 273, 277-278, 280, 284-285, 290, 297, 367, 500, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 891, 1025-1027, 1040-1043, 1088, 1126, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 25617,
28709	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 920-924, 1009, 1050-1027, 880, 880, 897, 912, 914, 414, 49514, 499282, 4282, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 4284, 422, 4284, 4284, 422, 428, 8, 428, 428, 428, 428, 422, 428, 428, 428, 422, 428, 422, 422, 422, 422, 422, 8, 422, 8, 8, 8, 422, 8, 8, 8, 8, 8, 422, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 7, 8, 8, 8, 8, 8, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664
29614	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 33698-33700, 37796
29616	933, 944, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25520, 25525, 29621, 33717, 37813
29620	937, 948, 950, 1370, 5033, 5046, 9142, 25524
29621	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 33717, 37813

Tabulka 674. IBM MQ for z/OS Podpora převodu CCSID (pokračování)	
CCSID	**Převádí do a z CCSID**
33698	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33699-33700, 37796
33699	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698, 33700, 37796
33700	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33699, 37796
33717	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 37813
37796	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700
37813	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717

IBM i IBM i podpora převodu

Úplný seznam CCSID a převodů podporovaných produktem IBM inaleznete v příslušné publikaci IBM i .

Podporované kódové stránky jsou uvedeny v části Podporovaná mapování CCSID.

Podpora převodu Unicode

Některé platformy podporují převod uživatelských dat do nebo z kódování Unicode. Podporovány jsou dvě formy kódování Unicode: UTF-16 (CCSID 1200, 13488 a 17584) a UTF-8 (CCSID 1208). Měli byste používat CCSID 1200 nebo 1208, protože představují nejnovější podporovanou verzi Unicode.

UTF-16 náhradní dvojice (dvojice dvoubajtových znaků UTF-16 v rozsahu od X'D800'do X'DFFF', které představují bod kódu Unicode nad U + FFFF) jsou podporovány. Pokud cílový CCSID neobsahuje mapování pro bod kódu reprezentovaný náhradním párem UTF-16 , dvojice znaků se převede na jeden zástupný znak.

Kombinace znakových posloupností je podporována produktem IBM MQ. To znamená, že v některých případech bude předsložený znak ve zdrojovém CCSID převeden na kombinující posloupnost znaků v cílovém CCSID, nebo naopak.

Poznámka: Produkt IBM MQ nepodporuje CCSID správce front UTF-16 , takže data záhlaví zpráv nelze zakódovat do UTF-16.

IBM MQ AIX podpora kódování Unicode

AIX

Při převodu produktu IBM MQ for AIX na podporované identifikátory CCSID Unicode a z nich (nejlépe 1200 nebo 1208) jsou podporovány pro identifikátory CCSID jiné než Unicode v následujícím seznamu:

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, 1153, 1156, 1157
1200, 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

IBM MQ for Windows a Linux podpora kódování Unicode

Windows Linux

V systémech IBM MQ for Windows a IBM MQ pro převod Linux na podporované CCSID Unicode a z nich (nejlépe 1200 nebo 1208) jsou podporovány pro CCSID jiné než Unicode v následujícím seznamu:

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" na stránce 986, 915, 916, 918, 920, 921, 922, 923, 924, 927, 928, 930, 931"1" na stránce 986, 932"2" na stránce 986, 933, 935, 937, 938"3" na stránce 986, 939, 941, 942, 943, 947, 948, 949, 950, 951, 954"4" na stránce 986, 964, 970
1006, 1025, 1026, 1027, 1040, 1041, 1042, 1043, 1046, 1047, 1051, 1088, 1089, 1097, 1098
1112, 1114, 1115, 1122, 1123, 1124, 1129, 1130, 1132, 1133, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1153, 1156, 1157
1200, 1208, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1275, 1280, 1281, 1282, 1283
1363, 1364, 1374, 1375, 1376, 1377, 1378, 1379, 1380, 1381, 1383, 1386, 1388
4899
5050, 5346, 5347, 5348, 5349, 5350, 5351, 5352, 5353, 5354, 5488"5" na stránce 986
9044, 9048, 9449
12712
13488
17584
33722"4" na stránce 986

Notes:

1. 931 používá pro převod 939.
2. 932 používá pro převod 942.
3. 938 používá pro převod 948.
4. 954 a 33722 používají pro převod 5050.
5. Pouze v systémech Windows a Linux .

IBM i podpora pro Unicode

IBM i

Podrobnosti o podpoře UNICODE naleznete v příslušné publikaci IBM i týkající se vašeho operačního systému.

IBM MQ for z/OS podpora pro Unicode

z/OS

Při převodu produktu IBM MQ for z/OS na podporované identifikátory CCSID Unicode a z nich (nejlépe 1200 nebo 1208) jsou podporovány pro identifikátory CCSID jiné než Unicode v následujícím seznamu:

37
256, 259, 273, 275, 277, 278, 280, 282, 284, 285, 290, 293, 297
300, 301, 367
420, 423, 424, 437
500
720, 737, 775
803, 806, 808, 813, 819, 833, 834, 835, 836, 837, 838, 848, 849, 850, 851, 852, 855, 856, 857, 858, 859, 860, 861, 862, 863, 864, 865, 866, 867, 868, 869, 870, 871, 872, 874, 875, 878, 880, 891, 895, 896, 897
901, 902, 903, 904, 905, 912, 914, 915, 916, 918, 920, 921, 922, 923, 924, 927, 928, 930, 932, 933, 935, 937, 939, 941, 942, 943, 944, 946, 947, 948, 949, 950, 951
1004, 1006, 1008, 1009, 1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017, 1018, 1019, 1025, 1026, 1027, 1040, 1041, 1042, 1043, 1046, 1047, 1051, 1088, 1089, 1097, 1098
1112, 1114, 1115, 1122, 1123, 1124, 1125, 1126, 1129, 1130, 1131, 1132, 1133, 1137, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1153, 1154, 1155, 1156, 1157, 1158, 1159, 1160, 1161, 1162, 1164
1200, 1208, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1275, 1276, 1277, 1280, 1281, 1282, 1283, 1284, 1285
1351, 1362, 1363, 1364, 1370, 1371, 1380, 1381, 1385, 1386, 1388, 1390, 1399
4899, 4909, 4930, 4933, 4948, 4951, 4952, 4960, 4971
5012 5039 5104 5123 5142 5210 5346 5347 5348 5349 5350 5351 5352 5353 5354 5488
8482 8612
9027 9030 9044 9048 9049 9056 9061 9066 9238 9449
1166
12712
13121, 13218, 13488, 1374, 1375, 1376, 1377, 1378, 1379
16684, 16804
17248, 17584
21427
28709

Standardy kódování na 64bitových platformách

Pomocí těchto informací získáte informace o standardech kódování na 64bitových platformách a preferovaných datových typech.

Upřednostňované datové typy

Tyto typy nikdy nemění velikost a jsou k dispozici na 32bitových i 64bitových platformách IBM MQ :

Tabulka 675. Názvy a délky datových typů

Název	Délka
MQLONG	4 bajty
MQULONG (NEPOUŽITELNÝ)	4 bajty
MQINT32	4 bajty
MQUINT32	4 bajty
MQINT64	8 bajtů
MQUINT64	8 bajtů

ALW Standardní datové typy na systému AIX, Linux, and Windows

Získejte informace o standardních datových typech v 32bitových aplikacích AIX and Linuxa 64bitových aplikacích AIX, Linux, and Windows .

32bitové aplikace AIX and Linux

Linux AIX

Tabulka 676. Názvy a délky datových typů pro 32bitové aplikace AIX and Linux

Název	Délka
ZNAK	1 bajt
short	2 bajty
celé číslo	4 bajty
long	4 bajty
float	4 bajty
dvojitý	8 bajtů
Dlouhý dvojitý	8 bajtů
Ukazatel	4 bajty
ptrdiff_t	4 bajty
velikost_t	4 bajty
čas_t	4 bajty
hodiny_t	4 bajty
wchar_t	4 bajty AIX Všimněte si, že v systému AIX je hodnota wchar_t 2 bajty.

64bitové aplikace AIX and Linux

Linux AIX

Tabulka 677. Názvy a délky datových typů pro 64bitové aplikace AIX and Linux

Název	Délka
ZNAK	1 bajt
short	2 bajty

Tabulka 677. Názvy a délky datových typů pro 64bitové aplikace AIX and Linux (pokračování)

Název	Délka
celé číslo	4 bajty
long	8 bajtů
float	4 bajty
dvojitý	8 bajtů
Dlouhý dvojitý	8 bajtů
Ukazatel	8 bajtů
ptrdiff_t	8 bajtů
velikost_t	8 bajtů
čas_t	8 bajtů
hodiny_t	4 bajty
wchar_t	4 bajty
	AIX Všimněte si, že v systému AIX je hodnota wchar_t 2 bajty.

Windows 64bitové aplikace

Windows

Tabulka 678. Názvy a délky datových typů pro 64bitové aplikace Windows

Název	Délka
ZNAK	1 bajt
short	2 bajty
celé číslo	4 bajty
long	4 bajty
float	4 bajty
dvojitý	8 bajtů
Dlouhý dvojitý	8 bajtů
Ukazatel	8 bajtů
	Všimněte si, že všechny ukazatele jsou 8 bajtů.
ptrdiff_t	8 bajtů
velikost_t	8 bajtů
čas_t	8 bajtů
hodiny_t	4 bajty
wchar_t	2 bajty
Word	2 bajty
DWORD	4 bajty
aplikace	8 bajtů
SOUBOR	4 bajty

Aspekty kódování na webu Windows

Windows

HANDLE hf;

Použijte

```
hf = CreateFile((LPCTSTR) FileName,
                Access,
                ShareMode,
                xihSecAttsNTRestrict,
                Create,
                AttrAndFlags,
                NULL);
```

Nepoužívat

```
HFILE hf;
hf = (HFILE) CreateFile((LPCTSTR) FileName,
                        Access,
                        ShareMode,
                        xihSecAttsNTRestrict,
                        Create,
                        AttrAndFlags,
                        NULL);
```

protože to způsobí chybu.

size_t len fgets

Použijte

```
size_t len
while (fgets(string1, (int) len, fp) != NULL)
len = strlen(buffer);
```

Nepoužívat

```
int len;

while (fgets(string1, len, fp) != NULL)
len = strlen(buffer);
```

printf

Použijte

```
printf("My struc pointer: %p", pMyStruc);
```

Nepoužívat

```
printf("My struc pointer: %x", pMyStruc);
```

Pokud potřebujete hexadecimální výstup, musíte vytisknout horní a dolní 4 bajty odděleně.

char * ptr

Použijte

```
char * ptr1;
char * ptr2;
size_t bufLen;

bufLen = ptr2 - ptr1;
```

Nepoužívat

```
char *ptr1;
char *ptr2;
UINT32 bufLen;

bufLen = ptr2 - ptr1;
```

alignBytes

Použijte

```
alignBytes = (unsigned short) ((size_t) address % 16);
```

Nepoužívat

```
void *address;
unsigned short alignBytes;

alignBytes = (unsigned short) ((UINT32) address % 16);
```

DÉLKA

Použijte

```
len = (UINT32) ((char *) address2 - (char *) address1);
```

Nepoužívat

```
void *address1;
void *address2;
UINT32 len;

len = (UINT32) ((char *) address2 - (char *) address1);
```

sscanf (skenování)

Použijte

```
MQLONG SBCSprt;

sscanf(line, "%d", &SBCSprt);
```

Nepoužívat

```
MQLONG SBCSprt;

sscanf(line, "%1d", &SBCSprt);
```

Produkt `%ld` se pokusí vložit 8bajtový typ do 4bajtového typu. Použijte hodnotu `%l` pouze v případě, že se jedná o skutečný datový typ `long` . MQLONG, UINT32 a INT32 jsou definovány jako čtyři bajty, stejné jako `int` na všech IBM MQ platformách:

IBM i IBM i Application Programming Reference (ILE/RPG) (Referenční příručka pro programování aplikací)

Programování aplikací pro IBM i.

Tyto informace vám pomohou při vývoji aplikací pro systém IBM i.

- “Popisy datových typů na IBM i” na stránce 993

- “Volání funkcí na systému IBM i” na stránce 1244
- “Atributy objektů v systému IBM i” na stránce 1360
- “Aplikace” na stránce 1405
- “Návratové kódy pro IBM i (ILE RPG)” na stránce 1418
- “Pravidla pro ověření voleb MQI pro IBM i (ILE RPG)” na stránce 1419
- “Kódování počítačů na IBM i” na stránce 1422
- “Volby sestavy a příznaky zpráv na IBM i” na stránce 1425

Zamítnutí režimu kompatibility pro aplikace v jazycích RPG a COBOL v systému IBM i

Od produktu IBM MQ for IBM i 9.0již produkt neposkytuje podporu pro aplikace v jazycích RPG nebo COBOL, které používají dynamické propojení známé jako režim kompatibility. Tento provozní režim byl potřebný pro aplikace, které byly napsány před produktem MQSeries 5.1, a následné verze produktu poskytovaly kompatibilní běhové prostředí pro tyto aplikace, i když zakladače potřebné pro jejich kompilaci byly odebrány v produktu IBM WebSphere MQ 6.0. Dynamické sestavení (režim kompatibility) bylo poskytnuto následujícími programy v knihovně QMQM, které byly odebrány v adresáři IBM MQ for IBM i 9.0:

- AMQVSTUB
- AMQZSTUB
- QMQM
- MQCLOSE
- MQCONN
- MQDISC
- MQGET
- MQINQ
- MQOPEN
- MQPUT
- MQPUT1
- MQSET

V produktu IBM MQ for IBM i 9.0je třeba aplikace používající tento režim kompatibility znovu zkompilovat, aby používaly statická vázaná volání produktu MQ poskytovaná servisními programy LIBMQM a LIBMQM_R. Ukázkové programy, například AMQ3PUT4 a AMQ3GET4, ukazují, jak používat tento programovací model. Další informace o použití těchto volání MQ naleznete v příručce IBM i Application Programming Reference (ILE/RPG).

Notes:

- Chcete-li místo toho použít servisní program LIBMQM, musíte znovu kódovat aplikace, které aktuálně používají rozhraní CALL 'QMQM'.

  Programové objekty a servisní programy v předchozím seznamu, například QMQM, MQCONN, MQPUT, AMQVSTUB a AMQZSTUB, jsou odebrány v produktu IBM MQ for IBM i 9.0a aplikace, které byly kódovány, aby používaly režim kompatibility, přestanou fungovat.
- Pokud jsou aplikace svázány se servisním programem LIBMQM na adrese IBM MQ for IBM i 8.0, neměli byste tyto aplikace znovu kompilovat nebo znovu propojit na adrese IBM MQ for IBM i 9.0 nebo novější.
- Ve stejné oblasti nelze nainstalovat více než jednu verzi produktu IBM MQ for IBM i .

Chcete-li zjistit, zda váš program v jazyce RPG nebo COBOL používá režim kompatibility, použijte k zobrazení externích programů volaných aplikačním programem příkaz **DSPPGMREF** (Display Program Reference). Pokud existují odkazy na programy uvedené v této sekci, program nebude spuštěn na IBM

MQ for IBM i 9.0 nebo později. Následující příklad výstupu příkazu **DSPPGMREF** zobrazuje tři zamítnuté programové objekty, MQCONN, MQOPEN, MQCLOSE:

```
Program . . . . . :  MYAPPPGM
Library . . . . . :    MYLIB
Text 'description'. . . . . :     ILE/COBOL SAMPLE PUT TO QUEUE (MQPUT)
Number of objects referenced  . . . . . :        5
Object  . . . . . :    MQCONN
Library . . . . . :      *LIBL
Object type . . . . . :        *PGM
Object  . . . . . :    MQOPEN
Library . . . . . :      *LIBL
Object type . . . . . :        *PGM
Object  . . . . . :    MQCLOSE
Library . . . . . :      *LIBL
Object type . . . . . :        *PGM
```

Tyto programy musí být překompilovány pomocí metody Bound Procedurální Call popsané v tématu Příprava programů v jazyce COBOL v produktu IBM i.

Pokusíte-li se spustit aplikační program v systému IBM MQ for IBM i 9.0 nebo novějším, který používá režim kompatibility, nejčastěji se zobrazí první chyba MCH3401 při pokusu o volání programu MQCONN nebo QMQM.

Související úlohy

Vývoj aplikací

IBM i Popisy datových typů na IBM i

Tato kolekce témat obsahuje popisy datových typů používaných při programování v jazyce IBM i .

Konvence používané v popisu datových typů

Pro každý základní datový typ tato informace poskytuje popis jejího použití ve formě, která je nezávislá na programovacím jazyku. Následuje typická deklarace ve verzi ILE programovacího jazyka RPG. Definice elementárních datových typů jsou zde zahrnuty, aby byla zajištěna konzistence. RPG používá specifikace 'D', kde lze pracovní pole deklarovat pomocí atributů, které potřebujete. Můžete to však provést ve specifikacích výpočtu, kde je pole použito.

Chcete-li použít základní datové typy, vytvořte:

- člen /COPY obsahující všechny datové typy nebo
- Externí datová struktura (PF) obsahující všechny datové typy. Poté musíte zadat svá pracovní pole s atributy 'LIKE' odpovídající pole datového typu.

Výhodou druhé volby je, že definice lze použít jako 'FIELD REFERENCE FILE' pro ostatní objekty IBM i . Pokud se změní definice datového typu IBM MQ , je relativně jednoduché tyto objekty znovu vytvořit.

Základní datové typy

Všechny ostatní datové typy popsané v tomto oddíle se rovnou buď přímo těmto elementárním datovým typům, nebo agregacím těchto elementárních datových typů (polí nebo struktur).

Tabulka 679. Základní datové typy

Datový typ	Zastoupení
MQBOOL	10místné celé číslo se znaménkem
MQBYTE	1bajtové alfanumerické pole
MQBYTE16	16bajtové alfanumerické pole
MQBYTE24	24bajtové alfanumerické pole
MQBYTE32	32bajtové alfanumerické pole

Tabulka 679. Základní datové typy (pokračování)	
Datový typ	**Zastoupení**
MQBYTE64	64bitové alfanumerické pole
MQCHAR	1bajtové alfanumerické pole
MQCHAR4	4bajtové alfanumerické pole
MQCHAR8	8bajtové alfanumerické pole
MQCHAR12	12bajtové alfanumerické pole
MQCHAR16	16bajtové alfanumerické pole
MQCHAR20	20bajtové alfanumerické pole
MQCHAR28	28bajtové alfanumerické pole
MQCHAR32	32bajtové alfanumerické pole
MQCHAR48	48bajtové alfanumerické pole
MQCHAR64	64bitové alfanumerické pole
MQCHAR128	128bajtové alfanumerické pole
MQCHAR256	256bajtové alfanumerické pole
MQFLOAT32	4bajtové číslo s pohyblivou řádovou čárkou
MQFLOAT64	8bajtové číslo s pohyblivou řádovou čárkou
MQHCONFIG	Popisovač konfigurace
MQHCONN	10místné celé číslo se znaménkem
MQHMSG	Popisovač zprávy, který poskytuje přístup ke zprávě
MQHOBJ	10místné celé číslo se znaménkem
MQINT8	8bitové celé číslo se znaménkem
MQINT16	16bitové celé číslo se znaménkem
MQINT32	32bitové celé číslo se znaménkem
MQINT64	64bitové celé číslo se znaménkem
MQLONG	32bitové celé číslo se znaménkem
MQPID	Identifikátor procesu
MQPTR	Ukazatel
MQTID	Identifikátor podprocesu
MQUINT8	8bitové celé číslo bez znaménka
MQUINT16	16bitové celé číslo bez znaménka
MQUINT32	32bitové celé číslo bez znaménka
MQUINT64	64bitové celé číslo bez znaménka
MQULONG (NEPOUŽITELNÝ)	32bitové celé číslo bez znaménka
PMQACH pro všechny	Ukazatel na datovou strukturu typu MQACH
PMQAIR	Ukazatel na datovou strukturu typu MQAIR

Tabulka 679. Základní datové typy (pokračování)	
Datový typ	**Zastoupení**
PMQAXC	Ukazatel na datovou strukturu typu MQAXC
PMAXP	Ukazatel na datovou strukturu typu MAXP
PMQBMHO	Ukazatel na datovou strukturu typu MQBMHO
PMQBO	Ukazatel na datovou strukturu typu MQBO
PMQBOOL	Ukazatel na data typu MQBOOL
PMQBYTE	Ukazatel na data typu MQBYTE
PMQBYTEn	Ukazatel na data typu MQBYTEn
PMQCBC	Ukazatel na datovou strukturu typu MQCBC
PMQCBD	Ukazatel na datovou strukturu typu MQCBD
PMQCHAR	Ukazatel na datovou strukturu typu MQCHAR
PMQCHARV	Ukazatel na datovou strukturu typu MQCHARV
PMQCHARn	Ukazatel na data typu MQCHARn
PMQCIH	Ukazatel na datovou strukturu typu MQCIH
PMQCMHO	Ukazatel na datovou strukturu typu MQCMHO
PMQCNO	Ukazatel na datovou strukturu typu MQCNO
PMQCSP.	Ukazatel na datovou strukturu typu MQCSP.
PMQCTLO	Ukazatel na datovou strukturu typu MQCTLO
PMQDH	Ukazatel na datovou strukturu typu MQDH
PMQDHO	Ukazatel na datovou strukturu typu MQDHO
PMQDLH	Ukazatel na datovou strukturu typu MQDLH
PMQDMHO	Ukazatel na datovou strukturu typu MQDMHO
PMQDMPO	Ukazatel na datovou strukturu typu MQDMPO
PMQEPH	Ukazatel na datovou strukturu typu MQEPH
PMQFLOAT32	Ukazatel na data typu MQFLOAT32
PMQFLOAT64	Ukazatel na data typu MQFLOAT64
PMQFUNC-uživatelské rozhraní	Ukazatel na funkci
PMQGMO	Ukazatel na datovou strukturu typu MQGMO
PMQHCONFIG	Ukazatel na data typu MQHCONFIG
PMQHCONN	Ukazatel na data typu MQHCONN
PMQHMSG	Ukazatel na data typu MQHMSG
PMQHOBJ	Ukazatel na data typu MQHOBJ
PMQIIH	Ukazatel na datovou strukturu typu MQIIH
PMQIMPO	Ukazatel na datovou strukturu typu MQIMPO
PMQINT8	Ukazatel na data typu MQINT8

Tabulka 679. Základní datové typy (pokračování)

Datový typ	**Zastoupení**
PMQINT16	Ukazatel na data typu MQINT16
PMQINT32	Ukazatel na data typu MQINT32
PMQINT64	Ukazatel na data typu MQINT64
PMQLONG	Ukazatel na data typu MQLONG
PMQMD	Ukazatel na datovou strukturu typu MQMD
PMQMDE	Ukazatel na datovou strukturu typu MQMDE
PMQMD1	Ukazatel na datovou strukturu typu MQMD1
PMQMD2	Ukazatel na datovou strukturu typu MQMD2
PMQMHBO	Ukazatel na datovou strukturu typu MQMHBO
PMQOD	Ukazatel na datovou strukturu typu MQOD
PMQOR	Ukazatel na datovou strukturu typu MQOR
PMQPD	Ukazatel na datovou strukturu typu MQPD
PMQPID	Ukazatel na identifikátor procesu MQPID
PMQPMO	Ukazatel na datovou strukturu typu MQPMO
PMQPTR	Ukazatel na data typu MQPTR
PMQRFH	Ukazatel na datovou strukturu typu MQRFH
PMQRFH2	Ukazatel na datovou strukturu typu MQRFH2
PMQRMH	Ukazatel na datovou strukturu typu MQRMH
PMQRR	Ukazatel na datovou strukturu typu MQRR
PMQSCO	Ukazatel na datovou strukturu typu MQSCO
PMQSD	Ukazatel na datovou strukturu typu MQSD
PMQSMPO	Ukazatel na datovou strukturu typu MQSMPO
PMQSRO	Ukazatel na datovou strukturu typu MQSRO
PMQSTS	Ukazatel na datovou strukturu typu MQSTS
PMQTID	Ukazatel na identifikátor podprocesu MQTID
PMQTM	Ukazatel na datovou strukturu typu MQTM
PMQTMC2	Ukazatel na datovou strukturu typu MQTMC2
PMQUINT8	Ukazatel na data typu MQUINT8
PMQUINT16	Ukazatel na data typu MQUINT16
PMQUINT32	Ukazatel na data typu MQUINT32
PMQUINT64	Ukazatel na data typu MQUINT64
PMQULONG	Ukazatel na data typu MQULONG
PMQVOID	Ukazatel
PMQWIH	Ukazatel na datovou strukturu typu MQWIH

Tabulka 679. Základní datové typy (pokračování)	
Datový typ	**Zastoupení**
PMQXQH	Ukazatel na datovou strukturu typu MQXQH

IBM i *MQBOOL na IBM i*

Datový typ MQBOOL představuje logickou hodnotu. Hodnota 0 představuje hodnotu false. Jakákoli jiná hodnota představuje hodnotu true.

Pro datový typ MQLONG musí být zarovnán objekt MQBOOL.

IBM i *MQBYTE v systému IBM i*

Datový typ MQBYTE představuje jeden bajt dat.

Na bajt není umístěna žádná konkrétní interpretace-je s ním zacházeno jako s řetězcem bitů, a ne jako s binárním číslem nebo znakem. Není vyžadováno žádné speciální zarovnání.

Pole MQBYTE se někdy používá k reprezentaci oblasti hlavní paměti s povahou, která není správci front známa. Oblast může například obsahovat data zprávy aplikace nebo strukturu. Zarovnání hranic této oblasti musí být kompatibilní s povahou dat v ní obsažených.

IBM i *MQBYTEn (Řetězec n bajtů) na IBM i*

Každý datový typ MQBYTEn představuje řetězec *n* bajtů.

Kde *n* může mít jednu z následujících hodnot:

- 16, 24, 32 nebo 64.

Každý bajt je popsán datovým typem MQBYTE. Není vyžadováno žádné speciální zarovnání.

Pokud jsou data v řetězci kratší než definovaná délka řetězce, musí být data vyplněna nulami, aby se vyplnil řetězec.

Když správce front vrátí aplikaci bajtové řetězce (například ve volání MQGET), správce front vždy provede výpad s nulami na definovanou délku řetězce.

K dispozici jsou konstanty, které definují délku polí bajtového řetězce.

IBM i *MQCHAR (znak) na IBM i*

Datový typ MQCHAR představuje jeden znak.

Identifikátor kódované znakové sady je identifikátor správce front (viz atribut **CodedCharSetId** v tématu CodedCharSetId). Není vyžadováno žádné speciální zarovnání.

Poznámka: Data zpráv aplikace určená ve voláních MQGET, MQPUT a MQPUT1 jsou popsána datovým typem MQBYTE, nikoli datovým typem MQCHAR.

IBM i *MQCHARn (String of n characters) na IBM i*

Každý datový typ MQCHARn představuje řetězec *n* znaků.

Kde *n* může mít jednu z následujících hodnot:

- 4, 8, 12, 16, 20, 28, 32, 48, 64, 128 nebo 256

Každý znak je popsán datovým typem MQCHAR. Není vyžadováno žádné speciální zarovnání.

Pokud jsou data v řetězci kratší než definovaná délka řetězce, musí být data vyplněna mezerami, aby se řetězec vyplnil. V některých případech lze k předčasnému ukončení řetězce použít znak null namísto vyplnění mezerami; znak null a následující znaky jsou považovány za mezery, až do definované délky řetězce. Místa, kde lze použít hodnotu null, jsou identifikována v popisech volání a datových typů.

Když správce front vrátí aplikaci znakové řetězce (například ve volání MQGET), správce front vždy vyplní znaky s mezerami na definovanou délku řetězce; správce front k oddělení řetězce nepoužívá znak null.

K dispozici jsou konstanty, které definují délku polí znakového řetězce.

IBM i *MQFLOAT32 on IBM i*

Datový typ MQFLOAT32 je 32bitové číslo s pohyblivou řádovou čárkou reprezentované pomocí standardního formátu IEEE s pohyblivou řádovou čárkou.

MQFLOAT32 musí být zarovnán na 4bajtové hranici.

IBM i *MQFLOAT64 na IBM i*

Datový typ MQFLOAT64 je 64bitové číslo s pohyblivou řádovou čárkou reprezentované pomocí standardního formátu IEEE s pohyblivou řádovou čárkou.

MQFLOAT64 musí být zarovnán na 8bajtové hranici.

MQHCONFIG-popisovač konfigurace

Datový typ MQHCONFIG představuje popisovač konfigurace, tj. komponentu, která se konfiguruje pro konkrétní instalovatelnou službu. Popisovač konfigurace musí být zarovnán na své přirozené hranici.

Poznámka: Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

IBM i *MQHCONN (manipulátor připojení) na systému IBM i*

Datový typ MQHCONN představuje manipulátor připojení, tj. připojení ke konkrétnímu správci front.

Manipulátor připojení musí být zarovnán na své přirozené hranici.

Poznámka: Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

IBM i *MQHMSG (popisovač zprávy) na systému IBM i*

Datový typ **MQHMSG** představuje popisovač zprávy, který poskytuje přístup ke zprávě.

Popisovač zprávy musí být zarovnán na 8bajtové hranici.

Poznámka: Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

IBM i *MQHOBJ (popisovač objektu) na systému IBM i*

Datový typ MQHOBJ představuje popisovač objektu, který poskytuje přístup k objektu.

Popisovač objektu musí být zarovnán na své přirozené hranici.

Poznámka: Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

IBM i *MQINT8 (8bitové celé číslo se znaménkem) na IBM i*

Datový typ MQINT8 je 8bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -128 až +127, není-li kontext jinak omezen.

IBM i *MQINT16 (16bitové celé číslo se znaménkem) na IBM i*

Datový typ MQINT16 je 16bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -32 768 až +32 767, není-li kontext jinak omezen.

MQINT16 musí být zarovnán na 2bajtové hranici.

IBM i *MQINT32 (32bitové celé číslo) na IBM i*

Datový typ MQINT32 je 32bitové celé číslo se znaménkem.

Jedná se o ekvivalent k MQLONG.

IBM i *MQINT64 (64bitové celé číslo) na IBM i*

Datový typ MQINT64 je 64bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -9 223 372 036 854 775 808 až + 9 223 372 036 854 775 807, pokud kontext nestanoví jinak.

Pro COBOL je platný rozsah omezen na -999 999 999 999 999 999 999 999 až +999 999 999 999 999 999 999 999. MQINT64 by měl být zarovnán na 8bajtové hranici.

IBM i *MQLONG (dlouhé celé číslo) na IBM i*

Datový typ MQLONG je 32bitové binární celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -2 147 483 648 až + 2 147 483 647, není-li jinak omezeno kontextem, zarovnané na jeho přirozené hranici.

MQPID-identifikátor procesu

Identifikátor procesu IBM MQ .

Jedná se o stejný identifikátor, který se používá ve výpisech paměti IBM MQ trace a FFST , ale může se lišit od identifikátoru procesu operačního systému.

MQPTR-ukazatel

Datový typ MQPTR je adresa dat libovolného typu. Ukazatel musí být zarovnán na své přirozené hranici; jedná se o 16bajtovou hranici na IBM i.

Některé programovací jazyky podporují typované ukazatele; rozhraní MQI je také používá v několika případech.

MQTID-identifikátor podprocesu

Identifikátor podprocesu MQ .

Jedná se o stejný identifikátor, který se používá v trasování produktu MQ a ve výpisech paměti produktu FFST , ale může se lišit od identifikátoru podprocesu operačního systému.

IBM i *MQUINT8 (8bitové celé číslo bez znaménka) na IBM i*

Datový typ MQUINT8 je 8bitové celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu od 0 do +255, není-li kontext jinak omezen.

MQUINT16 -16bitové celé číslo bez znaménka

Datový typ MQUINT16 je 16bitové celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu 0 až +65 535, pokud kontext nestanoví jinak.

MQUINT16 musí být zarovnán na 2bajtové hranici.

IBM i *MQUINT32 (32bitové celé číslo bez znaménka) na IBM i*

Datový typ MQUINT32 je 32bitové celé číslo bez znaménka. Je ekvivalentní s MQULONG.

MQUINT64 -64bitové celé číslo bez znaménka

Datový typ MQUINT64 je 64bitové celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu od 0 do +18 446 744 073 709 551 615, není-li kontext jinak omezen.

Pro COBOL je platný rozsah omezen na 0 až +999 999 999 999 999 999. MQUINT64 by měl být zarovnán na 8bajtové hranici.

MQULONG-32bitové celé číslo bez znaménka
Datový typ MQULONG je 32bitové binární celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu 0 až + 4 294 967 294, není-li kontext jinak omezen.

Hodnota MQULONG musí být zarovnána na 4bajtové hranici.

PMQACH-ukazatel na datovou strukturu typu MQACH
Ukazatel na datovou strukturu typu MQACH.

PMQAIR-ukazatel na datovou strukturu typu MQAIR
Ukazatel na datovou strukturu typu MQAIR.

PMQAXC-ukazatel na datovou strukturu typu MQAXC
Ukazatel na datovou strukturu typu MQAXC.

PMQAXP-ukazatel na datovou strukturu typu MQAXP
Ukazatel na datovou strukturu typu MQAXP.

PMQBMHO-ukazatel na datovou strukturu typu MQBMHO
Ukazatel na datovou strukturu typu MQBMHO.

PMQBO-ukazatel na datovou strukturu typu MQBO
Ukazatel na datovou strukturu typu MQBO.

PMQBOOL-ukazatel na data typu MQBOOL
Ukazatel na data typu MQBOOL.

Ukazatel na data typu MQBOOL.

PMQBYTE-ukazatel na datový typ MQBYTE
Ukazatel na datový typ MQBYTE.

PMQBYTEn-ukazatel na datovou strukturu typu MQBYTEn
Ukazatel na datovou strukturu typu MQBYTEn, kde n může být 8, 12, 16, 24, 32, 40, 48 nebo 128.

PMQCBC-ukazatel na datovou strukturu typu MQCBC
Ukazatel na datovou strukturu typu MQCBC.

PMQCBD-ukazatel na datovou strukturu typu MQCBD
Ukazatel na datovou strukturu typu MQCBD.

PMQCHAR-ukazatel na data typu MQCHAR
Ukazatel na data typu MQCHAR.

PMQCHARV-ukazatel na datovou strukturu typu MQCHARV
Ukazatel na datovou strukturu typu MQCHARV.

PMQCHARn-ukazatel na datový typ MQCHARn
Ukazatel na datový typ MQCHARn, kde n může být 4, 8, 12, 20, 28, 32, 64, 128, 256, 264.

PMQCIH-ukazatel na datovou strukturu typu MQCIH
Ukazatel na datovou strukturu typu MQCIH.

PMQCMHO-ukazatel na datovou strukturu typu MQCMHO
Ukazatel na datovou strukturu typu MQCMHO.

PMQCNO-ukazatel na datovou strukturu typu MQCNO
Ukazatel na datovou strukturu typu MQCNO.

PMQCSP-ukazatel na datovou strukturu typu MQCSP.
Ukazatel na datovou strukturu typu MQCSP.

PMQCTLO-ukazatel na datovou strukturu typu MQCTLO
Ukazatel na datovou strukturu typu MQCTLO.

PMQDH-ukazatel na datovou strukturu typu MQDH
Ukazatel na datovou strukturu typu MQDH.

PMQDHO-ukazatel na datovou strukturu typu MQDHO
Ukazatel na datovou strukturu typu MQDHO.

PMQDLH-ukazatel na datovou strukturu typu MQDLH
Ukazatel na datovou strukturu typu MQDLH.

PMQDMHO-ukazatel na datovou strukturu typu MQDMHO
Ukazatel na datovou strukturu typu MQDMHO.

PMQDMPO-ukazatel na datovou strukturu typu MQDMPO
Ukazatel na datovou strukturu typu MQDMPO.

Ukazatel na datovou strukturu typu MQDMPO.

PMQEPH-ukazatel na datovou strukturu typu MQEPH
Ukazatel na datovou strukturu typu MQEPH.

PMQFLOAT32 -ukazatel na data typu MQFLOAT32
Ukazatel na data typu MQFLOAT32.

PMQFLOAT64 -Ukazatel na data typu MQFLOAT64
Ukazatel na data typu MQFLOAT64.

PMQFUNC-ukazatel na funkci
Ukazatel na funkci.

PMQGMO-ukazatel na datovou strukturu typu MQGMO
Ukazatel na datovou strukturu typu MQGMO.

PMQHCONFIG-ukazatel na datový typ MQHCONFIG
Ukazatel na datový typ MQHCONFIG.

PMQHCONN-ukazatel na datový typ MQHCONN
Ukazatel na datový typ MQHCONN.

PMQHMSG-ukazatel na datový typ MQHMSG
Ukazatel na datový typ MQHMSG.

PMQHOBJ-ukazatel na data typu MQHOBJ
Ukazatel na data typu MQSMPO.

PMQIIH-ukazatel na datovou strukturu typu MQIIH
Ukazatel na datovou strukturu typu MQIIH.

PMQIMPO-ukazatel na datovou strukturu typu MQIMPO
Ukazatel na datovou strukturu typu MQIMPO.

PMQINT8 -ukazatel na data typu MQINT8
Ukazatel na data typu MQINT8.

PMQINT16 -Ukazatel na data typu MQINT16
Ukazatel na data typu MQINT16.

IBM i *PMQINT32 (ukazatel na data typu MQINT32) na systému IBM i*
Datový typ PMQINT32 je ukazatel na data typu MQINT32. Je ekvivalentní k PMQLONG.

IBM i *PMQINT64 (ukazatel na data typu MQINT64) na systému IBM i*
Datový typ PMQINT64 je ukazatel na data typu MQINT64.

PMQLONG-ukazatel na data typu MQLONG
Ukazatel na data typu MQLONG.

PMQMD-ukazatel na strukturu typu MQMD
Ukazatel na strukturu typu MQMD.

PMQMDE-ukazatel na datovou strukturu typu MQMDE
Ukazatel na datovou strukturu typu MQMDE.

PMQMDI-ukazatel na datovou strukturu typu MQMDI
Ukazatel na datovou strukturu typu MQMDI.

PMQMD2 -Ukazatel na datovou strukturu typu MQMD2 .
Ukazatel na datovou strukturu typu MQMD2

PMQMHBO-ukazatel na datovou strukturu typu MQMHBO
Ukazatel na datovou strukturu typu MQMHBO.

PMQOD-ukazatel na datovou strukturu typu MQOD
Ukazatel na datovou strukturu typu MQOD.

PMQOR-ukazatel na datovou strukturu typu MQOR
Ukazatel na datovou strukturu typu MQOR.

PMQPD-ukazatel na datovou strukturu typu MQPD
Ukazatel na datovou strukturu typu MQPD.

PMQPID-ukazatel na identifikátor procesu
Ukazatel na identifikátor procesu.

PMQPMO-ukazatel na datovou strukturu typu MQPMO
Ukazatel na datovou strukturu typu MQPMO.

PMQPTR-ukazatel na data typu MQPTR
Ukazatel na data typu MQPTR.

PMQRFH-ukazatel na datovou strukturu typu MQRFH
Ukazatel na datovou strukturu typu MQRFH.

PMQRFH2 -ukazatel na datovou strukturu typu MQRFH2
Ukazatel na datovou strukturu typu MQRFH2.

.

PMQRMH-ukazatel na datovou strukturu typu MQRMH
Ukazatel na datovou strukturu typu MQRMH.

PMQRR-ukazatel na datovou strukturu typu MQRR
Ukazatel na datovou strukturu typu MQRR.

PMQSCO-ukazatel na datovou strukturu typu MQSCO
Ukazatel na datovou strukturu typu MQSCO.

.

PMQSD-ukazatel na datovou strukturu typu MQSD
Ukazatel na datovou strukturu typu MQSD.

PMQSMPO-ukazatel na datovou strukturu typu MQSMPO
Ukazatel na datovou strukturu typu MQSMPO.

PMQSRO-ukazatel na datovou strukturu typu MQSRO
Ukazatel na datovou strukturu typu MQSRO.

PMQSTS-ukazatel na datovou strukturu typu MQSTS
Ukazatel na datovou strukturu typu MQSTS.

PMQTID-ukazatel na datovou strukturu typu MQTID
Ukazatel na datovou strukturu typu MQTID.

PMQTM-ukazatel na datovou strukturu typu MQTM
Ukazatel na datovou strukturu typu MQTM.

PMQTMC2 -ukazatel na datovou strukturu typu MQTMC2
Ukazatel na datovou strukturu typu MQTMC2.

PMQUINT8 -ukazatel na data typu MQUINT8
Ukazatel na data typu MQUINT8.

PMQUINT16 -ukazatel na data typu MQUINT16
Ukazatel na data typu MQUINT16.

IBM i *PMQUINT32 (ukazatel na data typu MQUINT32) na systému IBM i*
Datový typ PMQUINT32 je ukazatel na data typu MQUINT32. Je ekvivalentní s PMQULONG.

IBM i *PMQUINT64 (ukazatel na data typu MQUINT64) na systému IBM i*
Datový typ PMQUINT64 je ukazatel na data typu MQUINT64.

PMQULONG-ukazatel na data typu MQULONG
Ukazatel na data typu MQULONG.

PMQVOID-ukazatel
Ukazatel.

PMQWIH-ukazatel na datovou strukturu typu MQWIH
Ukazatel na datovou strukturu typu MQWIH.

PMQXQH-ukazatel na datovou strukturu typu MQXQH
Ukazatel na datovou strukturu typu MQXQH.

Jazykové aspekty
Toto téma obsahuje informace, které vám pomohou používat rozhraní MQI z programovacího jazyka RPG.

Některé z těchto jazykových aspektů jsou:

- “Soubory COPY” na stránce 1006
- “Volání” na stránce 1008
- “Parametry volání” na stránce 1008
- “Struktury” na stránce 1008
- “Pojmenované konstanty” na stránce 1008
- “Procedury MQI” na stránce 1008
- “Pokyny pro podprocesy” na stránce 1009

- “Vázané zpracování” na stránce 1009
- “Kódování vázaných volání” na stránce 1009
- “Notářské konvence” na stránce 1010

Soubory COPY

Různé soubory COPY jsou poskytovány jako pomůcka při psaní aplikačních programů RPG, které používají řazení zpráv do front. Existují tři sady souborů COPY:

- Soubory COPY s názvy končícími písmenem *G* jsou určeny pro použití s programy, které používají statické propojení. Tyto soubory jsou inicializovány s výjimkami uvedenými v souboru “Struktury” na stránce 1008.
- Soubory COPY s názvy končícími písmenem *H* jsou určeny pro programy, které používají statické sestavení, ale **nejsou** inicializovány.
- Soubory COPY s názvy končícími písmenem *R* jsou určeny pro použití s programy, které používají dynamické propojení. Tyto soubory jsou inicializovány s výjimkami uvedenými v souboru “Struktury” na stránce 1008.

Soubory COPY se nacházejí v QRPGLESRC v knihovně QMQM.

Pro každou sadu souborů COPY existují dva soubory obsahující pojmenované konstanty a jeden soubor pro každou ze struktur. Soubory COPY jsou shrnuty v souboru Tabulka 680 na stránce 1006.

Tabulka 680. RPG COPY soubory

Název souboru (statické propojení, inicializováno, CMQ* G)	**Název souboru (statické sestavení, neinicializováno, CMQ* H)**	**Název souboru (dynamické sestavení, inicializováno, CMQ* R)**	**Obsah**
CMQBOG	CMQBOH	-	Struktura voleb začátku
CMQCDG	CMQCDH	CMQCDR	Struktura definice kanálu
CMQCFBFG	CMQCFBFH	-	Parametr bitového filtru PCF
CMQCFG	-	-	Konstanty pro PCF a události
CMQCFBSG	CMQCFBSH	-	Bajtový řetězec PCF
CMQCFGRG	CMQCFGRH	-	Parametr skupiny PCF
CMQCFIFG	CMQCFIFH	-	Parametr filtru celého čísla PCF
CMQCFHG	CMQCFHH	-	Záhlaví PCF
CMQCFILG	CMQCFILH	-	Struktura parametrů seznamu celých čísel PCF
CMQCFING	CMQCFINH	-	Struktura celočíselných parametrů PCF
CMQCFSFG	CMQCFSFH	-	Parametr filtru řetězců PCF
CMQCFSLG	CMQCFSLH	-	Struktura parametrů seznamu řetězců PCF
CMQCFSTG	CMQCFSTH	-	Struktura řetězcového parametru PCF
CMQCFXLG	CMQCFXLH	-	Krátký název PCF pro CFIL64
CMQCFXNG	CMQCFXNH	-	Krátký název PCF pro CFIN64
CMQCIHG	CMQCIHH	-	Struktura záhlaví informací CICS

Tabulka 680. RPG COPY soubory (pokračování)

Název souboru (statické propojení, inicializováno, CMQ* G)	**Název souboru (statické sestavení, neinicializováno, CMQ* H)**	**Název souboru (dynamické sestavení, inicializováno, CMQ* R)**	**Obsah**
CMQCNOG	CMQCNOH	-	Struktura voleb připojení
CMQCSPG	CMQCSPH	-	Parametry zabezpečení
CMQCXPG	CMQCXPH	CMQCXPR	Struktura parametrů uživatelské procedury kanálu
CMQDHG	CMQDHH	CMQDHR	Struktura záhlaví distribuce
CMQDLHG	CMQDLHH	CMQDLHR	Struktura záhlaví nedoručovacího dopisu
CMQDXPG	CMQDXPH	CMQDXPR	Struktura parametrů uživatelské procedury pro převod dat
CMQEPHG	CMQEPHH	-	Vložená struktura záhlaví PCF
CMQG	-	CMQR	Pojmenované konstanty pro hlavní rozhraní MQI
CMQGMOG	CMQGMOH	CMQGMOR	Získat strukturu voleb zprávy
CMQIIHG:	CMQIIHH	CMQIIHR	Struktura záhlaví informací IMS
CMQMDEG	CMQMDEH	CMQMDER	Struktura rozšíření deskriptoru zpráv
CMQMDG	CMQMDH	CMQMDR	Struktura deskriptoru zpráv
CMQMD1G	CMQMD1H	CMQMD1R	Struktura deskriptoru zpráv verze 1
CMQMD2G	CMQMD2H	-	Struktura deskriptoru zpráv verze 2
CMQODG	CMQODH	CMQODR	Struktura deskriptoru objektu
CMQORG	CMQORH	CMQORR	Struktura záznamu objektu
CMQPMOG	CMQPMOH	CMQPMOR	Struktura voleb vložení zprávy
CMQPSG	-	-	Konstanty pro publikování/odběr
CMQRFHG	CMQRFHH	-	Pravidla a struktura záhlaví formátování
CMQRFH2G	CMQRFH2H	-	Struktura pravidel a formátování záhlaví 2
CMQRMHG	CMQRMHH	CMQRMHR	Struktura záhlaví referenční zprávy
CMQRRG	CMQRRH	CMQRRR	Struktura záznamu odezvy
CMQTMCG	CMQTMCH	CMQTMCR	Struktura zprávy spouštěče (znakový formát)
CMQTMC2G	CMQTMC2H	CMQTMC2R	Struktura zprávy spouštěče (znakový formát) verze 2
CMQTMG	CMQTMH	CMQTMR	Struktura zprávy spouštěče
CMQWIHG:	CMQWIHH	-	Struktura záhlaví informací o práci
CMQXG	-	CMQXR	Pojmenované konstanty pro uživatelskou proceduru převodu dat

Tabulka 680. RPG COPY soubory (pokračování)			
Název souboru (statické propojení, inicializováno, CMQ* G)	**Název souboru (statické sestavení, neinicializováno, CMQ* H)**	**Název souboru (dynamické sestavení, inicializováno, CMQ* R)**	**Obsah**
CMQXQHG	CMQXQHH	CMQXQHR	Struktura záhlaví přenosové fronty

Volání

Volání jsou popsána pomocí jejich individuálních jmen.

Parametry volání

Některé parametry předané rozhraní MQI mohou mít více než jednu souběžnou funkci. Je to proto, že předaná celočíselná hodnota je často testována na nastavení jednotlivých bitů v poli, a ne na jeho celkové hodnotě. To vám umožní 'přidat' několik funkcí dohromady a předat je jako jeden parametr.

Struktury

Všechny struktury IBM MQ jsou definovány s počátečními hodnotami pro pole s následujícími výjimkami:

- Libovolná struktura s příponou H.
- MQTMC
- MQTMC2

Tyto počáteční hodnoty jsou definovány v příslušné tabulce pro každou strukturu.

Deklarace struktury neobsahují příkazy DS . To umožňuje aplikaci deklarovat buď jednu datovou strukturu, nebo strukturu dat s více výskyty, kódováním příkazu DS a následným použitím příkazu /COPY ke zkopírování ve zbytku deklarace:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7
D* Declare an MQMD data structure with 5 occurrences
DMYMD        DS                            5
D/COPY CMQMDR
```

Pojmenované konstanty

Existuje mnoho celočíselných a znakových hodnot, které poskytují výměnu dat mezi aplikačním programem a správcem front. Pro usnadnění čitelnějšího a konzistentnějšího přístupu k používání těchto hodnot jsou pro ně definovány pojmenované konstanty. Tyto pojmenované konstanty můžete použít, a ne hodnoty, které představují, protože to zlepšuje čitelnost zdrojového kódu programu.

Když je soubor CMQG COPY zahrnut do programu pro definování konstant, kompilátor RPG vydá mnoho zpráv s nulovou závažností pro konstanty, které program nepoužívá; tyto zprávy jsou neškodné a lze je bezpečně ignorovat.

Procedury MQI

Při použití volání ILE bound musíte při vytváření programu vytvořit vazbu s procedurami MQI. Tyto procedury jsou podle potřeby exportovány z následujících servisních programů:

QMQM/LIBMQM
Tento servisní program obsahuje vazby s jedním podprocesem pro verzi 5.1 a vyšší. Speciální aspekty při psaní aplikací s podporou podprocesů naleznete v následující části.

QMQM/LIBMQM_R
Tento servisní program obsahuje vazby s podporou podprocesů pro verzi 5.1 a vyšší. Speciální aspekty při psaní aplikací s podporou podprocesů naleznete v následující části.

QMQM/LIBMQIC
Tento servisní program je určen pro vázání aplikací klienta bez podprocesů.

QMQM/LIBMQIC_R
Tento servisní program je určen pro vázání aplikací klienta s podporou podprocesů.

K vytvoření programů použijte příkaz CRTPGM . Například následující příkaz vytvoří program s jedním podprocesem, který používá volání ILE:

```
CRTPGM PGM(MYPROGRAM) BNDSRVPGM(QMQM/LIBMQM)
```

Pokyny pro podprocesy

Kompilátor RPG použitý pro produkt IBM i je součástí WebSphere Development Toolset a WebSphere Development Studio for IBM i a je znám jako kompilátor ILE RPG IV.

Obecně platí, že programy RPG by neměly používat servisní programy s podporou podprocesů. Výjimkami jsou programy RPG vytvořené pomocí kompilátoru ILE RPG IV a obsahující klíčové slovo THREAD(*SERIALIZE) ve specifikaci řízení. Avšak i když jsou tyto programy bezpečné pro podprocesy, je třeba pečlivě zvážit celkový návrh aplikace, protože produkt THREAD(*SERIALIZE) vynucuje serializaci procedur RPG na úrovni modulu, což může mít nepříznivý vliv na celkový výkon.

Pokud jsou programy RPG používány jako uživatelské procedury pro převod dat, musí být zabezpečeny podprocesy a měly by být znovu zkompilovány pomocí kompilátoru ILE RPG verze 4.4 nebo vyšší s parametrem THREAD(*SERIALIZE) uvedeným ve specifikaci řízení.

Další informace o použití podprocesů naleznete v příručce *IBM i IBM MQ Development Studio: ILE RPG Referenc*e a v příručce *IBM i IBM MQ Development Studio: ILE RPG Programmer's Guide*.

Vázané zpracování

Funkce synchronizačního bodu MQI MQCMIT a MQBACK jsou k dispozici programům ILE RPG spuštěným v normálním režimu; tato volání umožňují programu potvrdit a vrátit zpět změny prostředků MQ .

Kódování vázaných volání

Procedury MQI ILE jsou uvedeny v seznamu Tabulka 681 na stránce 1009.

Tabulka 681. Volání vázaná na ILE RPG podporovaná každým servisním programem

Název volání	LIBMQM a LIBMQM_R	LIBMQIC a LIBMQIC_R
MQBACK	Y	Y
MQBEGIN	Y	Y
MQCMIT	Y	Y
MQCLOSE	Y	Y
MQCONN	Y	Y
MQCONNX	Y	Y
MQDISC	Y	Y
MQGET	Y	Y
MQINQ	Y	Y
MQOPEN	Y	Y

Tabulka 681. Volání vázaná na ILE RPG podporovaná každým servisním programem (pokračování)		
Název volání	**LIBMQM a LIBMQM_R**	**LIBMQIC a LIBMQIC_R**
MQPUT	Y	Y
MQPUT1	Y	Y
MQSET	Y	Y
MQXCNVC	Y	Y

Chcete-li použít tyto procedury, musíte:

1. Ve specifikacích "D" definujte externí procedury. Všechny jsou k dispozici v rámci členu CMQG souboru COPY, který obsahuje pojmenované konstanty.
2. K volání procedury spolu s jejími parametry použijte kód operace CALLP.

Například volání MQOPEN vyžaduje zahrnutí následujícího kódu:

```
D*------------------------------------------------------------------
D**  MQOPEN Call -- Open Object  (From COPY file CMQG)           **
D*------------------------------------------------------------------
D*
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQOPEN           PR                  EXTPROC('MQOPEN')
D* Connection handle
D HCONN                          10I 0 VALUE
D* Object descriptor
D OBJDSC                        224A
D* Options that control the action of MQOPEN
D OPTS                           10I 0 VALUE
D* Object handle
D HOBJ                           10I 0
D* Completion code
D CMPCOD                         10I 0
D* Reason code qualifying CMPCOD
D REASON                         10I 0
D*
```

Chcete-li volat proceduru po inicializaci různých parametrů, potřebujete následující kód:

```
...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+....8
    C                   CALLP     MQOPEN(HCONN : MQOD : OPTS : HOBJ :
    C                             CMPCOD : REASON)
```

Zde je struktura MQOD definována pomocí člena COPY CMQODG, který ji rozdělí do svých komponent.

Notářské konvence

Poslední témata v této části ukazují, jak:

- Volání by měla být vyvolána
- Parametry by měly být deklarovány
- Měly by být deklarovány různé datové typy

V řadě případů jsou parametry pole nebo znakové řetězce s velikostí, která není pevná. Pro tyto účely se k reprezentaci číselné konstanty používá malá písmena "n" . Je-li deklarace pro tento parametr kódována, musí být hodnota "n" nahrazena požadovanou číselnou hodnotou.

IBM i MQAIR (záznam ověřovacích informací) na IBM i

Struktura MQAIR představuje záznam ověřovacích informací.

Přehled

Účel: Struktura MQAIR umožňuje aplikaci spuštěné jako klient IBM MQ zadat informace o ověřovateli, který má být použit pro připojení klienta. Struktura je vstupní parametr volání MQCONNX.

Znaková sada a kódování: Data v MQAIR musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT.

- "Pole" na stránce 1011
- "Počáteční hodnoty" na stránce 1013
- "Deklarace RPG" na stránce 1013

Pole

Struktura MQAIR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

AICN (10místné celé číslo se znaménkem)

Jedná se buď o název hostitele, nebo síťovou adresu hostitele, na kterém je spuštěn server LDAP. Za ním může následovat volitelné číslo portu uzavřené v závorkách.

Pokud je hodnota kratší než délka pole, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku pole. Pokud hodnota není platná, volání selže s kódem příčiny RC2387.

Výchozí číslo portu je 389.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou LNAICN. Počáteční hodnota tohoto pole je prázdná.

AITYP (10místné celé číslo se znaménkem)

Jedná se o typ ověřovacích informací obsažených v záznamu.

Hodnota musí být:

AITLDP
Odvolání certifikátu pomocí serveru LDAP.

Pokud hodnota není platná, volání selže s kódem příčiny RC2386.

Toto je vstupní pole. Počáteční hodnota tohoto pole je AITLDP.

AIPW (10místné celé číslo se znaménkem)

Toto je heslo potřebné pro přístup k serveru CRL LDAP.

Pokud je hodnota kratší než délka pole, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku pole. Pokud server LDAP nevyžaduje heslo, nebo pokud vynecháte jméno uživatele LDAP, *AIPW* musí mít hodnotu null nebo musí být prázdné. Pokud vynecháte jméno uživatele LDAP a *AIPW* nemá hodnotu null nebo není prázdné, volání selže s kódem příčiny RC2390.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou LNLDPW. Počáteční hodnota tohoto pole prázdných znaků.

AILUL (celé číslo se znaménkem 10 číslic)

Jedná se o délku (v bajtech) jména uživatele LDAP adresovaného polem *AILUP* nebo *AILUO* . Hodnota musí být v rozsahu nula až LNDISN. Pokud hodnota není platná, volání selže s kódem příčiny RC2389.

Pokud daný server LDAP nevyžaduje jméno uživatele, nastavte toto pole na nulu.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

AILUO (10místné celé číslo se znaménkem)

Jedná se o posun v bajtech jména uživatele LDAP od začátku struktury MQAIR.

Posun může být kladný nebo záporný. Je-li hodnota *LDAPUserNameLength* nula, pole se ignoruje.

Můžete použít buď *LDAPUserNamePtr* , nebo *LDAPUserNameOffset* , abyste uvedli jméno uživatele LDAP, ale ne obojí; podrobnosti viz popis pole *LDAPUserNamePtr* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

AILUP (10místné celé číslo se znaménkem)

Jedná se o jméno uživatele LDAP.

Skládá se z rozlišujícího názvu uživatele, který se pokouší o přístup k serveru CRL LDAP. Pokud je hodnota kratší než délka uvedená parametrem *AILUL*, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku *AILUL*. Je-li hodnota *AILUL* nula, pole se ignoruje.

Jméno uživatele LDAP můžete zadat jedním ze dvou způsobů:

- Pomocí pole ukazatele *AILUP*

  V tomto případě může aplikace deklarovat řetězec, který je oddělený od struktury MQAIR, a nastavit *AILUP* na adresu řetězce.

  Zvažte použití produktu *AILUP* pro programovací jazyky, které podporují datový typ ukazatele způsobem, který je přenositelný do různých prostředí (například programovací jazyk C).

- Pomocí pole offsetu *AILUO*

  V tomto případě musí aplikace deklarovat složenou strukturu obsahující strukturu MQSCO následovanou polem záznamů MQAIR následovaným řetězci jména uživatele LDAP a nastavit hodnotu *AILUO* na posun příslušného řetězce názvu od začátku struktury MQAIR. Ujistěte se, že je tato hodnota správná a má hodnotu, kterou lze umístit v rámci MQLONG (nejrestriktivnějším programovacím jazykem je COBOL, pro který je platný rozsah -999 999 999 999 až +999 999 999 999).

  Zvažte použití *AILUO* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele způsobem, který nemusí být přenositelný do různých prostředí (například programovací jazyk COBOL).

Bez ohledu na zvolenou techniku použijte pouze jednu z možností *AILUP* a *AILUO* ; volání selže s kódem příčiny RC2388.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou typu all-null.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

AISID (10místné celé číslo se znaménkem)

Hodnota musí být:

AISIDV
Identifikátor záznamu ověřovacích informací.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je AISIDV.

AIVER (10místné celé číslo se znaménkem)

Hodnota musí být:

AIVER1
Záznam ověřovacích informací Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

AIRVERC
Aktuální verze záznamu ověřovacích informací.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je AIVER1.

Počáteční hodnoty

Tabulka 682. Pole v MQAIR pro MQAIR		
Název pole	**Název konstanty**	**Hodnota konstanty**
AISID	AISIDV	'AIR¬'
AIVER	AIVERC	1
AITYP	AITLDP	1
AICN	Není	Prázdný řetězec nebo mezery
AILUP	Není	Ukazatel Null nebo bajty s hodnotou Null
AILUO	Není	0
AILUL	Není	0
AIPW	Není	Prázdný řetězec nebo mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQAIR Structure
D*
D* Structure identifier
D  AISID                  1      4    INZ('AIR ')
D* Structure version number
D  AIVER                  5      8I 0 INZ(1)
D* Type of authentication information
D  AITYP                  9     12I 0 INZ(1)
D* Connection name of CRL LDAP server
D  AICN                  13    276    INZ
D* Address of LDAP user name
D  AILUP                277    292*   INZ(*NULL)
D* Offset of LDAP user name from start of MQAIR structure
D  AILUO                293    296I 0 INZ(0)
D* Length of LDAP user name
D  AILUL                297    300I 0 INZ(0)
D* Password to access LDAP server
D  AIPW                 301    332    INZ
```

IBM i MQBMHO (volby vyrovnávací paměti pro obsluhu zpráv) v systému IBM i

Struktura definující volby zpracování vyrovnávací paměti pro zprávu.

Přehled

Účel: Struktura MQBMHO umožňuje aplikacím určit volby, které řídí způsob zpracování zpráv z vyrovnávacích pamětí. Struktura je vstupní parametr volání MQBUFMH.

Znaková sada a kódování: Data v MQBMHO musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1014
- "Počáteční hodnoty" na stránce 1014

- “Deklarace RPG” na stránce 1014

Pole

Struktura MQBMHO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

BMSID (10místné celé číslo se znaménkem)

Struktura popisovače vyrovnávací paměti pro zprávu-pole StrucId .

Jedná se o identifikátor struktury. Hodnota musí být:

BMSIDV
Identifikátor vyrovnávací paměti pro strukturu popisovače zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BMSIDV.

BMVER (10místné celé číslo se znaménkem)

Vyrovnávací paměť pro strukturu popisovače zprávy-pole Verze.

Toto je číslo verze struktury. Hodnota musí být:

BMVER1
Číslo verze pro vyrovnávací paměť pro strukturu popisovače zprávy.

Následující konstanta určuje číslo verze aktuální verze:

BMVERVC
Aktuální verze vyrovnávací paměti pro strukturu popisovače zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BMVER1.

BMOPT (10místné celé číslo se znaménkem)

Struktura vyrovnávací paměti pro zpracování zpráv-pole Volby.

Hodnota může být následující:

BMDLPR
Vlastnosti přidané do popisovače zprávy jsou odstraněny z vyrovnávací paměti. Pokud volání selže, nebudou odstraněny žádné vlastnosti.

Výchozí volby: Pokud nepotřebujete popsanou volbu, použijte následující volbu:

BMNONE
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BMDLPR.

Počáteční hodnoty

Tabulka 683. Pole v MQBMHO

Název pole	Název konstanty	Hodnota konstanty
BMSID	BMSIDV	'BMHO'
BMVER	BMVER1	1
BMOPT	BMNONE	0

Deklarace RPG

```
     D* MQBMHO Structure
     D*
     D*
     D* Structure identifier
     D  BMSID                  1      4    INZ('BMHO')
```

```
     D*
     D* Structure version number
     D  BMVER                  5      8I 0 INZ(1)
     D*
     D* Options that control the action of MQBUFMH
     D  BMOPT                  9     12I 0 INZ(1)
```

IBM i MQBO (začátek voleb) na IBM i

Struktura MQBO umožňuje aplikaci určit volby související s vytvořením pracovní jednotky.

Přehled

Účel: Struktura je vstupní/výstupní parametr volání MQBEGIN.

Znaková sada a kódování: Data v obchodním objektu MQBO musí být ve znakové sadě zadané atributem správce front **CodedCharSetId** a v kódování lokálního správce front daném ENNAT.

- "Pole" na stránce 1015
- "Počáteční hodnoty" na stránce 1016
- "Deklarace RPG" na stránce 1016

Pole

Struktura MQBO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

BOOPT (10místné celé číslo se znaménkem)

Volby, které řídí akci MQBEGIN.

Hodnota musí být:

BONONE
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BONONE.

BOSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

BOSIDV
Identifikátor pro strukturu voleb začátku.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BOSIDV.

BOVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

BOVER1
Číslo verze pro strukturu voleb začátku.

Následující konstanta určuje číslo verze aktuální verze:

Přehled
Aktuální verze struktury voleb začátku.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BOVER1.

Počáteční hodnoty

Tabulka 684. Pole v objektu MQBO		
Název pole	**Název konstanty**	**Hodnota konstanty**
BOSID	BOSIDV	'BO¬¬'
BOVER	BOVER1	1
BOOPT	BONONE	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQBO Structure
D*
D* Structure identifier
D  BOSID                 1      4    INZ('BO  ')
D* Structure version number
D  BOVER                 5      8I 0 INZ(1)
D* Options that control the action of MQBEGIN
D  BOOPT                 9     12I 0 INZ(0)
```

IBM i MQCBC (kontext zpětného volání) na systému IBM i

Struktura popisující rutinu zpětného volání.

Přehled

Účel

Struktura MQCBC se používá k určení informací o kontextu, které se předávají funkci zpětného volání.

Struktura je vstupní/výstupní parametr volání rutiny spotřebitele zpráv.

Verze

Aktuální verze MQCBC je CBCV2.

Znaková sada a kódování

Data v MQCBC jsou ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , je struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1016
- "Počáteční hodnoty" na stránce 1022
- "Deklarace RPG" na stránce 1022

Pole

Struktura MQCBC obsahuje následující pole; pole jsou popsána v abecedním pořadí:

CBCBUFFLEN (10místné celé číslo se znaménkem)

Vyrovnávací paměť může být větší než hodnota MaxMsgLength definovaná pro spotřebitele a hodnota ReturnedLength v MQGMO.

Struktura kontextu zpětného volání-pole BufferLength .

Jedná se o délku vyrovnávací paměti zpráv v bajtech, která byla předána této funkci.

Skutečná délka zprávy je uvedena v poli DataLength .

Aplikace může používat celou vyrovnávací paměť pro své vlastní účely po dobu trvání funkce zpětného volání.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny výjimek.

CBCCALLBA (10místné celé číslo se znaménkem)

Struktura kontextu zpětného volání-pole CallbackArea .

Jedná se o pole, které je k dispozici pro použití funkcí zpětného volání.

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a je předán beze změny z pole CBDCALLBA ve struktuře MQCBD, což je parametr volání MQCB použitý k definování funkce zpětného volání.

Změny v souboru *CBCCALLBA* jsou zachovány v rámci vyvolání funkce zpětného volání pro položku *CBCHOBJ*. Toto pole není sdíleno s funkcemi zpětného volání pro jiné manipulátory.

Toto je vstupní/výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

CBCCALLT (celé číslo se znaménkem 10 číslic)

Struktura kontextu zpětného volání-pole CallType .

Pole obsahující informace o tom, proč byla tato funkce volána. Jsou definovány následující typy volání.

Typy volání doručení zpráv: Tyto typy volání obsahují informace o zprávě. Parametry **CBCLEN** a **CBCBUFFLEN** jsou platné pro tyto typy volání.

CBCTMR
Funkce spotřebitele zpráv byla vyvolána se zprávou, která byla destruktivně odebrána z popisovače objektu.

Je-li hodnota *CBCCC* CCWARN, hodnota pole *Reason* je RC2079 nebo jeden z kódů, které označují problém s převodem dat.

CBCTMN
Funkce spotřebitele zpráv byla vyvolána se zprávou, která ještě nebyla destruktivně odebrána z popisovače objektu. Zprávu lze destruktivně odebrat z popisovače objektu pomocí konzoly *MsgToken*.

Zpráva možná nebyla odebrána, protože:

- Volby MQGMO požadovali operaci procházení, GMBR*
- Zpráva je větší než dostupná vyrovnávací paměť a volby MQGMO neuvádějí gmatm

Je-li hodnota *CBCCC* CCWARN, hodnota pole *Reason* je RC2080 nebo jeden z kódů, které označují problém s převodem dat.

Typy volání řízení zpětného volání: Tyto typy volání obsahují informace o ovládacím prvku zpětného volání a neobsahují podrobnosti o zprávě. Tyto typy volání jsou požadovány pomocí CBDOPT ve struktuře MQCBD.

Parametry **CBCLEN** a **CBCBUFFLEN** nejsou pro tyto typy volání platné.

CBCTRC
Účelem tohoto typu volání je umožnit funkci zpětného volání provést počáteční nastavení.

Funkce zpětného volání je vyvolána okamžitě po registraci zpětného volání, tj. po návratu z volání MQCB pomocí hodnoty pole *Operation* CBREG.

Tento typ volání se používá jak pro spotřebitele zpráv, tak pro obslužné rutiny událostí.

Je-li požadováno, jedná se o první vyvolání funkce zpětného volání.

Hodnota pole *CBCREA* je RCNONE.

CBCTSC
Účelem tohoto typu volání je umožnit funkci zpětného volání provést při spuštění určité nastavení, například obnovení prostředků, které byly vyčištěny při předchozím zastavení.

Funkce zpětného volání je vyvolána, když je připojení spuštěno buď pomocí CTLSR, nebo CTLSW.

Je-li funkce zpětného volání registrována v rámci jiné funkce zpětného volání, je tento typ volání vyvolán při vrácení zpětného volání.

Tento typ volání se používá pouze pro spotřebitele zpráv.

Hodnota pole *CBCREA* je RCNONE.

CBCTTC
Účelem tohoto typu volání je umožnit funkci zpětného volání provést určité vyčištění, když je na chvíli zastavena, například vyčistit další prostředky, které byly získány během příjmu zpráv.

Funkce zpětného volání je vyvolána, když je vydáno volání MQCTL s použitím hodnoty pro pole *Operation* v CTLSP.

Tento typ volání se používá pouze pro spotřebitele zpráv.

Hodnota pole *CBCREA* je nastavena tak, aby označovala příčinu zastavení.

CBCTDC
Účelem tohoto typu volání je umožnit funkci zpětného volání provést konečné vyčištění na konci procesu spotřeby. Funkce zpětného volání je vyvolána při:

- Registrace funkce zpětného volání je zrušena pomocí volání MQCB s BCUNR.
- Fronta je zavřená, což způsobuje implicitní zrušení registrace. V této instanci je funkce zpětného volání předána HOUNUH jako popisovač objektu.
- Volání MQDISC je dokončeno, což způsobí implicitní zavření, a tedy zrušení registrace. V tomto případě není připojení okamžitě odpojeno a žádná probíhající transakce není dosud potvrzena.

Pokud jsou některé z těchto akcí provedeny uvnitř samotné funkce zpětného volání, akce se vyvolá, jakmile se zpětné volání vrátí.

Tento typ volání se používá jak pro spotřebitele zpráv, tak pro obslužné rutiny událostí.

Je-li požadováno, jedná se o poslední vyvolání funkce zpětného volání.

Hodnota pole *CBCREA* je nastavena tak, aby označovala příčinu zastavení.

CBCTEC
Funkce obslužné rutiny událostí

Funkce obslužné rutiny událostí byla vyvolána bez zprávy, když:

- Je vydáno volání MQCTL s hodnotou pole *Operation* CTLSP, nebo
- Správce front nebo připojení se zastaví nebo uvede do klidového stavu.

Toto volání lze použít k provedení příslušné akce pro všechny funkce zpětného volání.

- **Funkce spotřebitele zpráv**

  Funkce spotřebitele zpráv byla vyvolána bez zprávy, když byla zjištěna chyba (*CBCCC* = CCFAIL), která je specifická pro popisovač objektu; například *CBCREA* code = RC2016 .

Hodnota pole *CBCREA* je nastavena tak, aby označovala příčinu volání.

Toto je vstupní pole. CBCTMR a CMCTMN jsou použitelné pouze pro funkce spotřebitele zpráv.

CBCCC (10místné celé číslo se znaménkem)

Struktura kontextu zpětného volání-pole CompCode .

Toto je kód dokončení. Označuje, zda došlo k problémům při přijímání zprávy; jedná se o jednu z následujících možností:

CCOK
Úspěšné dokončení

CCWARN (varování)
Varování (částečné dokončení)

CCFAIL
Volání selhalo

Toto je vstupní pole. Počáteční hodnota tohoto pole je CCOK.

CBCCONNAREA (10místné celé číslo se znaménkem)

Struktura kontextu zpětného volání-pole ConnectionArea .

Jedná se o pole, které je k dispozici pro použití funkcí zpětného volání.

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a předává se beze změny z pole ConnectionArea ve struktuře MQCTLO, což je parametr volání MQCTL použitého k řízení funkce zpětného volání.

Veškeré změny provedené v tomto poli funkcemi zpětného volání jsou zachovány v rámci vyvolání funkce zpětného volání. Tuto oblast lze použít k předávání informací, které mají být sdíleny všemi funkcemi zpětného volání. Na rozdíl od *CallbackArea*je tato oblast společná pro všechna zpětná volání pro manipulátor připojení.

Toto je vstupní a výstupní pole. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

CBCLEN (10místné celé číslo se znaménkem)

Jedná se o délku dat aplikace ve zprávě v bajtech. Je-li hodnota nula, znamená to, že zpráva neobsahuje žádná data aplikace.

Pole CBCLEN obsahuje délku zprávy, ale ne nutně délku dat zprávy předaných spotřebiteli. Je možné, že zpráva byla oříznuta. Pomocí pole GMRL v MQGMO určete, kolik dat bylo předáno spotřebiteli.

Pokud kód příčiny označuje, že zpráva byla oříznuta, můžete pomocí pole CBCLEN určit, jak velká je skutečná zpráva. To vám umožňuje určit velikost vyrovnávací paměti požadované pro uložení dat zprávy a poté zadat volání MQCB pro aktualizaci CBDMML v MQCBD s odpovídající hodnotou.

Je-li zadána volba GMCONV, převedená zpráva může být větší než hodnota vrácená pro DataLength. V takových případech aplikace pravděpodobně potřebuje zadat volání MQCB, aby aktualizovala CBDMML v MQCBD tak, aby byla větší než hodnota vrácená správcem front pro DataLength.

Chcete-li se vyvarovat problémům s oříznutím zprávy, zadejte MaxMsgLength jako CBDFM. To způsobí, že správce front přidělí vyrovnávací paměť pro celou délku zprávy po převodu dat. Uvědomte si však, že i když je tato volba uvedena, je stále možné, že není k dispozici dostatek paměti pro správné zpracování požadavku. Aplikace by měly vždy zkontrolovat vrácený kód příčiny. Pokud například není možné přidělit dostatek paměti pro převod zprávy, zprávy se vrátí do aplikace nepřevedené.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny událostí.

CBCFLG (10místné celé číslo se znaménkem)

Příznaky obsahující informace o tomto spotřebiteli.

Je definována následující volba:

CBCFBE
Tento příznak může být vrácen, pokud předchozí volání MQCLOSE pomocí volby COQSC selhalo s kódem příčiny RC2458.

Tento kód označuje, že je vrácena poslední dopředná zpráva a že vyrovnávací paměť je nyní prázdná. Pokud aplikace vydá další volání MQCLOSE pomocí volby COQSC, bude úspěšná.

Všimněte si, že aplikaci není zaručeno, že obdrží zprávu s touto sadou příznaků, protože stále mohou existovat zprávy ve vyrovnávací paměti pro čtení napřed, které neodpovídají aktuálním kritériím výběru. V této instanci je funkce spotřebitele vyvolána s kódem příčiny RC2019 .

Je-li vyrovnávací paměť dopředného čtení prázdná, je spotřebitel vyvolán s příznakem CBCFBE a kódem příčiny RC2518.

Toto je vstupní pole pro funkci spotřebitele zpráv; není relevantní pro funkci obslužné rutiny událostí.

CBCHOBJ (10místné celé číslo se znaménkem)

Struktura kontextu zpětného volání-pole CBCHOBJ.

Pro volání spotřebitele zpráv se jedná o manipulátor pro objekt související se spotřebitelem zpráv.

Pro obslužnou rutinu událostí je tato hodnota HONONE

Aplikace může použít tento popisovač a token zprávy v bloku Volby získání zprávy k získání zprávy, pokud zpráva nebyla odebrána z fronty.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je HOUNUH.

CBCRCD (10místné celé číslo se znaménkem)

CBCRCD označuje, jak dlouho správce front čeká, než se znovu pokusí o připojení. Pole může být upraveno obslužnou rutinou událostí, aby se zcela změnila prodleva nebo zastavilo opětovné připojení.

Pole **CBCRCD** použijte pouze v případě, že hodnota pole **Reason** v kontextu zpětného volání je RC2545.

Při vstupu do obslužné rutiny událostí je hodnota **CBCRCD** počet milisekund, po které bude správce front čekat před provedením pokusu o opětovné připojení. Tabulka 685 na stránce 1020 obsahuje seznam hodnot, které lze nastavit pro úpravu chování správce front při návratu z obslužné rutiny událostí.

*Tabulka 685. **CBCRCD** hodnoty*

Hodnota	Popis
-1	Neprovádět žádné další pokusy o opětovné připojení. Aplikaci se vrátí chyba.
0	Pokuste se okamžitě znovu připojit.
>0	Počkejte na tento počet milisekund, než se znovu pokusíte o připojení.

CBCREA (10místné celé číslo se znaménkem)

Struktura kontextu zpětného volání-pole příčiny.

Toto je kód příčiny, který kvalifikuje *CBCCC*

Toto je vstupní pole. Počáteční hodnota tohoto pole je RCNONE.

CBCSTATE (10místné celé číslo se znaménkem)

Indikace stavu aktuálního spotřebitele. Toto pole má největší hodnotu pro aplikaci, když je funkci spotřebitele předán nenulový kód příčiny.

Toto pole můžete použít ke zjednodušení programování aplikací, protože nemusíte kódovat chování pro každý kód příčiny.

Toto je vstupní pole. Počáteční hodnota tohoto pole je CSNONE.

Tabulka 686. Hodnoty CBCSTATE a výsledné akce		
Stav	**Akce správce front**	**Hodnota konstanty**
CSNONE Tento kód příčiny představuje normální volání bez dalších informací o příčině.	Není; jedná se o normální operaci.	0
CSSUST Tyto kódy příčiny představují dočasné podmínky.	Rutina zpětného volání je volána k ohlášení podmínky a poté pozastavena. Po určité době se systém může pokusit o operaci znovu, což může vést k opětovnému vytvoření stejné podmínky.	1
CSSUSU Tyto kódy příčiny představují podmínky, kdy musí zpětné volání jednat, aby vyřešil podmínku.	Odběratel je pozastaven a je volána rutina zpětného volání pro ohlášení podmínky. Rutina zpětného volání by měla vyřešit podmínku, je-li to možné, a buď POKRAČOVAT, nebo ukončit připojení.	2
CSSUS Tyto kódy příčiny představují selhání, která zabraňují dalším zpětným voláním zpráv.	Správce front automaticky pozastaví funkci zpětného volání. Je-li funkce zpětného volání obnovena, je pravděpodobné, že znovu obdrží stejný kód příčiny.	3
CSSTOP Tyto kódy příčiny představují konec spotřeby zpráv.	Doručeno obslužné rutině výjimek a zpětným voláním, která určila CBDTC. Nelze přijímat žádné další zprávy.	4

CBCSID (10místné celé číslo se znaménkem)

Struktura kontextu zpětného volání-pole StrucId .

Toto je identifikátor struktury; hodnota musí být:

CBCSI
Identifikátor pro strukturu kontextu zpětného volání.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CBCSI.

CBCVER (10místné celé číslo se znaménkem)

Struktura kontextu zpětného volání-pole Verze.

Toto je číslo verze struktury; hodnota musí být:

CBCV1
Struktura kontextu zpětného volání Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

CBCCV
Aktuální verze struktury kontextu zpětného volání.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CBCV1.

Počáteční hodnoty

Tabulka 687. Pole v MQCBC		
Název pole	**Název konstanty**	**Hodnota konstanty**
CBCSID	CBCSI	'CBC¬'
CBCVER	CBCV1	1
CBCCALLT	Není	0
CBCHOBJ	HOUNUH	-1
CBCCALLBA	Není	Ukazatel Null nebo bajty s hodnotou Null
CBCCONNAREA	Není	Ukazatel Null nebo bajty s hodnotou Null
CBCCC	CCOK	0
CBCREA	RCNONE	0
CBCSTATE	CSNONE	0
CBCLEN	Není	0
CBCBUFFLEN	Není	0
CBCFLG	Není	0
CBCRCD	Není	0

Poznámka:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D* MQCBC Structure
D*
D*
D* Structure identifier
D  CBCSID                 1      4    INZ('CBC ')
D*
D* Structure version number
D  CBCVER                 5      8I 0 INZ(1)
D*
D* Why Function was called
D  CBCCALLT               9     12I 0 INZ(0)
D*
D* Object Handle
D  CBCHOBJ               13     16I 0 INZ(-1)
D*
D* Callback data passed to the function
D  CBCCALLBA             17     32*   INZ(*NULL)
D*
D* MQCTL Data area passed to the function
D  CBCCONNAREA           33     48*   INZ(*NULL)
D*
D* Completion Code
D  CBCCC                 49     52I 0 INZ(0)
D*
D* Reason Code
D  CBCREA                53     56I 0 INZ(0)
D*
D* Consumer State
D  CBCSTATE              57     60I 0 INZ(0)
D*
D* Message Data Length
D  CBCLEN                61     64I 0 INZ(0)
```

```
D*
D* Buffer Length
D  CBCBUFFLEN            65     68I 0 INZ(0)
D*
** Flags containing information about
D* this consumer
D  CBCFLG                69     72I 0 INZ(0)
D* Ver:1 **
D* Number of milliseconds before reconnect attempt
D  CBCRCD                73     76I 0 INZ(0)
D* Ver:2 **
D*
```

IBM i MQCBD (deskriptor zpětného volání) na systému IBM i

Struktura určující funkci zpětného volání.

Přehled

Účel: Struktura MQCBD se používá k určení funkce zpětného volání a voleb, které řídí její použití správcem front.

Struktura je vstupní parametr volání MQCB.

Verze: Aktuální verze MQCBD je CBDV1.

Znaková sada a kódování: Data v MQCBD musí být ve znakové sadě a kódování lokálního správce front; ty jsou dány atributem správce front **CodedCharSetId** a ENNAT. Pokud je však aplikace spuštěna jako IBM MQ MQI client, musí být struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1023
- "Počáteční hodnoty" na stránce 1027
- "Deklarace RPG" na stránce 1027

Pole

Struktura MQCBD obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CBDCALLBA (10místné celé číslo se znaménkem)

Jedná se o pole, které je k dispozici pro použití funkcí zpětného volání.

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a předává se beze změny z pole CBCCALLBA ve struktuře MQCBD, což je parametr deklarace funkce zpětného volání.

Hodnota se používá pouze na systému *Operation* , který má hodnotu CBREG, bez aktuálně definovaného zpětného volání, nenahradí předchozí definici.

Toto je vstupní a výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

CBDCALLBF (10místné celé číslo se znaménkem)

Funkce zpětného volání je vyvolána jako volání funkce.

Toto pole slouží k určení ukazatele na funkci zpětného volání.

Musíte zadat buď *CallbackFunction* , nebo *CallbackName*. Zadáte-li obojí, vrátí se kód příčiny RC2486 .

Není-li *CallbackName* ani *CallbackFunction* nastaveno, volání selže s kódem příčiny RC2486.

Tato volba není podporována v následujících prostředích:

- CICS na platformě z/OS
- Programovací jazyky a kompilátory, které nepodporují odkazy ukazatele funkce

V takových situacích volání selže s kódem příčiny RC2486.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

CBDCALLBN (10místné celé číslo se znaménkem)

Funkce zpětného volání je vyvolána jako dynamicky propojený program.

Musíte zadat buď *CallbackFunction* , nebo *CallbackName*. Zadáte-li obojí, vrátí se kód příčiny RC2486 .

Pokud buď *CallbackName* , nebo *CallbackFunction* není true, volání selže s kódem příčiny RC2486.

Modul je načten při registraci první rutiny zpětného volání, která má být použita, a uvolněn při zrušení registrace poslední rutiny zpětného volání.

Kromě toho, kde je uvedeno v následujícím textu, je název zarovnán vlevo uvnitř pole bez vložených mezer; samotný název je vyplněn mezerami na délku pole. V popisech, které následují, hranaté závorky ([]) označují nepovinné informace:

IBMi

Název zpětného volání může mít jeden z následujících formátů:

- Knihovna "/" Program
- Knihovna "/" ServiceProgram "("FunctionName")"

Například `MyLibrary/MyProgram(MyFunction)`.

Název knihovny může být *LIBL. Názvy knihoven i programů jsou omezeny na maximálně 10 znaků.

AIX and Linux

Název zpětného volání je název dynamicky zaveditelného modulu nebo knihovny s příponou názvu funkce nacházející se v této knihovně. Název funkce musí být uzavřen v závorkách. Název knihovny může mít volitelně předponu s cestou k adresáři:

```
[path]library(function)
```

Není-li cesta uvedena, použije se systémová vyhledávací cesta.

Délka názvu je omezena na maximálně 128 znaků.

Windows

Název zpětného volání je název knihovny dynamického propojení doplněný názvem funkce nacházející se v této knihovně. Název funkce musí být uzavřen v závorkách. Název knihovny může mít volitelně předponu s cestou k adresáři a jednotkou:

```
[d:][path]library(function)
```

Pokud jednotka a cesta nejsou uvedeny, použije se systémová vyhledávací cesta.

Délka názvu je omezena na maximálně 128 znaků.

z/OS

Název zpětného volání je název zaváděcího modulu, který je platný pro specifikaci parametru EP makra LINK nebo LOAD.

Délka názvu je omezena na maximálně 8 znaků.

z/OS CICS

Název zpětného volání je název zaváděcího modulu, který je platný pro specifikaci parametru PROGRAM makra příkazu EXEC CICS LINK.

Délka názvu je omezena na maximálně 8 znaků.

Program lze definovat jako vzdálený pomocí volby REMOTESYTEM nainstalované definice PROGRAM nebo pomocí dynamického směrovacího programu.

Vzdálená oblast CICS musí být připojena k produktu IBM MQ , pokud má program používat volání rozhraní API IBM MQ . Všimněte si však, že pole CBCHOBJ ve struktuře MQCBC není ve vzdáleném systému platné.

Dojde-li při pokusu o načtení souboru *CallbackName*k selhání, vrátí se aplikaci jeden z následujících kódů chyb:

- RC2495
- RC2496
- RC2497

Do protokolu chyb se také zapíše zpráva obsahující název modulu, pro který byl proveden pokus o načtení, a kód příčiny selhání z operačního systému.

Toto je vstupní pole. Počáteční hodnota tohoto pole je prázdný řetězec nebo mezery.

CBDCALLBT (10místné celé číslo se znaménkem)

Jedná se o typ funkce zpětného volání. Hodnota musí být jedna z následujících:

CBTMC
Definuje toto zpětné volání jako funkci spotřebitele zpráv.

Funkce zpětného volání spotřebitele zpráv je volána, když je zpráva splňující zadaná kritéria výběru k dispozici na popisovači objektu a připojení je spuštěno.

CBTEH
Definuje toto zpětné volání jako rutinu asynchronních událostí; není řízeno tak, aby spotřebovávalo zprávy pro manipulátor.

Ve volání MQCB, které definuje obslužnou rutinu událostí, není hodnota *Hobj* vyžadována a je-li zadána, je ignorována.

Obslužná rutina událostí je volána pro podmínky, které ovlivňují celé prostředí spotřebitele zpráv. Funkce spotřebitele je vyvolána bez zprávy, dojde-li k události, například k zastavení správce front nebo připojení, nebo k uvedení do klidového stavu. Není volán pro podmínky, které jsou specifické pro jednoho spotřebitele zpráv, například RC2016.

Události jsou doručeny do aplikace bez ohledu na to, zda je připojení spuštěno nebo zastaveno, s výjimkou následujících prostředí:

- CICS v prostředí z/OS
- aplikace bez podprocesů

Pokud volající nepředá jednu z těchto hodnot, volání selže s kódem příčiny RC2483

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CBTMC.

CBDMML (10místné celé číslo se znaménkem)

Jedná se o délku nejdelší zprávy v bajtech, kterou lze načíst z manipulátoru a předat rutině zpětného volání. Pokud má zpráva delší délku, rutina zpětného volání obdrží *MaxMsgLength* bajtů zprávy a kód příčiny:

- RC2080 nebo
- RC2079 , pokud jste zadali GMATM.

Skutečná délka zprávy je zadána v poli “CBCLEN (10místné celé číslo se znaménkem)” na stránce 1019 struktury MQCBC.

Je definována následující speciální hodnota:

CBDFM
Délka vyrovnávací paměti je systémem upravena tak, aby vracela zprávy bez oříznutí.

Pokud není k dispozici dostatek paměti pro přidělení vyrovnávací paměti pro přijetí zprávy, systém zavolá funkci zpětného volání s kódem příčiny RC2071 .

Pokud například požadujete převod dat a není k dispozici dostatek paměti pro převod dat zprávy, nepřevedená zpráva se předá do funkce zpětného volání.

Toto je vstupní pole. Počáteční hodnota pole *MaxMsgLength* je CBDFM.

CBDOPT (10místné celé číslo se znaménkem)

Struktura deskriptoru zpětného volání-pole Volby.

Lze zadat libovolnou z následujících možností. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace). Kombinace, které nejsou platné, jsou uvedeny; všechny ostatní kombinace jsou platné.

CBDFQ
Volání MQCB se nezdaří, pokud se správce front nachází ve stavu uvedení do klidového stavu.

V systému z/OStato volba také vynutí selhání volání MQCB, pokud je připojení (pro aplikaci CICS nebo IMS) v klidovém stavu.

Zadejte GMFIQ ve volbách MQGMO předaných na volání MQCB, abyste způsobili oznámení spotřebitelům zpráv při jejich uvedení do klidového stavu.

Volby řízení: Následující volby řídí, zda je volána funkce zpětného volání bez zprávy, když se změní stav spotřebitele:

CBDRC
Funkce zpětného volání je vyvolána s typem volání CBCTRC

.

CBDSC
Funkce zpětného volání je vyvolána s typem volání CBCTSC.

CBDTC
Funkce zpětného volání je vyvolána s typem volání CBCTTC.

CBDDC
Funkce zpětného volání je vyvolána s typem volání CBCTDC.

Další podrobnosti o těchto typech volání viz “CBCCALLT (celé číslo se znaménkem 10 číslic)” na stránce 1017 .

Výchozí volba: Pokud nepotřebujete žádnou z popsaných voleb, použijte následující volbu:

CBDNO
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

CBDNO je definován jako pomůcka pro dokumentaci programu. Není zamýšleno, aby byla tato volba použita s jinou, ale protože je její hodnota nula, nelze takové použití zjistit.

Toto je vstupní pole. Počáteční hodnota pole *Options* je CBDNO.

CBDSID (10místné celé číslo se znaménkem)

Struktura deskriptoru zpětného volání-pole StrucId .

Toto je identifikátor struktury; hodnota musí být:

CBDSI
Identifikátor pro strukturu deskriptoru zpětného volání.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CBDSI.

CBDVER (10místné celé číslo se znaménkem)

Struktura deskriptoru zpětného volání-pole Verze.

Toto je číslo verze struktury; hodnota musí být:

CBDV1
Version-1 struktura deskriptoru zpětného volání.

Následující konstanta určuje číslo verze aktuální verze:

CBDCV
Aktuální verze struktury deskriptoru zpětného volání.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CBDV1.

Počáteční hodnoty

Tabulka 688. Pole v MQCBD

Název pole	Název konstanty	Hodnota konstanty
StrucId	CBDSI	'CBD¬'
Version	CBDV1	1
CallBackType	CBTMC	1
Options	CBDNO	0
CallbackArea	Není	Počet bajtů s hodnotou null
CallbackFunction	Není	Počet bajtů s hodnotou null
CallbackName	Není	Mezery
MaxMsgLength	CBDFM	-1

Poznámka:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D* MQCBD Structure
D*
D*
D* Structure identifier
D  CBDSID                 1      4    INZ('CBD ')
D*
D* Structure version number
D  CBDVER                 5      8I 0 INZ(1)
D*
D* Callback function type
D  CBDCALLBT              9     12I 0 INZ(1)
D*
** Options controlling message
D* consumption
D  CBDOPT                13     16I 0 INZ(0)
D*
D* User data passed to the function
D  CBDCALLBA             17     32*
D*
D* FP: Callback function pointer
D  CBDCALLBF             33     48*
D*
D* Callback name
D  CBDCALLBN             49    176    INZ('\0')
D*
D* Maximum message length
D  CBDMML               177    180I 0 INZ(-1)
```

IBM i MQCHARV (Variable Length String) na systému IBM i

Použijte strukturu MQCHARV k popisu řetězce proměnné délky.

Přehled

Znaková sada a kódování: Data v MQCHARV musí být v kódování lokálního správce front, které je dáno ENNAT a znakovou sadou pole VCHRC v rámci struktury. Pokud je aplikace spuštěna jako IBM MQ MQI client, struktura musí být v kódování klienta. Některé znakové sady mají reprezentaci, která závisí na kódování. Pokud je jednou z těchto znakových sad VCHRC, použité kódování je stejné jako kódování ostatních polí v MQCHARV. Znaková sada identifikovaná pomocí VSCCSID může být dvoubajtová znaková sada (DBCS).

Použití: Struktura MQCHARV adresuje data, která mohou být nesouvislá se strukturou, která ji obsahuje. Pro adresování těchto dat lze použít pole deklarovaná s datovým typem ukazatele.

- "Pole" na stránce 1028
- "Počáteční hodnoty" na stránce 1029
- "Deklarace RPG" na stránce 1029
- "Změna definice CSAPL" na stránce 1030

Pole

Struktura MQCHARV obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

VCHRC (10místné celé číslo se znaménkem)

Toto je identifikátor znakové sady řetězce proměnné délky adresovaného polem VCHRP nebo VCHRO.

Počáteční hodnota tohoto pole je CSAPL. Tento parametr je definován produktem IBM MQ , aby indikoval, že by měl být změněn správcem front na skutečný identifikátor znakové sady správce front. Je to stejné, jako se chová CSQM. Výsledkem je, že hodnota CSAPL není nikdy přidružena k řetězci s proměnnou délkou. Počáteční hodnotu tohoto pole lze změnit definováním jiné hodnoty pro konstantní CSAPL pro vaši kompilační jednotku pomocí vhodných prostředků pro programovací jazyk vaší aplikace.

VCHRL (10místné celé číslo se znaménkem)

Délka řetězce proměnné délky v bajtech adresovaná polem VCHRP nebo VCHRO.

Počáteční hodnota tohoto pole je 0. Hodnota musí být větší nebo rovna nule nebo následující speciální hodnota, která je rozpoznána:

VSNLT

Není-li VSNLT uveden, jsou bajty VCHRL zahrnuty jako součást řetězce. Jsou-li přítomny znaky null, neoddělují řetězec.

Je-li zadána hodnota VSNLT, je řetězec oddělen první hodnotou null zjištěnou v řetězci. Hodnota null sama o sobě není zahrnuta jako součást tohoto řetězce.

Poznámka: Znak null použitý k ukončení řetězce, pokud je uveden VSNLT, je null z kódové sady uvedené VCHRC.

Například v UTF-16 (CCSID 1200, 13488 a 17584) se jedná o dvoubajtové kódování Unicode, kde je hodnota null reprezentována 16bitovým číslem všech nul. V souboru UTF-16 je běžné najít jednotlivé bajty nastavené na nulu, které jsou součástí znaků (například 7bitové znaky ASCII), ale řetězce budou ukončeny hodnotou null pouze v případě, že jsou na sudé hranici bajtů nalezeny dva bajty 'nula'. Je možné získat dva 'nula' bajty na liché hranici, když jsou každou částí platných znaků. Například x '01' x '00' x '00' x 'x' 30 ' představuje dva platné znaky Unicode a neukončuje řetězec hodnotou null.

VCHRO (10místné celé číslo se znaménkem)

Posun v bajtech řetězce proměnné délky od začátku MQCHARV nebo struktury, která jej obsahuje.

Je-li struktura MQCHARV vložena do jiné struktury, je tato hodnota posunutím řetězce proměnné délky od začátku struktury, která obsahuje tuto strukturu MQCHARV, v bajtech. Není-li struktura MQCHARV vložena do jiné struktury, například je-li zadána jako parametr pro volání funkce, je posun relativní ke začátku struktury MQCHARV.

Posun může být kladný nebo záporný. Můžete použít buď pole VCHRP, nebo VCHRO, abyste uvedli řetězec proměnné délky, ale ne obojí.

Počáteční hodnota tohoto pole je 0.

VCHRP (ukazatel)

Jedná se o ukazatel na řetězec proměnné délky.

Můžete použít buď pole VCHRP, nebo VCHRO, abyste uvedli řetězec proměnné délky, ale ne obojí.

Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

VCHRS (10místné celé číslo se znaménkem)

Velikost vyrovnávací paměti adresované polem VCHRP nebo VCHRO v bajtech.

Je-li struktura MQCHARV použita jako výstupní pole pro volání funkce, musí být toto pole inicializováno s poskytnutou délkou vyrovnávací paměti. Pokud je hodnota VCHRL větší než VCHRS, pak budou volajícímu ve vyrovnávací paměti vráceny pouze datové bajty VCHRS.

Hodnota musí být větší nebo rovna nule nebo následující speciální hodnota, která je rozpoznána:

VSUSL
Je-li zadána hodnota VSUSL, bude délka vyrovnávací paměti převzata z pole VCHRL ve struktuře MQCHARV. Tato speciální hodnota není vhodná, když je struktura použita jako výstupní pole a je poskytnuta vyrovnávací paměť. Toto je počáteční hodnota tohoto pole.

Počáteční hodnoty

Tabulka 689. Počáteční hodnoty MQCHARV pro konstanty

Název pole	Název konstanty	Hodnota konstanty
VCHRP	Není	Nulový ukazatel nebo nulový počet bajtů.
VCHRO	Není	0
VCHRS	VSUSL	-1
VCHRL	Není	0
VCHRC	CSAPL	-3

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQCHARV Structure
D*
D* Address of variable length string
D VCHRP                   1     16*
D* Offset of variable length string
D VCHRO                  17     20I 0
D* Size of buffer
D VCHRS                  21     24I 0
D* Length of variable length string
D VCHRL                  25     28I 0
```

```
     D* CCSID of variable length string
     D VCHRC                 29     32I 0
```

Změna definice CSAPL

Na rozdíl od programovacích jazyků podporovaných na jiných platformách, RPG nemá způsob, jak předefinovat definovanou konstantu, takže musíte nastavit každou VCHRC specificky, pokud chcete použít jinou hodnotu než CSAPL.

IBM i MQCIH (záhlavíCICS bridge) na IBM i

Struktura MQCIH popisuje informace, které mohou být přítomny na začátku zprávy odeslané do CICS bridge přes IBM MQ for z/OS.

Přehled

Název formátu: FMCICS.

Verze: Aktuální verze MQCIH je CIVER2. Pole, která existují pouze v novější verzi struktury, jsou jako taková identifikována v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi MQCIH s počáteční hodnotou pole *CIVER* nastavenou na hodnotu CIVER2.

Znaková sada a kódování: Speciální podmínky se vztahují na znakovou sadu a kódování použité pro strukturu MQCIH a data zpráv aplikace:

- Aplikace, které se připojují ke správci front vlastnícímu frontu CICS bridge , musí poskytovat strukturu MQCIH, která je ve znakové sadě a kódování správce front. Důvodem je skutečnost, že v tomto případě není proveden převod dat struktury MQCIH.
- Aplikace, které se připojují k jiným správcům front, mohou poskytovat strukturu MQCIH, která je v libovolné z podporovaných znakových sad a kódování; převod MQCIH provádí přijímající agent kanálu zpráv připojený ke správci front, který vlastní frontu CICS bridge .

  Poznámka: Existuje jedna výjimka. Pokud správce front, který vlastní frontu CICS bridge , používá CICS pro distribuované řazení do front, musí být MQCIH ve znakové sadě a kódování správce front, který vlastní frontu CICS bridge .
- Data zprávy aplikace následující po struktuře MQCIH musí být ve stejné znakové sadě a kódování jako struktura MQCIH. Pole *CICSI* a *CIENC* ve struktuře MQCIH nelze použít k určení znakové sady a kódování dat zprávy aplikace.

  Uživatelská procedura převodu dat musí být poskytnuta uživatelem pro převod dat zprávy aplikace, pokud data nejsou jedním z vestavěných formátů podporovaných správcem front.

Použití: Pokud jsou hodnoty požadované aplikací stejné jako počáteční hodnoty zobrazené v souboru Tabulka 691 na stránce 1039a most je spuštěn s AUTH=LOCAL nebo AUTH=IDENTIFY, lze strukturu MQCIH ze zprávy vynechat. Ve všech ostatních případech musí být struktura přítomna.

Most přijímá strukturu MQCIH version-1 nebo version-2 , ale pro transakce 3270 musí být použita struktura version-2 .

Aplikace musí zajistit, aby pole dokumentovaná jako pole "požadavek" měla ve zprávě odeslané do mostu odpovídající hodnoty; tato pole jsou vstupem pro most.

Pole dokumentovaná jako pole "response" jsou nastavena pomocí CICS bridge ve zprávě odpovědi, kterou most odesílá aplikaci. Informace o chybě jsou vráceny v polích *CIRET*, *CIFNC*, *CICC*, *CIREA*a *CIAC* , ale ne všechny jsou nastaveny ve všech případech. Tabulka 690 na stránce 1031 ukazuje, která pole jsou nastavena pro různé hodnoty *CIRET*.

Tabulka 690. Obsah polí s informacemi o chybách ve struktuře MQCIH				
CIRET	***CIFNC***	***CICC***	***CIREA***	***CIAC***
CRC000	-	-	-	-
CRC003	-	-	Doporučení FBC*	-
CRC002 CRC008	IBM MQ název volání	IBM MQ *CMPCOD*	IBM MQ *REASON*	-
CRC001 CRC006 CRC007 CRC009	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	-
CRC004 CRC005	-	-	-	CICS ABCODE

- "Pole" na stránce 1031
- "Počáteční hodnoty" na stránce 1039
- "Deklarace RPG" na stránce 1041

Pole

Struktura MQCIH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CIAC (4bajtový znakový řetězec)

Kód nestandardního ukončení.

Hodnota vrácená v tomto poli je významná pouze v případě, že pole *CIRET* má hodnotu CRC005 nebo CRC004. Pokud ano, *CIAC* obsahuje hodnotu CICS ABCODE.

Toto je pole odezvy. Délka tohoto pole je dána hodnotou LNABNC. Počáteční hodnota tohoto pole je 4 prázdné znaky.

Jedná se o indikátor, který určuje, zda mají být deskriptory ADS odesílány na požadavky SEND a RECEIVE BMS. Jsou definovány tyto hodnoty:

ADNONE
Neodesílejte ani nepřijímejte deskriptor ADS.

ADSEND (odeslání)
Odeslat deskriptor ADS.

ADRECV
Přijmout deskriptor ADS.

ADMSGF
Použijte formát zprávy pro deskriptor ADS.

To způsobí, že deskriptor ADS bude odeslán nebo přijat pomocí dlouhého tvaru deskriptoru ADS. Dlouhý formulář má pole, která jsou zarovnána na 4bajtové hranice.

Pole *CIADS* by mělo být nastaveno takto:

- Pokud se deskriptory ADS nepoužívají, nastavte pole na ADNONE.
- Pokud se *používají* deskriptory ADS a s *stejným* CCSID v každém prostředí, nastavte pole na součet ADSEND a ADRECV.
- Pokud se *používají* deskriptory ADS, ale s *různými* identifikátory CCSID v každém prostředí, nastavte pole na součet ADSEND, ADRECV a ADMSGF.

Toto pole požadavku se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je ADNONE.

CIADS (10místné celé číslo se znaménkem)

Deskriptor ADS pro odeslání/příjem.

Jedná se o indikátor, který určuje, zda mají být deskriptory ADS odesílány na požadavky SEND a RECEIVE BMS. Jsou definovány tyto hodnoty:

ADNONE
Neodesílejte ani nepřijímejte deskriptor ADS.

ADSEND (odeslání)
Odeslat deskriptor ADS.

ADRECV
Přijmout deskriptor ADS.

ADMSGF
Použijte formát zprávy pro deskriptor ADS.

To způsobí, že deskriptor ADS bude odeslán nebo přijat pomocí dlouhého tvaru deskriptoru ADS. Dlouhý formulář má pole, která jsou zarovnána na 4bajtové hranice.

Pole *CIADS* by mělo být nastaveno takto:

- Pokud se deskriptory ADS nepoužívají, nastavte pole na ADNONE.
- Pokud se *používají* deskriptory ADS a s *stejným* CCSID v každém prostředí, nastavte pole na součet ADSEND a ADRECV.
- Pokud se *používají* deskriptory ADS, ale s *různými* identifikátory CCSID v každém prostředí, nastavte pole na součet ADSEND, ADRECV a ADMSGF.

Toto pole požadavku se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je ADNONE.

CIAI (4bajtový znakový řetězec)

Klíč AID.

Jedná se o počáteční hodnotu klíče AID při spuštění transakce. Jedná se o 1bajtovou hodnotu, zarovnanou vlevo.

Toto pole požadavku se používá pouze pro transakce 3270. Délka tohoto pole je dána hodnotou LNATID. Počáteční hodnota tohoto pole je 4 mezery.

CIAUT (8bajtový znakový řetězec)

Heslo nebo heslo.

Jedná se o heslo nebo přístupovou nálepku. Pokud je ověření identifikátoru uživatele pro CICS bridgeaktivní, *CIAUT* se použije s identifikátorem uživatele v kontextu identity MQMD k ověření odesilatele zprávy.

Toto je pole požadavku. Délka tohoto pole je dána hodnotou LNAUTH. Počáteční hodnota tohoto pole je 8 mezer.

CICC (10místné celé číslo se znaménkem)

IBM MQ kód dokončení nebo CICS EIBRESP.

Hodnota vrácená v tomto poli závisí na *CIRET* ; viz Tabulka 690 na stránce 1031.

Toto je pole odezvy. Počáteční hodnota tohoto pole je CCOK.

CICNC (4bajtový znakový řetězec)

Kód nestandardního ukončení transakce.

Jedná se o kód nestandardního ukončení, který se má použít k ukončení transakce (obvykle se jedná o konverzační transakci, která požaduje více dat). Jinak je toto pole nastaveno na mezery.

Toto pole požadavku se používá pouze pro transakce 3270. Délka tohoto pole je dána hodnotou LNCNCL. Počáteční hodnota tohoto pole je 4 mezery.

CICP (10místné celé číslo se znaménkem)

Pozice kurzoru.

Toto je počáteční pozice kurzoru při spuštění transakce. Později, pro konverzační transakce, je pozice kurzoru ve vektoru RECEIVE.

Toto pole požadavku se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0. Toto pole není přítomno, pokud je hodnota *CIVER* menší než CIVER2.

CICSI (10místné celé číslo se znaménkem)

Vyhrazeno.

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

CICT (10místné celé číslo se znaménkem)

Zda může být úloha konverzační.

Jedná se o indikátor, který určuje, zda by mělo být úloze povoleno vydávat požadavky na další informace, nebo zda by mělo dojít k nestandardnímu ukončení. Hodnota musí být jedna z následujících:

CTYES
Úloha je konverzační.

CTNO
Úloha není konverzační.

Toto pole požadavku se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je CTNO.

CIENC (10místné celé číslo se znaménkem)

Vyhrazeno.

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

CIEO (10místné celé číslo se znaménkem)

Posunutí chyby ve zprávě.

Jedná se o pozici neplatných dat zjištěných uživatelskou procedurou mostu. Toto pole poskytuje posun od začátku zprávy k umístění neplatných dat.

Toto je pole odpovědi používané pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0. Toto pole není přítomno, pokud je hodnota *CIVER* menší než CIVER2.

CIFAC (8bajtový bitový řetězec)

Token prostředku mostu.

Jedná se o 8bajtový token prostředku mostu. Účelem tokenu prostředku mostu je umožnit více transakcím v pseudokonverzaci používat stejný prostředek mostu (virtuální terminál 3270). V první nebo jediné zprávě v pseudokonverzaci by měla být nastavena hodnota FCNONE; to sděluje produktu CICS , aby pro tuto zprávu přidělil nový prostředek mostu. Token prostředku mostu je vrácen ve zprávách odezvy, je-li ve vstupní zprávě zadána nenulová hodnota *CIFKT* . Následné vstupní zprávy pak mohou používat stejný token prostředku mostu.

Je definována následující speciální hodnota:

FCNONE
Nebyl zadán žádný token BVT.

Toto je pole požadavku i pole odpovědi používané pouze pro transakce 3270. Délka tohoto pole je dána LNFAC. Počáteční hodnota tohoto pole je FCNONE.

CIFKT (10místné celé číslo se znaménkem)

Čas uvolnění zařízení mostu.

Jedná se o dobu v sekundách, po kterou bude prostředek mostu uchován po ukončení uživatelské transakce. Pro nekonverzační transakce by měla být hodnota nula.

Toto pole požadavku se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0.

CIFL (4bajtový znakový řetězec)

Atributy emulované terminálem.

Jedná se o název instalovaného terminálu, který má být použit jako model pro zařízení mostu. Hodnota mezer znamená, že hodnota *CIFL* je převzata z definice profilu transakce mostu nebo je použita výchozí hodnota.

Toto pole požadavku se používá pouze pro transakce 3270. Délka tohoto pole je dána LNFACL. Počáteční hodnota tohoto pole je 4 mezery.

CIFLG (10místné celé číslo se znaménkem)

Příznaky.

Hodnota musí být:

CIFNON
Žádné příznaky.

Toto je pole požadavku. Počáteční hodnota tohoto pole je CIFNON.

CIFMT (8bajtový znakový řetězec)

IBM MQ název formátu dat, která následují za MQCIH.

Tato volba určuje název formátu IBM MQ dat, která následují za strukturou MQCIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *MDFMT* v MQMD.

Tento název formátu se také používá pro zprávu odpovědi, pokud má pole *CIRFM* hodnotu FMNONE.

- Pro požadavky DPL musí být *CIFMT* název formátu COMMAREA.
- Pro požadavky 3270 *CIFMT* musí být CSQCBDCIa *CIRFM* musí být CSQCBDC0.

Uživatelské procedury pro převod dat pro tyto formáty musí být nainstalovány ve správci front, kde mají být spuštěny.

Pokud zpráva požadavku vyústí v generování zprávy s chybovou odpovědí, má zpráva s chybovou odpovědí název formátu FMSTR.

Toto je pole požadavku. Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

CIFNC (4bajtový znakový řetězec)

IBM MQ název volání nebo CICS funkce EIBFN.

Hodnota vrácená v tomto poli závisí na *CIRET* ; viz Tabulka 690 na stránce 1031. Následující hodnoty jsou možné, když *CIFNC* obsahuje název volání IBM MQ :

CFCONN
Volání MQCONN.

CFGET
Volání MQGET.

CFINQ (velká fronta)
Volání MQINQ.

CFOPEN
Volání MQOPEN.

CFPUT
Volání MQPUT.

CFPUT1
Volání MQPUT1 .

CFNONE
Žádné volání.

Toto je pole odezvy. Délka tohoto pole je dána hodnotou LNFUNC. Počáteční hodnota tohoto pole je CFNONE.

CIGWI (10místné celé číslo se znaménkem)

Interval čekání na volání MQGET vydané úlohou mostu.

Toto pole lze použít pouze v případě, že má parametr *CIUOW* hodnotu CUFRST. Umožňuje odesílající aplikaci určit přibližnou dobu v milisekundách, po kterou mají volání MQGET vydaná mostem čekat na druhou a následující zprávu požadavku pro pracovní jednotku spuštěnou touto zprávou. Tato volba potlačí výchozí interval čekání používaný mostem. Lze použít následující speciální hodnoty:

WIDFLT
Výchozí interval čekání.

To způsobí, že agent CICS bridge bude čekat po dobu určenou při spuštění mostu.

WIULIM
Neomezený interval čekání.

Toto je pole požadavku. Počáteční hodnota tohoto pole je WIDFLT.

CIII (10místné celé číslo se znaménkem)

Vyhrazeno.

Toto je vyhrazené pole. Hodnota musí být 0. Toto pole není přítomno, pokud je hodnota *CIVER* menší než CIVER2.

CILEN (10místné celé číslo se znaménkem)

Délka struktury MQCIH.

Hodnota musí být jedna z následujících:

CILEN1
Délka struktury záhlaví informací version-1 CICS .

CILEN2
Délka struktury záhlaví informací version-2 CICS .

Následující konstanta určuje délku aktuální verze:

CILENC
Délka aktuální verze struktury záhlaví informací CICS .

Toto je pole požadavku. Počáteční hodnota tohoto pole je CILEN2.

CILT (10místné celé číslo se znaménkem)

Typ odkazu.

Označuje typ objektu, který by se most měl pokusit propojit. Hodnota musí být jedna z následujících:

LTPROG
Program pro DPL.

Zleva doprava
Transakce 3270.

Toto je pole požadavku. Počáteční hodnota tohoto pole je LTPROG.

CINTI (4bajtový znakový řetězec)

Další transakce, která se má připojit.

Toto je název další transakce vrácené transakcí uživatele (obvykle EXEC CICS RETURN TRANSID). Pokud neexistuje žádná další transakce, toto pole se nastaví na mezery.

Toto je pole odpovědi používané pouze pro transakce 3270. Délka tohoto pole je dána hodnotou LNTRID. Počáteční hodnota tohoto pole je 4 mezery.

CIODL (10místné celé číslo se znaménkem)

Délka výstupních dat COMMAREA.

Jedná se o délku uživatelských dat, která mají být vrácena klientovi ve zprávě odpovědi. Tato délka zahrnuje název 8bajtového programu. Délka COMMAREA předaná propojenému programu je maximum tohoto pole a délka uživatelských dat ve zprávě požadavku minus 8.

Poznámka: Délka uživatelských dat ve zprávě je délka zprávy *bez* struktury MQCIH.

Pokud je délka uživatelských dat ve zprávě požadavku menší než *CIODL*, použije se volba DATALENGTH příkazu LINK ; to umožní funkci LINK , která je efektivně dodávána do jiné oblasti CICS .

Lze použít následující speciální hodnotu:

OLINPT
Délka výstupu je stejná jako délka vstupu.

Tato hodnota může být potřebná i v případě, že není požadována žádná odpověď, aby se zajistilo, že COMMAREA předaná propojenému programu má dostatečnou velikost.

Toto je pole požadavku použité pouze pro programy DPL. Počáteční hodnota tohoto pole OLINPT.

CIREA (10místné celé číslo se znaménkem)

IBM MQ kód příčiny nebo zpětné vazby nebo CICS EIBRESP2.

Hodnota vrácená v tomto poli závisí na *CIRET* ; viz Tabulka 690 na stránce 1031.

Toto je pole odezvy. Počáteční hodnota tohoto pole je RCNONE.

CIRET (10místné celé číslo se znaménkem)

Návratový kód z mostu.

Toto je návratový kód z souboru CICS bridge , který popisuje výsledek zpracování provedeného mostem. Pole *CIFNC*, *CICC*, *CIREA*a *CIAC* mohou obsahovat další informace (viz Tabulka 690 na stránce 1031). Hodnota je jedna z následujících:

CRC000
(0, X'000 ') Žádná chyba.

CRC001
(1, X'001 ') Příkaz EXEC CICS zjistil chybu.

CRC002
(2, X'002 ') Volání IBM MQ detekovalo chybu.

CRC003
(3, X'003 ') CICS bridge zjistil chybu.

CRC004
(4, X'004 ') CICS bridge skončilo abnormálně.

CRC005
(5, X'005 ') Aplikace skončila abnormálně.

CRC006
(6, X'006 ') Došlo k chybě zabezpečení.

CRC007
(7, X'007 ') Program není k dispozici.

CRC008
(8, X'008 ') Druhá nebo pozdější zpráva v rámci aktuální pracovní jednotky nebyla přijata v uvedeném čase.

CRC009
(9, X'009 ') Transakce není k dispozici.

Toto je pole odezvy. Počáteční hodnota tohoto pole je CRC000.

CIRFM (8bajtový znakový řetězec)

IBM MQ formátovat název zprávy odpovědi.

Jedná se o název formátu IBM MQ zprávy odpovědi, která bude odeslána jako odpověď na aktuální zprávu. Pravidla pro kódování jsou stejná jako pro pole *MDFMT* v deskriptoru MQMD.

Toto je pole požadavku použité pouze pro programy DPL. Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

CIRSI (4bajtový znakový řetězec)

Vyhrazeno.

Toto je vyhrazené pole. Hodnota musí být 4 mezery. Délka tohoto pole je dána hodnotou LNRSID.

CIRS1 (8bajtový znakový řetězec)

Vyhrazeno.

Toto je vyhrazené pole. Hodnota musí být 8 mezer.

CIRS2 (8bajtový znakový řetězec)

Vyhrazeno.

Toto je vyhrazené pole. Hodnota musí být 8 mezer.

CIRS3 (8bajtový znakový řetězec)

Vyhrazeno.

Toto je vyhrazené pole. Hodnota musí být 8 mezer.

CIRS4 (10místné celé číslo se znaménkem)

Vyhrazeno.

Toto je vyhrazené pole. Hodnota musí být 0. Toto pole není přítomno, pokud je hodnota *CIVER* menší než CIVER2.

CIRTI (4bajtový znakový řetězec)

Vyhrazeno.

Toto je vyhrazené pole. Hodnota musí být 4 mezery. Délka tohoto pole je dána hodnotou LNTRID.

CISC (4bajtový znakový řetězec)

Kód spuštění transakce.

Jedná se o indikátor určující, zda most emuluje transakci terminálu nebo transakci START. Hodnota musí být jedna z následujících:

SCSTRT
Spustit.

SCDATA
Počáteční data.

SCTERM
Ukončit vstup.

SCNONE
Není.

V odezvě z mostu je toto pole nastaveno na počáteční kód odpovídající dalšímu ID transakce obsaženému v poli *CINTI* . V odezvě jsou možné následující počáteční kódy:

- SCSTRT
- SCDATA
- SCTERM

Pro CICS Transaction Server 1.2 je toto pole pouze pole požadavku; jeho hodnota v odezvě není definována.

Pro produkt CICS Transaction Server 1.3 a následná vydání se jedná o pole požadavku i pole odezvy.

Toto pole se používá pouze pro transakce 3270. Délka tohoto pole je dána LNSTCO. Počáteční hodnota tohoto pole je SCNONE.

CISID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

CISIDV
Identifikátor pro strukturu záhlaví informací CICS .

Toto je pole požadavku. Počáteční hodnota tohoto pole je CISIDV.

CITES (celé číslo se znaménkem 10 číslic)

Stav na konci úlohy.

Toto pole zobrazuje stav transakce uživatele na konci úlohy. Vrátí se jedna z následujících hodnot:

TENOSY
Nesynchronizováno.

Transakce uživatele dosud nebyla dokončena a nebyla synchronizována. Pole *MDMT* v MQMD je v tomto případě MTRQST.

TECMIT (materiál)
Potvrdit pracovní jednotku.

Transakce uživatele ještě nebyla dokončena, ale synchronizovala první pracovní jednotku. Pole *MDMT* v deskriptoru MQMD má v tomto případě hodnotu MTDGRM.

TEBACK (ZPĚT)
Vycouvat z jednotky práce.

Transakce uživatele dosud nebyla dokončena. Aktuální jednotka práce bude vrácena zpět. Pole *MDMT* v deskriptoru MQMD má v tomto případě hodnotu MTDGRM.

TEENDT
Ukončení úlohy.

Transakce uživatele byla ukončena (nebo neukončena). Pole *MDMT* v deskriptoru MQMD má v tomto případě hodnotu MTRPLY.

Toto je pole odpovědi používané pouze pro transakce 3270. Počáteční hodnota tohoto pole je TENOSY.

CITI (4bajtový znakový řetězec)

Transakce, která se má připojit.

Pokud má *CILT* hodnotu LTTRAN, *CITI* je identifikátor transakce uživatele, která se má spustit; v tomto případě musí být uvedena neprázdná hodnota.

Má-li parametr *CILT* hodnotu LTPROG, *CITI* je kód transakce, pod kterým mají být spuštěny všechny programy v rámci pracovní jednotky. Je-li zadaná hodnota prázdná, použije se výchozí kód transakce mostu DPL CICS (CKBP). Je-li hodnota neprázdná, musí být definována jako CICS jako lokální

TRANSAKCE s počátečním programem CSQCBP00. Toto pole je použitelné pouze v případě, že má parametr *CIUOW* hodnotu CUFRST nebo CUONLY.

Toto je pole požadavku. Délka tohoto pole je dána hodnotou LNTRID. Počáteční hodnota tohoto pole je 4 mezery.

CIUOW (10místné celé číslo se znaménkem)

Řízení jednotky práce.

To řídí zpracování jednotky práce prováděné produktem CICS bridge. Můžete požádat most o spuštění jedné transakce nebo jednoho či více programů v rámci pracovní jednotky. Pole označuje, zda má CICS bridge spustit jednotku práce, provést požadovanou funkci v rámci aktuální jednotky práce, nebo ukončit jednotku práce potvrzením nebo jejím zálohováním. Pro optimalizaci toků přenosu dat jsou podporovány různé kombinace.

Hodnota musí být jedna z následujících:

POUZE KRYCHLE
Spusťte pracovní jednotku, proveďte funkci a poté potvrďte pracovní jednotku (DPL a 3270).

CUCONT
Další data pro aktuální pracovní jednotku (pouze 3270).

CUFRST
Spusťte pracovní jednotku a proveďte funkci (pouze DPL).

CUMIDL
Provést funkci v rámci aktuální pracovní jednotky (pouze DPL).

CULAST (HORNÍ)
Proveďte funkci a poté potvrďte pracovní jednotku (pouze DPL).

CUCMIT
Potvrdit pracovní jednotku (pouze DPL).

CUBACK
Vycouvat jednotku práce (pouze DPL).

Toto je pole požadavku. Počáteční hodnota tohoto pole je CUONLY.

CIVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

CIVER1
Version-1 CICS struktura záhlaví informací.

CIVER2
Version-2 CICS struktura záhlaví informací.

Pole, která existují pouze v novější verzi struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

CIVERC
Aktuální verze struktury záhlaví informací CICS .

Toto je pole požadavku. Počáteční hodnota tohoto pole je CIVER2.

Počáteční hodnoty

Tabulka 691. Pole v MQCIH

Název pole	Název konstanty	Hodnota konstanty
CISID	CISIDV	'CIH¬'
CIVER	CIVER2	2

Tabulka 691. Pole v MQCIH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
CILEN	CILEN2	180
CIENC	Není	0
CICSI	Není	0
CIFMT	FMNONE	Mezery
CIFLG	CIFNON	0
CIRET	CRC000	0
CICC	CCOK	0
CIREA	RCNONE	0
CIUOW	POUZE KRYCHLE	273
CIGWI	WIDFLT	-2
CILT	LTPROG	1
CIODL	OLINPT	-1
CIFKT	Není	0
CIADS	ADNONE	0
CICT	CTNO	0
CITES	TENOSY	0
CIFAC	FCNONE	Hodnoty null
CIFNC	CFNONE	Mezery
CIAC	Není	Mezery
CIAUT	Není	Mezery
CIRS1	Není	Mezery
CIRFM	FMNONE	Mezery
CIRSI	Není	Mezery
CIRTI	Není	Mezery
CITI	Není	Mezery
CIFL	Není	Mezery
CIAI	Není	Mezery
CISC	SCNONE	Mezery
CICNC	Není	Mezery
CINTI	Není	Mezery
CIRS2	Není	Mezery
CIRS3	Není	Mezery
CICP	Není	0
CIEO	Není	0

Tabulka 691. Pole v MQCIH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
CIII	Není	0
CIRS4	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQCIH Structure
D*
D* Structure identifier
D  CISID                  1      4    INZ('CIH ')
D* Structure version number
D  CIVER                  5      8I 0 INZ(2)
D* Length of MQCIH structure
D  CILEN                  9     12I 0 INZ(180)
D* Reserved
D  CIENC                 13     16I 0 INZ(0)
D* Reserved
D  CICSI                 17     20I 0 INZ(0)
D* MQ format name of data that followsMQCIH
D  CIFMT                 21     28    INZ('        ')
D* Flags
D  CIFLG                 29     32I 0 INZ(0)
D* Return code from bridge
D  CIRET                 33     36I 0 INZ(0)
D* MQ completion code or CICSEIBRESP
D  CICC                  37     40I 0 INZ(0)
D* MQ reason or feedback code, or CICSEIBRESP2
D  CIREA                 41     44I 0 INZ(0)
D* Unit-of-work control
D  CIUOW                 45     48I 0 INZ(273)
D* Wait interval for MQGET call issuedby bridge task
D  CIGWI                 49     52I 0 INZ(-2)
D* Link type
D  CILT                  53     56I 0 INZ(1)
D* Output COMMAREA data length
D  CIODL                 57     60I 0 INZ(-1)
D* Bridge facility release time
D  CIFKT                 61     64I 0 INZ(0)
D* Send/receive ADS descriptor
D  CIADS                 65     68I 0 INZ(0)
D* Whether task can beconversational
D  CICT                  69     72I 0 INZ(0)
D* Status at end of task
D  CITES                 73     76I 0 INZ(0)
D* Bridge facility token
D  CIFAC                 77     84    INZ(X'00000000000000-
D                                     00')
D* MQ call name or CICS EIBFNfunction
D  CIFNC                 85     88    INZ('    ')
D* Abend code
D  CIAC                  89     92    INZ
D* Password or passticket
D  CIAUT                 93    100    INZ
D* Reserved
D  CIRS1                101    108    INZ
D* MQ format name of reply message
D  CIRFM                109    116    INZ('        ')
D* Remote CICS system ID to use
D  CIRSI                117    120    INZ
D* CICS RTRANSID to use
D  CIRTI                121    124    INZ
D* Transaction to attach
D  CITI                 125    128    INZ
D* Terminal emulated attributes
D  CIFL                 129    132    INZ
D* AID key
D  CIAI                 133    136    INZ
```

```
     D* Transaction start code
     D  CISC                 137    140    INZ('    ')
     D* Abend transaction code
     D  CICNC                141    144    INZ
     D* Next transaction to attach
     D  CINTI                145    148    INZ
     D* Reserved
     D  CIRS2                149    156    INZ
     D* Reserved
     D  CIRS3                157    164    INZ
     D* Cursor position
     D  CICP                 165    168I 0 INZ(0)
     D* Offset of error in message
     D  CIEO                 169    172I 0 INZ(0)
     D* Reserved
     D  CIII                 173    176I 0 INZ(0)
     D* Reserved
     D  CIRS4                177    180I 0 INZ(0)
     D*
```

IBM i MQCMHO (Vytvoření voleb popisovače zprávy) v systému IBM i

Struktura **MQCMHO** umožňuje aplikacím určit volby, které řídí způsob vytváření manipulátorů zpráv.

Přehled

Účel
Struktura je vstupní parametr volání **MQCRTMH** .

Znaková sada a kódování
Data v souboru **MQCMHO** musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1042
- "Počáteční hodnoty" na stránce 1044
- "Deklarace RPG" na stránce 1044

Pole

Struktura MQCMHO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

CMOPT (10místné celé číslo se znaménkem)

Lze zadat jednu z následujících voleb:

CMVAL

Při volání **MQSETMP** pro nastavení vlastnosti v tomto popisovači zprávy je název vlastnosti ověřen, aby se zajistilo, že:

- neobsahuje žádné neplatné znaky.
- nezačíná "JMS" nebo "usr.JMS" s výjimkou následujících:
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType.
  - JMSXGroupID
  - JMSXGroupSeq

  Tyto názvy jsou vyhrazeny pro vlastnosti JMS .
- není jedním z následujících klíčových slov, v jakékoli směsi velkých nebo malých písmen:
  - "A"
  - "Mezi"
  - "ESCAPE"

- "FALSE"
- "V"
- "JE"
- "LIKE"
- "NIKOLI"
- "NULL" (nedefinováno)
- "NEBO"
- "TRUE"

- nezačíná "Tělo". nebo "Kořen." (s výjimkou "Root.MQMD.").

Je-li vlastnost MQ-defined ("mq. *") a název je rozpoznán, pole deskriptoru vlastností jsou nastavena na správné hodnoty pro vlastnost. Není-li vlastnost rozpoznána, pole *Support* deskriptoru vlastnosti je nastaveno na hodnotu **PDSUPO** (další informace viz PDSUP).

CMDEFV

Tato volba určuje, že dojde k výchozí úrovni ověřování názvů vlastností.

Výchozí úroveň ověření je ekvivalentní úrovni určené parametrem **CMVAL**.

V budoucím vydání může být definována administrativní volba, která změní úroveň ověření, ke kterému dojde při definování **CMDEFV** .

Toto je výchozí hodnota.

CMNOVA

Nedojde k žádnému ověření názvu vlastnosti. Viz popis položky **CMVAL**.

Výchozí volba: Pokud není požadována žádná z voleb dříve popsaných v této sekci, lze použít následující volbu:

CMNONE

Všechny volby předpokládají své výchozí hodnoty. Pomocí této hodnoty označíte, že nebyly zadány žádné další volby. Produkt **CMNONE** pomáhá s dokumentací k programům; není určeno, aby byla tato volba použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **CMDEFV**.

CMSID (10místné celé číslo se znaménkem)

Toto je identifikátor struktury; hodnota musí být:

CMSIDV
Identifikátor pro vytvoření struktury voleb popisovače zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **CMSIDV**.

CMVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury; hodnota musí být:

CMVER1
Version-1 vytvořte strukturu voleb popisovače zprávy.

Následující konstanta určuje číslo verze aktuální verze:

CMVERC
Aktuální verze struktury voleb popisovače vytvoření zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **CMVER1**.

Počáteční hodnoty

Tabulka 692. Pole v MQCMHO		
Název pole	**Název konstanty**	**Hodnota konstanty**
CMSID	CMSIDV	'CMHO'
CMVER	CMVER1	1
CMOPT	CMDEFV	0

Deklarace RPG

```
D* MQCMHO Structure
D*
D*
D* Structure identifier
D  CMSID                  1      4    INZ('CMHO')
D*
D* Structure version number
D  CMVER                  5      8I 0 INZ(1)
D*
D* Options that control the action of MQCRTMH
D  CMOPT                  9     12I 0 INZ(0)
```

IBM i MQCNO (Volby připojení) na systému IBM i

Struktura MQCNO umožňuje aplikaci určit volby související s připojením k lokálnímu správci front.

Přehled

Účel: Struktura je vstupní/výstupní parametr pro volání MQCONNX.

Verze: Aktuální verze MQCNO je CNVER6. Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi MQCNO, která je podporována prostředím, ale s počáteční hodnotou pole CNVER nastavenou na hodnotu CNVER1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole CNVER na číslo verze požadované verze.

Znaková sada a kódování: Data v MQCNO musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT.

- "Pole" na stránce 1044
- "Počáteční hodnoty" na stránce 1049
- "Deklarace RPG" na stránce 1050

Pole

Struktura MQCNO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CCDTUL (celé číslo se znaménkem 10 číslic)

CCDTUL je délka řetězce identifikovaného pomocí CCDTUP nebo CCDTUO, který obsahuje URL identifikující umístění tabulky kanálů připojení klienta, která má být použita pro připojení.

Volbu CCDTUL použijte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client.

Jedná se o programovou alternativu k nastavení proměnných prostředí MQCHLLIB a MQCHLTAB .

Pokud aplikace není spuštěna jako klient, bude hodnota CCDTUL ignorována.

Toto pole je ignorováno, pokud je CNVER menší než CNVER6.

CCDTUO (celé číslo se znaménkem 10 číslic)

CCDTUO je posun v bajtech od začátku struktury MQCNO k řetězci, který obsahuje URL identifikující umístění tabulky kanálů připojení klienta, která má být použita pro připojení. Posun může být kladný nebo záporný.

Volbu CCDTUL použijte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client.

Důležité: Můžete použít pouze jeden z CCDTUP a CCDTUO. Volání selže s kódem příčiny RC2600 , pokud jsou obě pole nenulová.

Jedná se o programovou alternativu k nastavení proměnných prostředí MQCHLLIB a MQCHLTAB .

Pokud aplikace není spuštěna jako klient, bude hodnota CCDTUO ignorována.

Toto pole je ignorováno, pokud je CNVER menší než CNVER6.

CCDTUP (ukazatel)

CCDTUP je volitelný ukazatel na řetězec, který obsahuje URLpro identifikaci umístění tabulky kanálů připojení klienta, která se má použít pro připojení.

Volbu CCDTUP použijte pouze v případě, že je aplikace vydávající volání MQCONNX spuštěna jako IBM MQ MQI client.

Důležité: Můžete použít pouze jeden z CCDTUP a CCDTUO. Volání selže s kódem příčiny RC2600 , pokud jsou obě pole nenulová.

Jedná se o programovou alternativu k nastavení proměnných prostředí MQCHLLIB a MQCHLTAB .

Pokud aplikace není spuštěna jako klient, bude CCDTUP ignorován.

Toto pole je ignorováno, pokud je CNVER menší než CNVER6.

CNAN (28bajtový znakový řetězec)

Název nastavený aplikací pro identifikaci připojení ke správci front. Počáteční hodnota pole je null znaků.

Toto pole je ignorováno, pokud je CNVER menší než CNVER7.

CNCCO (10místné celé číslo se znaménkem)

Toto je posun v bajtech struktury definice kanálu MQCD od začátku struktury MQCNO.

CNCCP (ukazatel)

Jedná se o ukazatel na strukturu definice kanálu MQCD.

CNCONID (24bajtový znakový řetězec)

Jedinečný identifikátor připojení. Toto pole umožňuje správci front spolehlivě identifikovat proces aplikace přiřazením jedinečného identifikátoru při prvním připojení ke správci front.

Aplikace používají identifikátor připojení pro účely korelace při volání PUT a GET. Všem připojením je správcem front přiřazen identifikátor bez ohledu na způsob vytvoření připojení.

Identifikátor připojení je možné použít k vynucení ukončení dlouho běžící jednotky práce. To provedete zadáním identifikátoru připojení pomocí příkazu PCF 'Zastavit připojení' nebo příkazu MQSC STOP CONN. Další informace o použití těchto příkazů naleznete v souvisejících odkazech.

Počáteční hodnota pole je 24 nulových bajtů.

CNCT (128bajtový bitový řetězec)

Jedná se o značku, kterou správce front přidruží k prostředkům, které jsou během tohoto připojení ovlivněny aplikací.

Značka připojení správce front.

Každá aplikace nebo instance aplikace musí pro značku používat jinou hodnotu, aby mohl správce front správně serializovat přístup k ovlivněným prostředkům. Další podrobnosti viz popisy voleb CN* CT*. Značka přestane být platná při ukončení aplikace nebo vydá volání MQDISC.

Pokud není požadována žádná značka, použijte následující speciální hodnotu:

CTNONE
Nebyla určena žádná značka připojení.

Hodnota je binární nula pro délku pole.

Toto je vstupní pole. Délka tohoto pole je dána LNCTAG. Počáteční hodnota tohoto pole je CTNONE. Toto pole je ignorováno, pokud je hodnota CNVER menší než CNVER3.

Při připojování ke správci front z/OS použijte pole ConnTag .

CNNORES2 (4bajtový znakový řetězec)

Vyhrazené pole pro vyplnění struktury na 64bitovou hranici. Počáteční hodnota pole je binární nula pro délku pole.

Toto pole je ignorováno, pokud je CNVER menší než CNVER7.

CNOPT (10místné celé číslo se znaménkem)

Volby, které řídí akci MQCONNX.

Volby vazeb

Volby vazby řídí typ použité vazby IBM MQ . Zadejte pouze jednu z těchto voleb:

CNSBND
Standardní vazba.

Volba standardní vazby způsobí, že se aplikace a lokální agent správce front spustí v oddělených jednotkách provedení, obvykle v oddělených procesech. Uspořádání udržuje integritu správce front; to znamená, že chrání správce front před chybným programem.

Volbu CNSBND použijte v situacích, kdy aplikace možná nebyla plně otestována nebo byla nespolehlivá či nedůvěryhodná. CNSBND je předvolba.

CNSBND je definován jako pomůcka pro programovou dokumentaci. Tuto volbu nepoužívejte s žádnou jinou volbou, která by řídila použitý typ vazby; ale protože je její hodnota nula, nelze takové použití zjistit.

Tato volba je podporována ve všech prostředích.

CNFBND
Vazba rychlé cesty.

Volba vazby rychlé cesty způsobí, že aplikace a lokální agent správce front budou součástí stejné jednotky provedení. Rychlá cesta je na rozdíl od standardní vazby, kde aplikace a lokální agent správce front běží v oddělených jednotkách provedení.

CNFBND se ignoruje, pokud správce front nepodporuje tento typ vazby; zpracování pokračuje, jako by volba nebyla uvedena.

Funkce CNFBND může být výhodou v situacích, kdy více procesů spotřebuje více prostředků než celkový prostředek používaný aplikací.Aplikace, která používá vazbu rychlé cesty, se nazývá *důvěryhodná aplikace*.

Při rozhodování, zda použít vazbu rychlé cesty, zvažte následující důležité body:

- **Použití volby CNFBND nebrání aplikaci měnit nebo poškozovat zprávy a další datové oblasti patřící správci front. Tuto volbu použijte pouze v situacích, kdy jste tyto problémy plně vyhodnotili.**

- Aplikace nesmí používat asynchronní signály nebo přerušení časovače (například `sigkill`) s CNFBND. Existují také omezení použití segmentů sdílené paměti.
- Aplikace nesmí mít v daném okamžiku k dispozici více než jeden podproces připojený ke správci front.
- K odpojení od správce front musí aplikace použít volání MQDISC .
- Před ukončením správce front příkazem endmqm musí být aplikace dokončena.

Pro použití funkce CNFBND v označených prostředích platí následující body:

- V systému IBM imusí být úloha spuštěna pod profilem uživatele QMQM , který patří do skupiny QMQMADM . Program také nesmí být ukončen abnormálně, jinak by mohlo dojít k nepředvídatelným výsledkům.

Další informace o důsledcích používání důvěryhodných aplikací naleznete v tématu Připojení ke správci front pomocí volání MQCONNX a Omezení pro důvěryhodné aplikace.

CNSHBD
Sdílené vazby.

Volba sdílených vazeb způsobí, že aplikace a lokální agent správce front budou spuštěny v oddělených jednotkách provedení, obvykle v oddělených procesech. Uspořádání udržuje integritu správce front; to znamená, že chrání správce front před chybným programem. Některé prostředky jsou však sdíleny mezi aplikací a lokálním agentem správce front. CNSHBD se ignoruje, pokud správce front nepodporuje tento typ vazby. Zpracování pokračuje, jako by volba nebyla uvedena.

CNIBND
Izolované vazby.

Volba izolovaných vazeb způsobí, že aplikace a lokální agent správce front budou spuštěny v oddělených jednotkách provedení, obvykle v oddělených procesech. Uspořádání udržuje integritu správce front; to znamená, že chrání správce front před chybným programem. Proces aplikace a lokální agent správce front jsou vzájemně izolovány v tom, že nesdílejí prostředky. CNIBND se ignoruje, pokud správce front nepodporuje tento typ vazby. Zpracování pokračuje, jako by volba nebyla uvedena.

Volby sdílení popisovače

Následující volby řídí sdílení manipulátorů mezi různými podprocesy (jednotkami paralelního zpracování) v rámci stejného procesu. Lze zadat pouze jednu z těchto voleb.

CNHSN
Žádné sdílení manipulátoru mezi podprocesy.

Volba bez sdílení manipulátorů mezi podprocesy označuje, že manipulátory připojení a objektu mohou být použity pouze podprocesem, který způsobil alokaci manipulátoru, tj. podprocesem, který vydal volání MQCONN, MQCONNXnebo MQOPEN . Manipulátory nemohou být použity jinými podprocesy náležejícími ke stejnému procesu.

CNHSB
Sériová rukojeť sdílení mezi vlákny, s blokováním volání.

Sériové sdílení manipulátorů mezi podprocesy s možností blokování volání označuje, že manipulátory připojení a objektů přidělené jedním podprocesem procesu mohou být použity jinými podprocesy náležejícími ke stejnému procesu. Avšak pouze jeden podproces v daném okamžiku může použít jakoukoli konkrétní rukojeť, to znamená, že je povoleno pouze sériové použití rukojeť. Pokud se podproces pokusí použít manipulátor, který je již používán jiným podprocesem, bude volání blokovat (čekat), dokud nebude manipulátor k dispozici.

CNHSNB
Sériová rukojeť sdílení mezi vlákny, bez blokování volání.

Sériová rukojeť sdílení mezi vlákny, bez blokování volání, volba je stejná jako " Volba *s* blokováním " s tím rozdílem, že pokud manipulátor používá jiný podproces, volání se okamžitě dokončí s CCFAIL a RC2219 namísto blokování, dokud nebude manipulátor k dispozici.

Podproces může mít žádný nebo jeden nesdílený popisovače, plus žádný nebo více sdílených popisovačů:

- Každé volání MQCONN nebo MQCONNX , které uvádí CNHSN , vrátí nový nesdílený popisovač pro první volání a stejný nesdílený popisovač pro následná volání (za předpokladu, že nepřichází v žádné intervenující volání MQDISC). Kód příčiny je RC2002 pro druhé a pozdější volání.
- Každé volání MQCONNX , které uvádí CNHSB nebo CNHSNB , vrátí nový sdílený manipulátor pro každé volání.

Manipulátory objektů dědí stejné vlastnosti sdílení jako manipulátor připojení určený ve volání MQOPEN , které vytvořilo manipulátor objektu. Jednotky práce také dědí stejné vlastnosti sdílení jako manipulátor připojení použitý ke spuštění jednotky práce; pokud je jednotka práce spuštěna v jednom podprocesu pomocí sdíleného manipulátoru, lze jednotku práce aktualizovat v jiném podprocesu pomocí stejného manipulátoru.

Pokud nezadáte volbu sdílení manipulátoru, výchozí nastavení bude určeno prostředím:

- V prostředí Microsoft Transaction Server (MTS) je výchozí nastavení stejné jako CNHSB.
- V jiných prostředích je výchozí hodnota stejná jako hodnota CNHSN.

Volby opětovného připojení

Volby opětovného připojení určují, zda lze připojení znovu připojit. Lze znovu připojit pouze klientská připojení.

CNRCDF
Volba opětovného připojení se interpretuje na výchozí hodnotu. Není-li nastavena žádná výchozí hodnota, hodnota této volby se interpretuje jako DISABLED. Hodnota volby je předána serveru a může být dotazována pomocí **PCF** a **MQSC**.

CNRC
Aplikaci lze znovu připojit k libovolnému správci front, který je konzistentní s hodnotou parametru MQCONNX **QMNAME** . Volbu CNRC použijte pouze v případě, že mezi klientskou aplikací a správcem front, se kterým původně navázala připojení, neexistuje žádná afinita. Hodnota volby je předána serveru a může být dotazována pomocí **PCF** a **MQSC**.

CNRCD
Aplikaci nelze znovu připojit. Hodnota volby není předána serveru.

CNRCQM
Aplikaci lze znovu připojit pouze ke správci front, ke kterému byla původně připojena. Tuto hodnotu použijte, pokud lze klienta znovu připojit, ale existuje afinita mezi aplikací klienta a správcem front, se kterým původně navázal připojení. Tuto hodnotu zvolte tehdy, chcete-li, aby se klient automaticky připojil znovu k instanci značně dostupného správce front, která je v pohotovostním režimu. Hodnota volby je předána serveru a může být dotazována pomocí **PCF** a **MQSC**.

Volby CNRC, CNRCDa CNRCQM použijte pouze pro připojení klienta. Pokud jsou volby použity pro připojení vazby, MQCONNX selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_OPTIONS_ERROR.

Výchozí volba: Pokud není požadována žádná z popsaných voleb, lze použít následující volbu:

CNNONE
Nejsou zadány žádné volby.

CNNONE je definováno jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby se tato volba používala s jinou volbou CN* , ale protože její hodnota je nula, nelze takové použití zjistit.

CNSCO (10místné celé číslo se znaménkem)

Toto je posun v bajtech struktury MQSCO od začátku struktury MQCNO.

Toto pole je ignorováno, pokud je CNVER menší než CNVER4.

CNSCP (ukazatel)

Jedná se o adresu struktury MQSCO.

Toto pole je ignorováno, pokud je CNVER menší než CNVER4.

CNSECPO (10místné celé číslo se znaménkem)

Posunutí parametrů zabezpečení. Posun struktury MQCSP použitý pro zadání ID uživatele a hesla.

Hodnota může být kladná nebo záporná. Počáteční hodnota tohoto pole je 0.

Toto pole se ignoruje, pokud je CNVER menší než CNVER5.

CNSECPP (ukazatel)

Ukazatel parametrů zabezpečení. Adresa struktury MQCSP, která se používá k určení ID uživatele a hesla.

Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

Toto pole se ignoruje, pokud je CNVER menší než CNVER5.

CNSID (4bajtový znakový řetězec)

Identifikátor struktury pro strukturu MQCNO.

Hodnota musí být:

CNSIDV
Identifikátor pro strukturu voleb připojení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CNSIDV.

CNVER (10místné celé číslo se znaménkem)

Číslo verze struktury pro strukturu MQCNO.

Hodnota musí být:

CNVER6
Struktura voleb připojení Version-6 .

Tato verze je podporována ve všech prostředích.

CNVER7
Version-7 struktura voleb připojení.

Tato verze je podporována ve všech prostředích.

Následující konstanta určuje číslo verze aktuální verze:

CNVERC-připojení
Aktuální verze struktury voleb připojení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CNVER7.

Počáteční hodnoty

Tabulka 693. Pole v MQCNO

Název pole	Název konstanty	Hodnota konstanty
CNSID	CNSIDV	'CNO¬
CNVER	CNVER5	1

Tabulka 693. Pole v MQCNO (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
CNOPT	CNNONE	0
CNCCO	Není	0
CNCCP	Není	Ukazatel Null nebo bajty s hodnotou Null
CNCT	CTNONE	Hodnoty null
CNSCP	Není	Ukazatel Null nebo bajty s hodnotou Null
CNSCO	Není	0
CNCONID	Není	Hodnoty null
CNSECPO	Není	0
CNSECPP	Není	Ukazatel Null nebo bajty s hodnotou Null
CCDTUL	Není	0
CCDTUO	Není	0
CCDTUP	Není	Ukazatel Null nebo bajty s hodnotou Null

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*****************************************************************
     D**                                                          **
     D**                   IBM MQ for IBM i                       **
     D**                                                          **
     D**  FILE NAME:      CMQCNOG                                 **
     D**                                                          **
     D**  DESCRIPTION:    MQCNO Structure -- Connect Options      **
     D**                                                          **
     D*************************************************************
     D**  <N_OCO_COPYRIGHT>                                       **
     D**  Licensed Materials - Property of IBM                    **
     D**                                                          **
     D**  5724-H72                                                **
     D**  (c) Copyright IBM Corp. 1993, 2024. All Rights Reserved.**
     D**                                                          **
     D**  US Government Users Restricted Rights - Use, duplication or **
     D**  disclosure restricted by GSA ADP Schedule Contract with **
     D**  IBM Corp.                                               **
     D**  <NOC_COPYRIGHT>                                         **
     D*************************************************************
     D**                                                          **
     D**  FUNCTION:       This file declares the structure MQCNO, **
     D**                  which is used by the main MQI.          **
     D**                                                          **
     D**  PROCESSOR:      RPG (ILE)                               **
     D**                                                          **
     D*************************************************************
     D*
     D*
     D*************************************************************
     D** <BEGIN_BUILDINFO>                                        **
     D** Generated on:  08/02/16 13:50                           **
     D** Build Level:   L000000                                  **
     D** Build Type:    Production                               **
     D** Pointer Size:  128 Bit                                  **
```

```
     D** Source File:                                                  **
     D** CMQCNOG                                                       **
     D** <END_BUILDINFO>                                               **
     D*****************************************************************
     D*
     D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     D*
     D*
     D* MQCNO Structure
     D*
     D* Structure identifier
     D  CNSID                  1      4    INZ('CNO ')
     D* Structure version number
     D  CNVER                  5      8I 0 INZ(1)
     D* Options that control the action of MQCONNX
     D  CNOPT                  9     12I 0 INZ(0)
     D* Ver:1 **
     D* Offset of MQCD structure for client connection
     D  CNCCO                 13     16I 0 INZ(0)
     D* Address of MQCD structure for client connection
     D  CNCCP                 17     32*   INZ(*NULL)
     D* Ver:2 **
     D* Queue managerconnection tag
     D  CNCT                  33    160    INZ(X'00000000000000000-
     D                                     0000000000000000000000000-
     D                                     0000000000000000000000000-
     D                                     0000000000000000000000000-
     D                                     0000000000000000000000000-
     D                                     0000000000000000000000000-
     D                                     0000000000000000000000000-
     D                                     0000000000000000000000000-
     D                                     0000000000000000000000000-
     D                                     0000000000000000000000000-
     D                                     00000000000000')
     D* Ver:3 **
     D* Address of MQSCO structure for client connection
     D  CNSCP                 161    176*   INZ(*NULL)
     D* Offset of MQSCO structure for client connection
     D  CNSCO                 177    180I 0 INZ(0)
     D* Ver:4 **
     D* Unique Connection Identifier
     D  CNCONID               181    204    INZ(X'00000000000000000-
     D                                     0000000000000000000000000-
     D                                     000000')
     D* Offset of MQCSP structure
     D  CNSECPO               205    208I 0 INZ(0)
     D* Address of MQCSP structure
     D  CNSECPP               209    224*   INZ(*NULL)
     D* Ver:5 **
     D* Address of CCDT URL string
     D  CNCCDTUP              225    240*   INZ(*NULL)
     D* Offset of CCDT URL string
     D  CNCCDTUO              241    244I 0 INZ(0)
     D* Length of CCDT URL
     D  CNCCDTUL              245    248I 0 INZ(0)
     D* Ver:6 **
     D*
     D*****************************************************************
     D**  End of CMQCNOG                                               **
     D*****************************************************************
```

IBM i MQCSP (parametry zabezpečení) na systému IBM i

Souhrn struktury MQCSP pro IBM i.

Přehled

Účel: Struktura MQCSP umožňuje autorizační službě ověřit ID uživatele a heslo. Strukturu parametrů zabezpečení připojení MQCSP zadáváte ve volání MQCONNX.

Znaková sada a kódování: Data v protokolu MQCSP musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT.

- "Pole" na stránce 1052
- "Počáteční hodnoty" na stránce 1054

- “Deklarace RPG” na stránce 1054

Pole

Struktura MQCSP obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CSAUTHT (10místné celé číslo se znaménkem)

Jedná se o typ ověřování, které má být provedeno.

Platné jsou tyto hodnoty:

CSAN
Nepoužívejte pole ID uživatele a hesla.

CSAUIAP
Ověřit pole ID uživatele a hesla.

Toto je vstupní pole. Počáteční hodnota tohoto pole je CSAN.

CSCPPL (10místné celé číslo se znaménkem)

Jedná se o délku hesla, které má být použito při ověřování.

Maximální délka hesla není závislá na platformě. Pokud je délka hesla větší, než je povoleno, požadavek na ověření selže s RC2035.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSCPPO (10místné celé číslo se znaménkem)

Jedná se o posun hesla v bajtech, který se má použít při ověření.

Posun může být kladný nebo záporný.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSCPPP (ukazatel)

Jedná se o adresu hesla, která má být použita při ověřování.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null.

CSCSPUIL (10místné celé číslo se znaménkem)

Jedná se o délku ID uživatele, které se má použít při ověřování.

Maximální délka ID uživatele není závislá na platformě. Pokud je délka ID uživatele větší, než je povoleno, požadavek na ověření selže s RC2035.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSCSPUIO (10místné celé číslo se znaménkem)

Toto je posun v bajtech ID uživatele, který se má použít při ověření.

Posun může být kladný nebo záporný.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSCSPUIP (ukazatel)

Jedná se o adresu ID uživatele, které má být použito při ověřování.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null. Toto pole je ignorováno, pokud je hodnota CSVER menší než CSVER5.

V 9.3.0 V 9.3.0 **CSINITKL (10místné celé číslo se znaménkem)**

Jedná se o délku výchozího klíče pro systém ochrany heslem.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

V 9.3.0 V 9.3.0 **CSINITKO (10místné celé číslo se znaménkem)**

Jedná se o posun počátečního klíče pro systém ochrany heslem v bajtech. Posun může být kladný nebo záporný.

K určení počátečního klíče můžete použít buď *CSINITKO* , nebo *CSINITKP* , ale ne obojí. Další informace viz popis pole *CSINITKP* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

V 9.3.0 V 9.3.0 **CSINITKP (ukazatel)**

Jedná se o adresu počátečního klíče pro systém ochrany heslem v bajtech.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null.

Produkt IBM MQ MQI clients může zadat hodnotu některých polí jako hodnoty, které byly zašifrovány pomocí systému IBM MQ pro ochranu heslem. Následující pole mohou obsahovat zašifrované hodnoty:

- Heslo úložiště klíčů ve struktuře MQSCO.

Počáteční klíč je používán šifrovacím algoritmem k šifrování a dešifrování těchto hodnot. Pokud je počáteční klíč zadán, když jsou hodnoty těchto polí šifrovány pomocí obslužného programu **runmqicred** , musí klient při připojení ke správci front zadat stejný počáteční klíč.

Počáteční klíč určený pomocí tohoto pole přepíše jakýkoli počáteční klíč určený pomocí proměnné prostředí *MQS_MQI_KEYFILE* nebo vlastnosti *MQIInitialKey* v sekci Zabezpečení konfiguračního souboru klienta.

K určení počátečního klíče můžete použít buď *CSINITKO* , nebo *CSINITKP* , ale ne obojí.

CSRE1 (4bajtový znakový řetězec)

Vyhrazené pole požadované pro zarovnání ukazatele na IBM i.

Toto je vstupní pole. Počáteční hodnota tohoto pole je null.

CSRS2 (8bajtový znakový řetězec)

Vyhrazené pole požadované pro zarovnání ukazatele na IBM i.

Toto je vstupní pole. Počáteční hodnota tohoto pole je null.

CSSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

CSSIDV
Identifikátor struktury parametrů zabezpečení.

CSVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

CSVER1
Struktura parametrů zabezpečení Version-1 .

V 9.3.0 V 9.3.0 V 9.3.0 **CSVER2**
Struktura parametrů zabezpečení Version-2 .

Následující konstanta určuje číslo verze aktuální verze:

CSVERC
Aktuální verze struktury parametrů zabezpečení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CSVER1.

Počáteční hodnoty

Tabulka 694. Pole v MQCNO		
Název pole	**Název konstanty**	**Hodnota konstanty**
CSSID	CSSIDV	'CSP¬'
CSVER	CSVER1	1
CSAUTHT	Není	0
CSRE1	Není	Hodnoty null
CSCSPUIP	Není	Nedefinovaný ukazatel.
CSCSPUIO	Není	0
CSCSPUIL	Není	0
CSRS2	Není	Hodnoty null
CSCPPP	Není	Nedefinovaný ukazatel.
CSCPPO	Není	0
CSCPPL	Není	0
V 9.3.0 V 9.3.0 *CSINITKP*	Není	Nedefinovaný ukazatel.
V 9.3.0 V 9.3.0 *CSINITKO*	Není	0
V 9.3.0 V 9.3.0 *CSINITKL*	Není	0

Poznámka:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
      D*
      D* MQCSP Structure
      D*
      D* Structure identifier
      D  CSSID                  1      4    INZ('CSP ')
      D* Structure version number
      D  CSVER                  5      8I 0 INZ(1)
      D* Type of authentication
      D  CSAUTHT                9     12I 0 INZ(0)
      D* Reserved
      D  CSRE1                 13     16    INZ(X'00000000')
      D* Address of user ID
      D  CSCSPUIP              17     32*   INZ(*NULL)
      D* Offset of user ID
      D  CSCSPUIO              33     36I 0 INZ(0)
      D* Length of user ID
      D  CSCSPUIL              37     40I 0 INZ(0)
      D* Reserved
      D  CSRS2                 41     48    INZ(X'0000000000000000')
      D* Address of password
      D  CSCPPP                49     64*   INZ(*NULL)
      D* Offset of password
      D  CSCPPO                65     68I 0 INZ(0)
```

```
      D* Length of password
      D  CSCPPL                69     72I 0 INZ(0)
```

IBM i MQCTLO (Struktura voleb zpětného volání řízení) na systému IBM i

Struktura určující funkci zpětného volání ovládacího prvku.

Přehled

Účel

Struktura MQCTLO se používá k určení voleb souvisejících s funkcí zpětného volání řízení.

Struktura je vstupní a výstupní parametr volání MQCTL .

Verze

Aktuální verze MQCTLO je CTLV1.

Znaková sada a kódování

Data v objektu MQCTLO musí být ve znakové sadě zadané atributem správce front **CodedCharSetId** a v kódování lokálního správce front dané tabulkou ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , musí být struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1055
- "Počáteční hodnoty" na stránce 1056
- "Deklarace RPG" na stránce 1056

Pole

Struktura MQCTLO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

COCONNAREA (10místné celé číslo se znaménkem)

Struktura voleb řízení-pole ConnectionArea .

Jedná se o pole, které je k dispozici pro použití funkcí zpětného volání.

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a je předán beze změny z pole CBCCONNAREA ve struktuře MQCBC, což je parametr volání MQCB.

Toto pole je ignorováno pro všechny operace kromě CTLSR a CTLSW.

Toto je vstupní a výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel Null nebo nulový počet bajtů.

COOPT (10místné celé číslo se znaménkem)

Volby, které řídí akci MQCTLO.

CTLFQ
Vynutit selhání volání MQCTLO, pokud je správce front nebo připojení ve stavu uvedení do klidového stavu.

Zadejte GMFIQ ve volbách MQGMO předaných na volání MQCB, abyste způsobili oznámení spotřebitelům zpráv při jejich uvedení do klidového stavu.

CTLTHR
Tato volba informuje systém, že aplikace vyžaduje, aby všichni spotřebitelé zpráv pro stejné připojení byli voláni ve stejném podprocesu.

Výchozí volba: Pokud nepotřebujete žádnou z popsaných voleb, použijte následující volbu:

CTLNO
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty. CTLNO je definován jako pomůcka pro dokumentaci programu; není zamýšleno, aby tato volba byla použita s jinou, ale protože její hodnota je nula, nelze takové použití zjistit.

Toto je vstupní pole. Počáteční hodnota pole *COOPT* je CTLNO.

CORSV (10místné celé číslo se znaménkem)

Toto je vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak.

COSID (10místné celé číslo se znaménkem)

Struktura voleb řízení-pole StrucId .

Toto je identifikátor struktury; hodnota musí být:

CTLSI
Identifikátor pro strukturu voleb řízení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CTLSI.

COVER (10místné celé číslo se znaménkem)

Struktura voleb řízení-pole Verze.

Toto je číslo verze struktury; hodnota musí být:

CTLV1
Version-1 Struktura voleb řízení.

Následující konstanta určuje číslo verze aktuální verze:

CTLCV
Aktuální verze struktury voleb řízení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CTLV1.

Počáteční hodnoty

Tabulka 695. Pole v MQCTLO

Název pole	Název konstanty	Hodnota konstanty
COSID	CTLSI	'CTLO'
COVER	CTLV1	1
COOPT	CTLNO	Hodnoty null
CORSV	Vyhrazené pole	
COCONNAREA	Není	Ukazatel Null nebo bajty s hodnotou Null

Deklarace RPG

```
D* MQCTLO Structure
D*
D*
D* Structure identifier
D  COSID                  1      4    INZ('CTLO')
D*
D* Structure version number
D  COVER                  5      8I 0 INZ(1)
D*
D* Options that control the action of MQCTL
D  COOPT                  9     12I 0 INZ(0)
D*
D* Reserved
D  CORSV                 13     16I 0 INZ(-1)
D*
D* MQCTL Data area passed to the function
D  COCONNAREA            17     32*   INZ(*NULL)
```

IBM i MQDH (záhlaví distribuce) na IBM i

Struktura MQDH popisuje další data, která jsou přítomna ve zprávě v případě, že se jedná o zprávu distribučního seznamu uloženou v přenosové frontě.

Přehled

Účel: Zpráva rozdělovníku je zpráva, která je odeslána do více cílových front. Další data se skládají ze struktury MQDH následované polem záznamů MQOR a polem záznamů MQPMR.

Tato struktura je určena pro použití specializovanými aplikacemi, které vkládají zprávy přímo do přenosových front nebo které odebírají zprávy z přenosových front (například agenti kanálů zpráv).

Tato struktura by neměla být používána běžnými aplikacemi, které prostě chtějí vkládat zprávy do distribučních seznamů. Tyto aplikace by měly používat strukturu MQOD k definování cílů v distribučním seznamu a strukturu MQPMO k určení vlastností zprávy nebo přijímání informací o zprávách odesílaných do jednotlivých míst určení.

Znaková sada a kódování: Data v MQDH musí být ve znakové sadě určené atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT pro programovací jazyk C.

Znaková sada a kódování MQDH musí být nastaveny do polí *MDCSI* a *MDENC* v:

- MQMD (pokud je struktura MQDH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQDH (všechny ostatní případy).

Použití: Když aplikace vloží zprávu do distribučního seznamu a některá nebo všechna místa určení jsou vzdálená, správce front vloží před data zprávy aplikace struktury MQXQH a MQDH a umístí zprávu do příslušné přenosové fronty. Data se proto vyskytují v následující posloupnosti, když je zpráva v přenosové frontě:

- Struktura MQXQH
- Struktura MQDH plus pole záznamů MQOR a MQPMR
- Data zprávy aplikace

V závislosti na cílech může správce front vygenerovat více než jednu takovou zprávu a umístit ji do různých přenosových front. V tomto případě struktury MQDH v těchto zprávách identifikují různé podmnožiny cílů definovaných v distribučním seznamu otevřeném aplikací.

Aplikace, která vkládá zprávu distribučního seznamu přímo do přenosové fronty, musí odpovídat dříve popsané posloupnosti a musí zajistit, aby byla struktura MQDH správná. Není-li struktura MQDH platná, může se správce front rozhodnout pro selhání volání MQPUT nebo MQPUT1 s kódem příčiny RC2135.

Zprávy lze uložit do fronty ve formě distribučního seznamu pouze v případě, že je fronta definována jako schopná podporovat zprávy distribučního seznamu (viz atribut fronty **DistLists** popsaný v části “Atributy pro fronty” na stránce 1360). Pokud aplikace vloží zprávu distribučního seznamu přímo do fronty, která nepodporuje distribuční seznamy, správce front rozdělí zprávu distribučního seznamu na jednotlivé zprávy a umístí je do fronty.

- “Pole” na stránce 1057
- “Počáteční hodnoty” na stránce 1060
- “Deklarace RPG” na stránce 1061

Pole

Struktura MQDH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

DHCNT (10místné celé číslo se znaménkem)

Počet přítomných záznamů MQOR.

Tím se definuje počet cílů. Rozdělovník musí vždy obsahovat alespoň jeden cíl, takže *DHCNT* musí být vždy větší než nula.

Počáteční hodnota tohoto pole je 0.

DHCSI (10místné celé číslo se znaménkem)

Identifikátor znakové sady dat, která následují za záznamy MQOR a MQPMR.

Určuje identifikátor znakové sady dat, která následují za poli záznamů MQOR a MQPMR; nevztahuje se na znaková data v samotné struktuře MQDH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota CSINHT vrácena voláním MQGET.

CSINHT nelze použít, pokud je hodnota pole *MDPAT* v MQMD ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

DHENC (10místné celé číslo se znaménkem)

Číselné kódování dat, která následují za záznamy MQOR a MQPMR.

Tato volba určuje číselné kódování dat, která následují za poli záznamů MQOR a MQPMR; nevztahuje se na číselná data v samotné struktuře MQDH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

DHFLG (10místné celé číslo se znaménkem)

Obecné příznaky.

Lze zadat následující příznak:

DHFNOVÝ
Generovat nové identifikátory zpráv.

Tento příznak označuje, že má být vygenerován nový identifikátor zprávy pro každé místo určení v rozdělovníku. Tuto hodnotu lze nastavit pouze v případě, že nejsou k dispozici žádné záznamy vložených zpráv, nebo pokud jsou tyto záznamy přítomny, ale neobsahují pole *PRMID* .

Pomocí tohoto příznaku odkládá generování identifikátorů zpráv až do posledního možného okamžiku, tedy do okamžiku, kdy je zpráva rozdělovníku nakonec rozdělena na jednotlivé zprávy. Tím se minimalizuje množství řídicích informací, které musí proudit se zprávou distribučního seznamu.

Když aplikace vloží zprávu do distribučního seznamu, správce front nastaví DHFNEW v MQDH, který vygeneruje, když jsou oba následující příkazy pravdivé:

- Aplikace neposkytla žádné záznamy vložených zpráv, nebo poskytnuté záznamy neobsahují pole *PRMID* .
- Pole *MDMID* v MQMD je MINONE nebo pole *PMOPT* v MQPMO obsahuje PMNMID

Nejsou-li potřeba žádné příznaky, lze zadat následující:

DHFNON
Žádné příznaky.

Tato konstanta označuje, že nebyly zadány žádné příznaky. DHFNON je definována jako pomocná programová dokumentace. Není zamýšleno, aby se tato konstanta používala s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Počáteční hodnota tohoto pole je DHFNON.

DHFMT (8bajtový znakový řetězec)

Název formátu dat, která následují za záznamy MQOR a MQPMR.

Tato volba určuje název formátu dat, která následují za poli záznamů MQOD a MQPMR (podle toho, co nastane později).

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *MDFMT* v MQMD.

Počáteční hodnota tohoto pole je FMNONE.

DHLEN (10místné celé číslo se znaménkem)

Délka struktury MQDH plus následující záznamy MQOR a MQPMR.

Jedná se o počet bajtů od začátku struktury MQDH po začátek dat zprávy následujících po polích záznamů MQOR a MQPMR. Data se vyskytují v následujícím pořadí:

- Struktura MQDH
- Pole záznamů MQOR
- Pole záznamů MQPMR
- Data zprávy

Pole záznamů MQOR a MQPMR jsou adresována offsety obsaženými ve struktuře MQDH. Pokud tyto odchylky vedou k nepoužitým bajtům mezi jednou nebo více strukturami MQDH, poli záznamů a daty zpráv, musí být tyto nepoužívané bajty zahrnuty do hodnoty *DHLEN*, ale obsah těchto bajtů není správcem front zachován. Pole záznamů MQPMR je platné před polem záznamů MQOR.

Počáteční hodnota tohoto pole je 0.

DHORO (10místné celé číslo se znaménkem)

Posunutí prvního záznamu MQOR od začátku MQDH.

V tomto poli je uveden posun prvního záznamu v poli záznamů objektů MQOR obsahujících názvy cílových front v bajtech. V tomto poli jsou záznamy *DHCNT* . Tyto záznamy (plus všechny bajty přeskočené mezi prvním záznamem objektu a předchozím polem) jsou zahrnuty do délky dané polem *DHLEN* .

Rozdělovník musí vždy obsahovat alespoň jeden cíl, takže *DHORO* musí být vždy větší než nula.

Počáteční hodnota tohoto pole je 0.

DHPRF (10místné celé číslo se znaménkem)

Příznaky označující, která pole MQPMR jsou přítomna.

Lze zadat nula nebo více následujících příznaků:

PFMID
Pole identifikátoru zprávy je přítomno.

PFCID
Pole identifikátoru korelace je přítomno.

PFGID
Pole identifikátoru skupiny je přítomno.

PFFB
Pole zpětné vazby je přítomno.

PFACC
Pole tokenu evidence je přítomno.

Nejsou-li uvedena žádná pole MQPMR, lze zadat následující:

PFNONE
Nejsou přítomna žádná pole záznamu vložená zpráva.

PFNONE je definován jako pomocná programová dokumentace. Není zamýšleno, aby tato konstanta byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Počáteční hodnota tohoto pole je PFNONE.

DHPRO (10místné celé číslo se znaménkem)

Posunutí prvního záznamu MQPMR od začátku MQDH.

V tomto poli je uveden posun v bajtech prvního záznamu v poli záznamů vložených zpráv MQPMR, které obsahují vlastnosti zprávy. Je-li přítomen, v tomto poli jsou záznamy *DHCNT* . Tyto záznamy (plus všechny bajty přeskočené mezi prvním záznamem vkládané zprávy a předchozím polem) jsou zahrnuty do délky dané polem *DHLEN* .

Záznamy vkládané zprávy jsou volitelné; pokud nejsou poskytnuty žádné záznamy, *DHPRO* je nula a *DHPRF* má hodnotu PFNONE.

Počáteční hodnota tohoto pole je 0.

DHSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

DHSIDV
Identifikátor pro strukturu záhlaví distribuce.

Počáteční hodnota tohoto pole je DHSIDV.

DHVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

DHVER1
Číslo verze pro strukturu záhlaví distribuce.

Následující konstanta určuje číslo verze aktuální verze:

DHVERC
Aktuální verze struktury záhlaví distribuce.

Počáteční hodnota tohoto pole je DHVER1.

Počáteční hodnoty

Tabulka 696. Pole v MQDH

Název pole	Název konstanty	Hodnota konstanty
DHSID	DHSIDV	'DH¬¬'
DHVER	DHVER1	1
DHLEN	Není	0
DHENC	Není	0
DHCSI	CSUNDF	0
DHFMT	FMNONE	Mezery
DHFLG	DHFNON	0
DHPRF	PFNONE	0

Tabulka 696. Pole v MQDH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
DHCNT	Není	0
DHORO	Není	0
DHPRO	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQDH Structure
D*
D* Structure identifier
D  DHSID                  1      4    INZ('DH  ')
D* Structure version number
D  DHVER                  5      8I 0 INZ(1)
D* Length of MQDH structure plusfollowing MQOR and MQPMR records
D  DHLEN                  9     12I 0 INZ(0)
D* Numeric encoding of data that followsthe MQOR and MQPMR records
D  DHENC                 13     16I 0 INZ(0)
D* Character set identifier of data thatfollows the MQOR and MQPMR
D* records
D  DHCSI                 17     20I 0 INZ(0)
D* Format name of data that follows theMQOR and MQPMR records
D  DHFMT                 21     28    INZ('        ')
D* General flags
D  DHFLG                 29     32I 0 INZ(0)
D* Flags indicating which MQPMR fieldsare present
D  DHPRF                 33     36I 0 INZ(0)
D* Number of MQOR records present
D  DHCNT                 37     40I 0 INZ(0)
D* Offset of first MQOR record from startof MQDH
D  DHORO                 41     44I 0 INZ(0)
D* Offset of first MQPMR record fromstart of MQDH
D  DHPRO                 45     48I 0 INZ(0)
```

IBM i MQDLH (záhlaví nedoručených dopisů) na IBM i

Přehled

Účel

Struktura MQDLH popisuje informace, které jsou před daty zpráv aplikace ve frontě nedoručených zpráv (nedoručených zpráv). Zpráva může být doručena do fronty nedoručených zpráv, protože ji správce front nebo agent kanálu zpráv přesměrovali do fronty. Aplikace může vložit zprávu přímo do fronty.

Název formátu

FMDLH

Znaková sada a kódování

MQDLH může být na začátku dat zprávy aplikace. Pokud ano, pole ve struktuře MQDLH jsou ve znakové sadě a kódování dané poli MDCSI a MDENC . Pokud ne, znaková sada a kódování jsou nastaveny pomocí polí MDCSI a MDENC ve struktuře záhlaví, která předchází MQDLH.

Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech front.

Pouľití

Aplikace, které vkládají zprávy přímo do fronty nedoručených zpráv, musí před data zprávy vložit strukturu MQDLH a inicializovat pole s odpovídajícími hodnotami. Správce front však nevyžaduje přítomnost struktury MQDLH nebo zadání platných hodnot pro pole.

Pokud je zpráva příliš dlouhá na to, aby ji bylo možné vložit do fronty nedoručených zpráv, musí aplikace zvážit jednu z následujících možností:

- Oříznětě data zprávy tak, aby se vešla do fronty nedoručených zpráv.
- Zaznamenejte zprávu do pomocné paměti a umístěte zprávu zprávy o výjimce do fronty nedoručených zpráv, která označuje, že zpráva je příliš dlouhá.
- Zahoďte zprávu a vraťte chybu jejímu původci. Pokud se jedná o kritickou zprávu. Zprávu zrušte pouze v případě, že je známo, že původce stále má kopii zprávy. Například zpráva přijatá agentem kanálu zpráv z komunikačního kanálu.

Která z možností je vhodná, závisí na návrhu aplikace.

Správce front provádí speciální zpracování, když je zpráva, která je segmentem, vložena se strukturou MQDLH na přední straně. Další podrobnosti viz popis struktury MQMDE .

- "Vkládání zpráv do fronty nedoručených zpráv" na stránce 1062
- "Získávání zpráv z fronty nedoručených zpráv" na stránce 1063
- "Pole" na stránce 1063
- "Počáteční hodnoty" na stránce 1066
- "Deklarace RPG" na stránce 1066

Vkládání zpráv do fronty nedoručených zpráv

Pokud je zpráva vložena do fronty nedoručených zpráv, struktura MQMD použitá pro volání MQPUT nebo MQPUT1 musí být stejná jako struktura MQMD přidružená ke zprávě. MQMD je obvykle ten, který vrací volání MQGET , s výjimkou následujících případů:

- Pole MDCSI a MDENC musí být nastavena na jakoukoli znakovou sadu a kódování použité pro pole ve struktuře MQDLH .
- Pole MDFMT musí být nastaveno na hodnotu FMDLH , aby indikoval, že data začínají strukturou MQDLH .
- Kontextová pole MDACC, MDAID, MDAOD, MDPAN, MDPAT, MDPD, MDPTa MDUID musí být nastavena pomocí kontextové volby odpovídající okolnostem:
  - Aplikace vkládající do fronty nedoručených zpráv zprávu, která nesouvisí s žádnou předchozí zprávou, musí použít volbu PMDEFC . Volba PMDEFC způsobí, že správce front nastaví všechna pole kontextu v deskriptoru zprávy na výchozí hodnoty.
  - Serverová aplikace vkládající do fronty nedoručených zpráv zprávu, kterou přijala, musí použít volbu PMPASA , aby zachovala původní informace o kontextu.
  - Serverová aplikace, která vkládá do fronty nedoručených zpráv odpověď na přijatou zprávu, musí použít volbu PMPASI . Volba PMPASI zachová informace o identitě, ale nastaví informace o původu na informace serverové aplikace.
  - Agent kanálu zpráv vkládající do fronty nedoručených zpráv zprávu, kterou přijal od svého komunikačního kanálu, musí použít volbu PMSETA . Volba PMSETA zachová původní informace o kontextu.

V samotné struktuře MQDLH musí být pole nastavena takto:

- Pole DLCSI, DLENCa *DLFMT* musí být nastavena na hodnoty popisující data, která odpovídají struktuře MQDLH . Tyto hodnoty jsou obvykle hodnoty z původního deskriptoru zprávy.
- Kontextová pole DLPAT, DLPAN, DLPDa DLPT musí být nastavena na hodnoty odpovídající aplikaci, která vkládá zprávu do fronty nedoručených zpráv. Tyto hodnoty nesouvisí s původní zprávou.
- Ostatní pole musí být nastavena podle potřeby.

Aplikace musí zajistit, aby všechna pole měla platné hodnoty a aby znaková pole byla vyplněna mezerami na definovanou délku pole. Znaková data nesmí být předčasně ukončena pomocí znaku null. Správce front nepřevádí znaky null a následné znaky na mezery ve struktuře MQDLH .

Získávání zpráv z fronty nedoručených zpráv

Aplikace, které získají zprávy z fronty nedoručených zpráv, musí ověřit, že zprávy začínají strukturou MQDLH . Aplikace může určit, zda je přítomna struktura MQDLH , tak, že prozkoumá pole MDFMT v deskriptoru zprávy MQMD. Pokud má pole hodnotu FMDLH, data zprávy začínají strukturou MQDLH . Zprávy ve frontě nedoručených zpráv mohou být oříznuty, pokud byly původně příliš dlouhé pro frontu, pro kterou byly určeny.

Pole

Struktura MQDLH obsahuje následující pole; pole jsou popsána v abecedním pořadí:

DLCSI (10místné celé číslo se znaménkem)

Identifikátor znakové sady dat, která následují za MQDLH.

DLCSI uvádí identifikátor znakové sady dat, která se řídí strukturou MQDLH . Data jsou obvykle z původní zprávy. Nevztahuje se na znaková data v samotné struktuře MQDLH .

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech následujících za touto strukturou jsou ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota CSINHT vrácena voláním funkce MQGET .

CSINHT nelze použít, pokud je hodnota pole MDPAT v MQMD ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

DLDM (48bajtový znakový řetězec)

Název původního cílového správce front.

Jedná se o název správce front, který byl původním cílem zprávy.

Délka tohoto pole je dána hodnotou LNQMN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

DLDQ (48bajtový znakový řetězec)

Název původní cílové fronty.

Jedná se o název fronty zpráv, která byla původním cílem zprávy.

Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

DLENC (10místné celé číslo se znaménkem)

Číselné kódování dat, která následují za MQDLH.

DLENC uvádí číselné kódování dat, která se řídí strukturou MQDLH . Data jsou obvykle z původní zprávy. Nevztahuje se na číselná data v samotné struktuře MQDLH .

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

DLFMT (8bajtový znakový řetězec)

Název formátu dat, která následují za MQDLH.

Uvádí název formátu dat, která se řídí strukturou MQDLH (obvykle data z původní zprávy).

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro pole MDFMT v produktu MQMD.

Délka tohoto pole je dána hodnotou LNFMT. Počáteční hodnota tohoto pole je FMNONE.

DLPAN (28bajtový znakový řetězec)

Název aplikace, která vložila zprávu do fronty nedoručených zpráv.

Formát názvu závisí na poli DLPAT . Viz popis pole MDPAN v souboru “MQMD (deskriptor zprávy) na IBM i” na stránce 1105.

Jedná-li se o správce front, který přesměruje zprávu do fronty nedoručených zpráv, obsahuje DLPAN prvních 28 znaků názvu správce front. Název je v případě potřeby vyplněn mezerami.

Délka tohoto pole je dána hodnotou LNPAN. Počáteční hodnota tohoto pole je 28 prázdných znaků.

DLPAT (10místné celé číslo se znaménkem)

Typ aplikace, která vložila zprávu do fronty nedoručených zpráv.

Toto pole má stejný význam jako pole MDPAT v deskriptoru zprávy MQMD (podrobnosti viz “MQMD (deskriptor zprávy) na IBM i” na stránce 1105).

Pokud je to správce front, který přesměruje zprávu do fronty nedoručených zpráv, DLPAT má hodnotu ATQM.

Počáteční hodnota tohoto pole je 0.

DLPD (8bajtový znakový řetězec)

Datum, kdy byla zpráva vložena do fronty nedoručených zpráv.

Formát použitý pro datum, kdy je toto pole generováno správcem front, je:

- YYYYMMDD

kde znaky představují:

YYYY
rok (čtyři číslice)

MM
měsíc v roce (01 až 12)

DD
den v měsíci (01 až 31)

Greenwichský střední čas (GMT) se používá pro pole DLPD a DLPT za předpokladu, že systémový čas je přesně nastaven na GMT.

Délka tohoto pole je dána hodnotou LNPDAT. Počáteční hodnota tohoto pole je osm prázdných znaků.

DLPT (8bajtový znakový řetězec)

Čas, kdy byla zpráva vložena do fronty nedoručených zpráv.

Formát použitý pro čas, kdy je toto pole generováno správcem front, je:

- HHMMSSTH

kde znaky představují (v pořadí):

HH
hodiny (00 až 23)

MM
minut (00 až 59)

SS
sekundy (00 až 59; viz poznámka později v tomto tématu)

T
desetiny sekundy (0 až 9)

H
setiny sekundy (0 až 9)

Poznámka: Pokud jsou systémové hodiny synchronizovány s přesným časovým standardem, je možné, že 60 nebo 61 bude vráceno pro sekundy v DLPT. Další sekunda se vyskytne, když jsou do globálního časového standardu vloženy přestupné sekundy.

Greenwichský střední čas (GMT) se používá pro pole DLPD a DLPT za předpokladu, že systémový čas je přesně nastaven na GMT.

Délka tohoto pole je dána hodnotou LNPTIM. Počáteční hodnota tohoto pole je osm prázdných znaků.

DLREA (10místné celé číslo se znaménkem)

Do fronty nedoručených zpráv (nedoručených zpráv) byla přijata zpráva o příčině.

To identifikuje příčinu, proč byla zpráva umístěna do fronty nedoručených zpráv místo do původní cílové fronty. Musí to být jedna z hodnot FB* nebo RC* (například RC2053). Podrobnosti o společných hodnotách FB*, které se mohou vyskytnout, naleznete v popisu pole *MDFB* v souboru "MQMD (deskriptor zprávy) na IBM i" na stránce 1105 .

Pokud je hodnota v rozsahu FBIFST až FBILST, skutečný kód chyby IMS lze určit odečtením FBIERR od hodnoty pole *DLREA* .

Některé hodnoty FB* se vyskytují pouze v tomto poli. Vztahují se ke zprávám úložiště, zprávám spouštěče nebo zprávám přenosové fronty, které jsou přeneseny do fronty nedoručených zpráv. Tyto hodnoty jsou:

FBABEG
Aplikaci nelze spustit.

Aplikace zpracovávající zprávu spouštěče nemohla spustit aplikaci uvedenou v poli TMAI zprávy spouštěče; viz "MQTM-zpráva spouštěče" na stránce 1229.

FBATYP
Chyba typu aplikace.

Aplikace zpracovávající zprávu spouštěče nemohla spustit aplikaci, protože pole TMAT zprávy spouštěče je neplatné; viz "MQTM-zpráva spouštěče" na stránce 1229.

FBBOCD
Přijímací kanál klastru byl odstraněn.

Zpráva se nacházela v přenosové frontě klastru určené pro frontu klastru, která byla otevřena s volbou FBIERR . Kanál příjemce vzdáleného klastru, který má být použit pro přenos zprávy do cílové fronty, byl odstraněn před odesláním zprávy. Vzhledem k tomu, že byla zadána volba FBIERR , lze k přenosu zprávy použít pouze kanál vybraný při otevření fronty. Vzhledem k tomu, že tento kanál již není k dispozici, byla zpráva umístěna do fronty nedoručených zpráv.

FBNARM
Zpráva není zprávou úložiště.

FBSBCX
Zpráva byla zastavena uživatelskou procedurou automatické definice kanálu.

FBSBMX
Zpráva byla zastavena uživatelskou procedurou pro zprávy kanálu.

FBTM
Struktura MQTM není platná nebo chybí.

Pole MDFMT v poli MQMD uvádí FMTM, ale zpráva nezačíná platnou strukturou MQTM . Například mnemonický poutač *TMSID* nemusí být platný. Soubor *TMVER* nemusí být rozpoznán. Délka zprávy spouštěče může být nedostatečná pro to, aby obsahovala strukturu MQTM .

FBXQME
Zpráva v přenosové frontě není ve správném formátu.

Agent kanálu zpráv zjistil, že zpráva v přenosové frontě není ve správném formátu. Agent kanálu zpráv vloží zprávu do fronty nedoručených zpráv pomocí tohoto kódu zpětné vazby.

Počáteční hodnota tohoto pole je RCNONE.

DLSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

DLSIDV
Identifikátor pro strukturu záhlaví nedoručených dopisů.

Počáteční hodnota tohoto pole je DLSIDV.

DLVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

DLVER1
Číslo verze pro strukturu záhlaví nedoručených dopisů.

Následující konstanta určuje číslo verze aktuální verze:

DLVERC
Aktuální verze struktury záhlaví nedoručených dopisů.

Počáteční hodnota tohoto pole je DLVER1.

Počáteční hodnoty

Tabulka 697. Pole v souboru MQDLH

Název pole	Název konstanty	Hodnota konstanty
DLSID	DLSIDV	'DLH¬'
DLVER	DLVER1	1
DLREA	RCNONE	0
DLDQ	Není	Mezery
DLDM	Není	Mezery
DLENC	Není	0
DLCSI	CSUNDF	0
DLFMT	FMNONE	Mezery
DLPAT	Není	0
DLPAN	Není	Mezery
DLPD	Není	Mezery
DLPT	Není	Mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQDLH Structure
D*
D* Structure identifier
D  DLSID                  1      4    INZ('DLH ')
D* Structure version number
D  DLVER                  5      8I 0 INZ(1)
```

```
     D* Reason message arrived on dead-letter(undelivered-message) queue
     D  DLREA                  9     12I 0 INZ(0)
     D* Name of original destination queue
     D  DLDQ                  13     60    INZ
     D* Name of original destination queue manager
     D  DLDM                  61    108    INZ
     D* Numeric encoding of data that followsMQDLH
     D  DLENC                109    112I 0 INZ(0)
     D* Character set identifier of data thatfollows MQDLH
     D  DLCSI                113    116I 0 INZ(0)
     D* Format name of data that followsMQDLH
     D  DLFMT                117    124    INZ('        ')
     D* Type of application that put messageon dead-letter
     D* (undelivered-message)queue
     D  DLPAT                125    128I 0 INZ(0)
     D* Name of application that put messageon dead-letter
     D* (undelivered-message)queue
     D  DLPAN                129    156    INZ
     D* Date when message was put ondead-letter (undelivered-message)queue
     D  DLPD                 157    164    INZ
     D* Time when message was put on thedead-letter (undelivered-message)queue
     D  DLPT                 165    172    INZ
```

IBM i MQDMHO (Odstranění voleb popisovače zprávy) v systému IBM i

Struktura **MQDMHO** umožňuje aplikacím určit volby, které řídí způsob odstranění manipulátorů zpráv.

Přehled

Účel: Struktura je vstupní parametr volání **MQDLTMH** .

Znaková sada a kódování: Data v **MQDMHO** musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1067
- "Počáteční hodnoty" na stránce 1068
- "Deklarace RPG" na stránce 1068

Pole

Struktura MQDMHO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

DMOPT (10místné celé číslo se znaménkem)

Hodnota musí být:

DMNONE

Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **DMNONE**.

DMSID (10místné celé číslo se znaménkem)

Toto je identifikátor struktury; hodnota musí být:

DMSIDV

Identifikátor pro strukturu voleb popisovače zprávy odstranění.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **DMSIDV**.

DMVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury; hodnota musí být:

DMVER1
Version-1 odstraňte strukturu voleb popisovače zprávy.

Následující konstanta určuje číslo verze aktuální verze:

DMVERC
Aktuální verze struktury voleb popisovače zprávy odstranění.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **DMVER1**.

Počáteční hodnoty

Tabulka 698. Pole v MQDMHO		
Název pole	**Název konstanty**	**Hodnota konstanty**
DMSID	DMSIDV	'DMHO'
DMVER	DMVER1	1
DMOPT	DMNONE	0

Deklarace RPG

```
D* MQDMHO Structure
D*
D*
D* Structure identifier
D  DMSID                  1      4    INZ('DMHO')
D*
D* Structure version number
D  DMVER                  5      8I 0 INZ(1)
D*
D* Options that control the action of MQDLTMH
D  DMOPT                  9     12I 0 INZ(0)
```

IBM i MQDMPO (Odstranění voleb vlastností zprávy) na IBM i

Struktura definující volby vlastnosti odstranění zprávy.

Přehled

Účel: Struktura MQDMPO umožňuje aplikacím určit volby, které řídí způsob odstraňování vlastností zpráv. Struktura je vstupní parametr volání MQDLTMP.

Znaková sada a kódování: Data v MQDMPO musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1068
- "Počáteční hodnoty" na stránce 1069
- "Deklarace RPG" na stránce 1069

Pole

Struktura MQDMPO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

DPOPT (10místné celé číslo se znaménkem)

Struktura voleb vlastností zprávy odstranění-pole DPOPT.

Volby umístění: Následující volby se vztahují k relativnímu umístění vlastnosti v porovnání s kurzorem vlastnosti.

DPDELF
Odstraní první vlastnost, která odpovídá zadanému názvu.

DPDELC
Odstraní vlastnost, na kterou ukazuje kurzor vlastnosti; to je vlastnost, která byla naposledy dotazována pomocí volby IPINQF nebo IPINQN.

Kurzor vlastnosti se resetuje při opětovném použití popisovače zprávy. Resetuje se také, když je popisovač zprávy uveden v poli *HMSG* MQGMO ve volání MQGET nebo ve struktuře MQPMO ve volání MQPUT.

Kurzor vlastnosti se resetuje při opětovném použití manipulátoru zprávy nebo při zadání manipulátoru zprávy v poli *HMSG* struktury MQGMO ve struktuře MQGET ve volání MQGET nebo ve struktuře MQPMO ve volání MQPUT.

Volání selže s kódem dokončení CCFAIL a důvodem RC2471 , pokud je tato volba použita, když kurzor vlastnosti ještě nebyl zaveden. Také se nezdaří s těmito kódy, pokud vlastnost, na kterou ukazuje kurzor vlastnosti, již byla odstraněna.

Pokud není požadována žádná z těchto voleb, lze použít následující volbu:

DPNONE
Nejsou uvedeny žádné volby.

Počáteční hodnota tohoto vstupního pole je DPDELF.

DPSID (10místné celé číslo se znaménkem)

Odstranit strukturu voleb vlastností zprávy-pole DPSID.

Jedná se o identifikátor struktury. Hodnota musí být:

DPSIDV
Identifikátor pro strukturu voleb vlastností zprávy odstranění.

Toto pole je vždy vstupní pole. Počáteční hodnota tohoto pole je DPSIDV.

DPVER (10místné celé číslo se znaménkem)

Struktura voleb vlastností zprávy odstranění-pole DPVER.

Toto je číslo verze struktury. Hodnota musí být:

DPVER1
Číslo verze pro strukturu voleb vlastností zprávy odstranění.

Následující konstanta určuje číslo verze aktuální verze:

DPVERC
Aktuální verze struktury voleb vlastností zprávy odstranění.

Toto pole je vždy vstupní pole. Počáteční hodnota tohoto pole je DPVER1.

Počáteční hodnoty

Tabulka 699. Pole v MQDPMO

Název pole	Název konstanty	Hodnota konstanty
DPSID	DPSIDV	'DMPO'
DPVER	DPVER1	1
DPOPT	Volby, které řídí akci MQDLTMP	DPNONE

Deklarace RPG

```
D* MQDMPO Structure
D*
D*
D* Structure identifier
```

```
     D  DPSID                  1      4    INZ('DMPO')
     D*
     D* Structure version number
     D  DPVER                  5      8I 0 INZ(1)
     D*
     ** Options that control the action of
     D* MQDLTMP
     D  DPOPT                  9     12I 0 INZ(0)
```

IBM i MQEPH (záhlaví Embedded PCF) na systému IBM i

Přehled

Účel

Struktura MQEPH popisuje další data, která jsou přítomna ve zprávě, když je tato zpráva ve formátu programovatelného příkazu (PCF). Pole *EPPFH* definuje parametry PCF, které se řídí touto strukturou, a to vám umožňuje sledovat data zprávy PCF s jinými záhlavími.

Název formátu

EPFMT

Znaková sada a kódování

Data v MQEPH musí být ve znakové sadě a kódování lokálního správce front; to je dáno atributem správce front **CCSID** .

Nastavte znakovou sadu a kódování MQEPH do polí *MDCSI* a *MDENC* v:

- MQMD (pokud je struktura MQEPH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQEPH (všechny ostatní případy).

Použití

Struktury MQEPH nelze použít k odesílání příkazů příkazovému serveru nebo jinému serveru akceptujícího PCF správce front.

Podobně příkazový server ani jiný správce front PCF-akceptující server negenerují odpovědi ani události obsahující struktury MQEPH.

- "Pole" na stránce 1070
- "Počáteční hodnoty" na stránce 1072
- "Deklarace RPG" na stránce 1072

Pole

Struktura MQEPH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

EPCSI (10místné celé číslo se znaménkem)

Jedná se o identifikátor znakové sady dat, která se řídí strukturou MQEPH a přidruženými parametry PCF; nevztahuje se na znaková data v samotné struktuře MQEPH.

Počáteční hodnota tohoto pole je EPCUND.

EPENC (10místné celé číslo se znaménkem)

Jedná se o číselné kódování dat, která se řídí strukturou MQEPH a přidruženými parametry PCF; nevztahuje se na znaková data v samotné struktuře MQEPH.

Počáteční hodnota tohoto pole je 0.

EPFLG (10místné celé číslo se znaménkem)

K dispozici jsou tyto hodnoty:

EPNONE
Nebyly zadány žádné příznaky. *MDCSI* Produkt EPNONE je definován jako pomůcka pro dokumentaci k programu. Není zamýšleno, aby tato konstanta byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

EPCSEM
Znaková sada parametrů obsahujících znaková data je v každé struktuře v poli *CCSID* uvedena jednotlivě. Znaková sada polí *EPSID* a *EPFMT* je definována pomocí *CCSID* ve struktuře záhlaví, která předchází struktuře MQEPH, nebo pomocí pole *MDCSI* v deskriptoru MQMD, pokud je MQEPH na začátku zprávy.

Počáteční hodnota tohoto pole je EPNONE.

EPFMT (8bajtový znakový řetězec)

Jedná se o název formátu dat, která následují za strukturou MQEPH a přidruženými parametry PCF.

Počáteční hodnota tohoto pole je EPFMNO.

EPLEN (10místné celé číslo se znaménkem)

Jedná se o množství dat před další strukturou záhlaví. Zahrnuje:

- Délka záhlaví MQEPH
- Délka všech parametrů PCF následujících za záhlavím
- Jakákoli prázdná výplň za těmito parametry

EPLEN musí být násobkem 4.

Část konstrukce s pevnou délkou je definována pomocí EPSTLF.

Počáteční hodnota tohoto pole je 68.

EPPCFH (MQCFH)

Toto je záhlaví formátu programovatelného příkazu (PCF) definující parametry PCF, které se řídí strukturou MQEPH. To vám umožní sledovat data zprávy PCF s dalšími záhlavími.

Záhlaví PCF je na počátku definováno s následujícími hodnotami:

Tabulka 700. Pole v EPPCFH

Název pole	Název konstanty	Hodnota konstanty
EP3TYP	CFTNON	0
EP3LEN	FHLENV	36
EP3VER	FHVER3	3
EP3CMD	CMNONE	0
EP3SEQ	Není	1
EP3CTL	CFCLST	1
EEP3CC	CCOK	0
EP3REA	RCNONE	0
EP3CNT	Není	0

Aplikace musí změnit EP3TYP z CFTNON na platný typ struktury pro použití vloženého záhlaví PCF.

EPSID (4bajtový znakový řetězec)

Hodnota musí být:

EPSTID
Identifikátor pro strukturu záhlaví Embedded PCF.

Počáteční hodnota tohoto pole je EPSTID.

EPVER (10místné celé číslo se znaménkem)

Hodnota může být následující:

EPVER1
Číslo verze pro vloženou strukturu záhlaví PCF.

Následující konstanta určuje číslo verze aktuální verze:

EPVER3
Aktuální verze vložené struktury záhlaví PCF.

Počáteční hodnota tohoto pole je EPVER3.

Počáteční hodnoty

Tabulka 701. Pole v MQEPH

Název pole	Název konstanty	Hodnota konstanty
EPSID	EPSTID	'EP¬¬'
EPVER	EPVER1	1
EPLEN	EPSTLF	68
EPENC	Není	0
EPCSI	EPCUND	0
EPFMT	EPFMNO	Mezery
EPFLG	EPNONE	0
EPPCFH	Názvy a hodnoty definované v souboru Tabulka 700 na stránce 1071	0

Poznámka:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQEPH Structure
D*
D* Structure identifier
D  EPSID                  1      4
D* Structure version number
D  EPVER                  5      8I 0
D* Total lenght of MQEPH including MQCFHand parameter structures
D* that follow
D  EPLEN                  9     12I 0
D* Numeric encoding of data that follows last PCF parameter structure
D  EPENC                 13     16I 0
D* Character set identifier of data that follows last PCF parameter
D* structure
D  EPCSI                 17     20I 0
D* Format name of data that follows last PCF parameter structure
D  EPFMT                 21     28
D* Flags
D  EPFLG                 29     32I 0
D* Programmable Command Format Header
D  EP3TYP                33     36I 0
D  EP3LEN                37     40I 0
D  EP3VER                41     44I 0
D  EP3CMD                45     48I 0
D  EP3SEQ                49     52I 0
D  EP3CTL                53     56I 0
```

```
     D  EP3CC                 57     60I 0
     D  EP3REA                61     64I 0
     D  EP3CNT                65     68I 0
```

IBM i MQGMO (Volby Get-message) na systému IBM i

Struktura MQGMO umožňuje aplikaci určit volby, které řídí způsob odebírání zpráv z front.

Přehled

Účel
Struktura je vstupní/výstupní parametr volání MQGET.

Verze
Aktuální verze MQGMO je GMVER4. Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi MQGMO, která je podporována prostředím, ale s počáteční hodnotou pole *GMVER* nastavenou na GMVER1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole *GMVER* na číslo verze požadované verze.

Znaková sada a kódování
Data v MQGMO musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , musí být struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1073
- "Počáteční hodnoty" na stránce 1092
- "Deklarace RPG" na stránce 1093

Pole

Struktura MQGMO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

GMGST (1bajtový znakový řetězec)

Příznak označující, zda je načtená zpráva ve skupině.

Má jednu z následujících hodnot:

GSNIG
Zpráva není ve skupině.

GSMIG
Zpráva je ve skupině, ale není poslední ve skupině.

GSLMIG
Zpráva je poslední ve skupině.

Tato hodnota je také vrácená hodnota, pokud se skupina skládá pouze z jedné zprávy.

Toto pole je výstupní pole. Počáteční hodnota tohoto pole je GSNIG. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER2.

GMMH (10místné celé číslo se znaménkem)

popisovač zprávy

Pokud je uvedena volba GMPRAQ a atribut fronty PRPCTL není nastaven na PRPRPRFH, pak je to popisovač zprávy, která je naplněna vlastnostmi zprávy načítané z fronty. Manipulátor je vytvořen voláním MQCRTMH. Všechny vlastnosti, které jsou již k popisovači přidruženy, jsou před načtením zprávy vymazány.

Lze také zadat následující hodnotu:
MQHM_NONE
Nebyl dodán žádný popisovač zprávy.

Pro volání MQGET není vyžadován žádný deskriptor zprávy, pokud je zadán platný manipulátor zprávy a používá se na výstupu, aby obsahoval vlastnosti zprávy, použije se deskriptor zprávy přidružený k manipulátoru zprávy pro vstupní pole.

Je-li ve volání MQGET určen deskriptor zprávy, má vždy přednost před deskriptorem zprávy přidruženým k manipulátoru zprávy.

Pokud je uveden parametr GMPRRF nebo GMPRAQ a atribut fronty PRPCTL je PRPRPRFH, pak volání selže s kódem příčiny RC2026 , když není uveden žádný parametr deskriptoru zprávy.

Při návratu z volání MQGET jsou vlastnosti a deskriptor zprávy přidružené k tomuto popisovači zprávy aktualizovány tak, aby odrážely stav načtené zprávy (stejně jako deskriptor zprávy, pokud byl zadán ve volání MQGET). Vlastnosti zprávy lze poté zjišťovat pomocí volání MQINQMP.

S výjimkou rozšíření deskriptoru zpráv není vlastnost, kterou lze požadovat pomocí volání MQINQMP, obsažena v datech zprávy; pokud zpráva ve frontě obsahovala vlastnosti v datech zprávy, jsou tyto vlastnosti odebrány z dat zprávy před vrácením dat do aplikace.

Není-li zadán žádný popisovač zprávy nebo je-li verze menší než GMVER4 , je třeba zadat platný deskriptor zprávy pro volání MQGET. Všechny vlastnosti zprávy (kromě vlastností obsažených v deskriptoru zprávy) jsou vráceny v předmětu dat zprávy s hodnotou voleb vlastností ve struktuře MQGMO a atributu fronty PRPCTL.

Toto pole je vždy vstupní pole. Počáteční hodnota tohoto pole je HMNONE. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER4.

GMMO (10místné celé číslo se znaménkem)

Volby určující kritéria výběru používaná pro příkaz MQGET.

Tyto volby umožňují aplikaci zvolit, která pole v parametru **MSGDSC** se použijí k výběru zprávy vrácené voláním MQGET. Aplikace nastaví požadované volby v tomto poli a poté nastaví odpovídající pole v parametru **MSGDSC** na hodnoty požadované pro tato pole. Pouze zprávy, které mají tyto hodnoty v deskriptoru MQMD pro zprávu, jsou kandidáty pro načtení pomocí tohoto parametru **MSGDSC** ve volání MQGET. Pole, pro která není zadána odpovídající volba shody, jsou při výběru zprávy, která má být vrácena, ignorována. Pokud ve volání MQGET nejsou použita žádná kritéria výběru (to znamená, že jakákoli zpráva je přijatelná), měl by být parametr *GMMO* nastaven na hodnotu MONONE.

Je-li zadána hodnota GMLOGO, jsou pro návrat při příštím volání MQGET vhodné pouze určité zprávy:

- Pokud neexistuje žádná aktuální skupina nebo logická zpráva, jsou pro návrat vhodné pouze zprávy, které mají *MDSEQ* rovno 1 a *MDOFF* rovno 0 . V této situaci lze použít jednu nebo více následujících voleb k výběru, která z vhodných zpráv je vrácena:
  - MOMSGI
  - MOCORI
  - MOGRPI
- Pokud existuje aktuální skupina nebo logická zpráva, pouze další zpráva ve skupině nebo další segment v logické zprávě je způsobilá pro návrat, a to nelze změnit uvedením voleb MO*.

V obou případech lze stále uvést volby shody, které nelze použít, ale hodnota příslušného pole v parametru **MSGDSC** se musí shodovat s hodnotou odpovídajícího pole ve zprávě, která se má vrátit; volání selže s kódem příčiny RC2247 je, že tato podmínka není splněna.

Parametr *GMMO* je ignorován, pokud je zadán parametr GMMUC nebo GMBRWC.

Lze zadat jednu nebo více následujících voleb:

MOMSGI
Načíst zprávu s uvedeným identifikátorem zprávy.

Tato volba určuje, že zpráva, která má být načtena, musí mít identifikátor zprávy, který odpovídá hodnotě pole *MDMID* v parametru **MSGDSC** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například identifikátor korelace).

Není-li tato volba uvedena, pole *MDMID* v parametru **MSGDSC** se ignoruje a libovolný identifikátor zprávy se shoduje.

Poznámka: Identifikátor zprávy MINONE je speciální hodnota, která odpovídá libovolnému identifikátoru zprávy v deskriptoru MQMD pro zprávu. Proto je uvedení MOMSGI s MINONE stejné jako neuvedení MOMSGI.

MOCORI
Načíst zprávu s uvedeným identifikátorem korelace.

Tato volba určuje, že zpráva, která má být načtena, musí mít identifikátor korelace, který odpovídá hodnotě pole *MDCID* v parametru **MSGDSC** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například identifikátor zprávy).

Není-li tato volba zadána, bude pole *MDCID* v parametru **MSGDSC** ignorováno a bude se shodovat libovolný identifikátor korelace.

Poznámka: Identifikátor korelace CINONE je speciální hodnota, která odpovídá libovolnému identifikátoru korelace v deskriptoru MQMD pro zprávu. Proto je zadání MOCORI s CINONE stejné jako neuvedení MOCORI.

MOGRPI
Načíst zprávu s uvedeným identifikátorem skupiny.

Tato volba určuje, že zpráva, která má být načtena, musí mít identifikátor skupiny, který odpovídá hodnotě pole *MDGID* v parametru **MSGDSC** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například identifikátor korelace).

Není-li tato volba uvedena, pole *MDGID* v parametru **MSGDSC** se ignoruje a libovolný identifikátor skupiny se shoduje.

Poznámka: Identifikátor skupiny GINONE je speciální hodnota, která odpovídá libovolnému identifikátoru skupiny v deskriptoru MQMD pro zprávu. Proto je uvedení MOGRPI s GINONE stejné jako neuvedení MOGRPI.

MOSEQN
Načíst zprávu s uvedeným pořadovým číslem zprávy.

Tato volba určuje, že zpráva, která má být načtena, musí mít pořadové číslo zprávy, které odpovídá hodnotě pole *MDSEQ* v parametru **MSGDSC** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například identifikátor skupiny).

Není-li tato volba uvedena, pole *MDSEQ* v parametru **MSGDSC** se ignoruje a libovolné pořadové číslo zprávy se shoduje.

MOOFFS
Načíst zprávu s uvedeným posunutím.

Tato volba určuje, že zpráva, která má být načtena, musí mít posunutí, které odpovídá hodnotě pole *MDOFF* v parametru **MSGDSC** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou použít (například pořadové číslo zprávy).

Není-li tato volba uvedena, pole *MDOFF* v parametru **MSGDSC** se ignoruje a všechny odchylky se shodují.

Pokud není uvedena žádná z popsaných voleb, lze použít následující volbu:

MONONE
Žádné shody.

Tato volba určuje, že při výběru zprávy, která má být vrácena, se nemají používat žádné shody; proto jsou všechny zprávy ve frontě vhodné pro načtení (ale podléhají řízení pomocí voleb GMAMSA, GMASGA a GMCMPM).

MONONE je definován jako pomocná programová dokumentace. Není zamýšleno, aby tato volba byla použita s jinou volbou MO*, ale protože její hodnota je nula, nelze takové použití zjistit.

Toto pole je vstupní pole. Počáteční hodnota tohoto pole je MOMSGI s MOCORI. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER2.

Poznámka: Počáteční hodnota pole *GMMO* je definována pro kompatibilitu se správci front dřívější verze. Při čtení řady zpráv z fronty bez použití kritérií výběru však tato počáteční hodnota vyžaduje, aby aplikace před každým voláním MQGET resetovala pole *MDMID* a *MDCID* na hodnoty MINONE a CINONE. Nutnosti resetovat *MDMID* a *MDCID* se lze vyhnout nastavením parametru *GMVER* na hodnotu GMVER2a parametru *GMMO* na hodnotu MONONE.

GMOPT (10místné celé číslo se znaménkem)

Volby, které řídí akci MQGET.

Lze uvést nula nebo více z následujících popsaných voleb. Je-li vyžadováno více hodnot, lze je přidat (nepřidávejte stejnou konstantu více než jednou). Kombinace voleb, které nejsou platné, jsou uvedeny; všechny ostatní kombinace jsou platné.

Volby čekání: Následující volby se týkají čekání na doručení zpráv do fronty:

GMWT
Počkejte na doručení zprávy.

Aplikace bude čekat na doručení vhodné zprávy. Maximální doba, po kterou aplikace čeká, je uvedena v souboru *GMWI*.

Jsou-li požadavky MQGET zablokovány nebo dojde-li k zablokování požadavků MQGET během čekání, čekání se zruší a volání se dokončí s kódem příčiny CCFAIL a kódem příčiny RC2016, bez ohledu na to, zda jsou ve frontě vhodné zprávy.

Tuto volbu lze použít s volbami GMBRWF nebo GMBRWN.

Pokud několik aplikací čeká ve stejné sdílené frontě, aplikace nebo aplikace, které jsou aktivovány při doručení vhodné zprávy, jsou popsány dále v této části.

Poznámka: V následujícím popisu se jedná o volání procházení MQGET, které určuje jednu z voleb procházení, nikoli však GMLK; volání MQGET určující volbu GMLK je považováno za volání bez procházení.

- Pokud čeká jedno nebo více volání MQGET bez procházení, ale žádná volání MQGET bez procházení nečeká, aktivuje se jedno.
- Pokud jeden nebo více volání MQGET pro procházení čeká, ale žádná volání MQGET pro neprocházení nečeká, všechny jsou aktivovány.
- Pokud čeká jedno nebo více volání MQGET bez procházení a jedno nebo více volání MQGET s procházením, aktivuje se jedno volání MQGET bez procházení a žádné volání MQGET s procházením nebo všechna volání MQGET s procházením. (Počet aktivovaných volání procházení MQGET nelze předpovědět, protože závisí na aspektech plánování operačního systému a dalších faktorech.)

Pokud ve stejné frontě čeká více než jedno volání MQGET bez procházení, aktivuje se pouze jedno. V této situaci se správce front pokusí udělit prioritu čekání na volání nonbrowse v následujícím pořadí:

1. Specifické požadavky get-wait, které mohou být uspokojeny pouze určitými zprávami, například těmi, které mají specifickou hodnotu *MDMID* nebo *MDCID* (nebo obojí).
2. Obecné požadavky get-wait, které mohou být uspokojeny libovolnou zprávou.

Je třeba uvést následující body:

- V rámci první kategorie není dána žádná další priorita specifickým požadavkům get-wait, například těm, které uvádějí jak *MDMID* , tak *MDCID*.
- V rámci jedné z těchto kategorií nelze předpovědět, která aplikace je vybrána. Zejména aplikace, která čeká nejdéle, nemusí být nutně vybrána.

- Délka cesty a aspekty plánování priorit operačního systému mohou znamenat, že čekající aplikace s nižší prioritou operačního systému, než se očekávalo, načte zprávu.
- Může se také stát, že aplikace, která nečeká, načte zprávu raději než takovou, která je.

GMWT se ignoruje, pokud je uvedeno s GMBRWC nebo GMMUC; neobjeví se žádná chyba.

GMNWT
Vrátit okamžitě, pokud není žádná vhodná zpráva.

Aplikace nemá čekat, pokud není k dispozici žádná vhodná zpráva. Jedná se o opak volby GMWT a je definována pro podporu programové dokumentace. Jedná se o předvolbu, není-li zadán ani jeden z nich.

GMFIQ
Selhání při uvedení správce front do klidového stavu.

Tato volba vynutí selhání volání MQGET, pokud je správce front ve stavu uvedení do klidového stavu.

Je-li tato volba zadána společně s GMWT a čekání probíhá v době, kdy správce front přejde do klidového stavu:

- Čekání je zrušeno a volání vrátí kód dokončení CCFAIL s kódem příčiny RC2161 .

Není-li parametr GMFIQ uveden a správce front přejde do klidového stavu, čekání se nezruší.

Volby Syncpoint: Následující volby se týkají účasti volání MQGET v rámci pracovní jednotky:

GMSYP
Získat zprávu s ovládacím prvkem synchronizačního bodu.

Požadavek je pracovat v rámci běžných protokolů jednotky práce. Zpráva je označena jako nedostupná pro jiné aplikace, ale je odstraněna z fronty pouze v případě, že je transakce potvrzena. Zpráva se znovu zpřístupní, pokud je jednotka práce vrácena zpět.

Není-li tato volba nebo GMNSYP uvedena, požadavek na získání není v rámci pracovní jednotky.

Tato volba není platná s žádnou z následujících voleb:

- GMBRWF
- GMBRWC
- GMBRWN
- GMLK
- GMNSYP
- GMPSYP
- GMUNLK

GMPSYP
Získat zprávu s ovládacím prvkem synchronizačního bodu, pokud je zpráva trvalá.

Požadavek je pracovat v rámci normálních protokolů jednotky práce, ale pouze v případě, že načtená zpráva je trvalá. Trvalá zpráva má v poli *MDPER* v deskriptoru MQMD hodnotu PEPER.

- Pokud je zpráva trvalá, správce front zpracuje volání, jako by aplikace určila GMSYP.
- Pokud zpráva není trvalá, správce front zpracuje volání, jako by aplikace zadala GMNSYP (podrobnosti naleznete v následující části).

Tato volba není platná s žádnou z následujících voleb:

- GMBRWF
- GMBRWC
- GMBRWN
- GMCMPM

- GMNSYP
- GMSYP
- GMUNLK

GMNSYP
Získat zprávu bez ovládacího prvku synchronizačního bodu.

Požadavek je pracovat mimo normální protokoly jednotky práce. Zpráva je z fronty odstraněna okamžitě (pokud se nejedná o požadavek na procházení). Zprávu nelze znovu zpřístupnit zálohováním pracovní jednotky.

Tato volba se předpokládá, pokud je uvedeno GMBRWF nebo GMBRWN.

Pokud tato volba a GMSYP nejsou uvedeny, požadavek na získání není v rámci pracovní jednotky.

Tato volba není platná s žádnou z následujících voleb:

- GMSYP
- GMPSYP

Volby procházení: Následující volby se týkají procházení zpráv ve frontě:

GMBRWF
Procházet od začátku fronty.

Když je fronta otevřena s volbou OOBRW, je vytvořen kurzor procházení, umístěný logicky před první zprávou ve frontě. Následná volání MQGET uvádějící volbu GMBRWF, GMBRWN nebo GMBRWC lze použít k nedestruktivnímu načtení zpráv z fronty. Kurzor procházení označuje pozici ve zprávách ve frontě, ze které další volání MQGET s GMBRWN hledá vhodnou zprávu.

Volání MQGET s GMBRWF způsobí, že předchozí pozice kurzoru procházení bude ignorována. Načte se první zpráva ve frontě, která splňuje podmínky uvedené v deskriptoru zprávy. Zpráva zůstává ve frontě a kurzor procházení je umístěn na této zprávě.

Po tomto volání je kurzor procházení umístěn na vrácené zprávě. Pokud je zpráva odebrána z fronty před vydáním dalšího volání MQGET s GMBRWN, kurzor procházení zůstane na pozici ve frontě, kde byla zpráva obsazena, i když je tato pozice nyní prázdná.

Volbu GMMUC pak lze v případě potřeby použít s voláním MQGET bez procházení, aby se zpráva odebrala z fronty.

Kurzor procházení není přesunut voláním MQGET bez procházení s použitím stejného popisovače *HOBJ* . Není přesunut ani voláním procházení MQGET, které vrací kód dokončení CCFAIL nebo kód příčiny RC2080 .

Spolu s touto volbou lze zadat volbu GMLK, která způsobí, že procházená zpráva bude uzamčena.

GMBRWF lze zadat s libovolnou platnou kombinací voleb GM* a MO*, které řídí zpracování zpráv ve skupinách a segmentech logických zpráv.

Je-li zadána hodnota GMLOGO, jsou zprávy procházeny v logickém pořadí. Je-li tato volba vynechána, jsou zprávy procházeny ve fyzickém pořadí. Je-li uvedeno GMBRWF, je možné přepínat mezi logickým a fyzickým pořadem, ale následná volání MQGET používající GMBRWN musí procházet frontu ve stejném pořadí jako poslední volání, které určilo GMBRWF pro popisovač fronty.

Informace o skupině a segmentu, které správce front uchovává pro volání MQGET, která procházejí zprávy ve frontě, jsou odděleny od informací o skupině a segmentu, které správce front uchovává pro volání MQGET, která odebírají zprávy z fronty. Je-li zadána volba GMBRWF, správce front ignoruje informace o skupině a segmentu pro procházení a skenuje frontu, jako by neexistovala žádná aktuální skupina a žádná aktuální logická zpráva. Pokud je volání MQGET úspěšné (kód dokončení CCOK nebo CCWARN), informace o skupině a segmentu pro procházení jsou nastaveny na informace o vrácené zprávě; pokud volání selže, informace o skupině a segmentu zůstanou stejné jako před voláním.

Tato volba není platná s žádnou z následujících voleb:

- GMBRWC
- GMBRWN
- GMMUC
- GMSYP
- GMPSYP
- GMUNLK

Také se jedná o chybu, pokud nebyla fronta otevřena pro procházení.

GMBRWN
Procházet z aktuální pozice ve frontě.

Kurzor procházení je rozšířen na další zprávu ve frontě, která splňuje kritéria výběru uvedená ve volání MQGET. Zpráva je vrácena aplikaci, ale zůstává ve frontě.

Po otevření fronty pro procházení má první volání procházení pomocí popisovače stejný efekt, ať už uvádí volbu GMBRWF nebo GMBRWN.

Pokud je zpráva odebrána z fronty před dalším voláním MQGET s GMBRWN, kurzor procházení logicky zůstane na pozici ve frontě, kterou zpráva zabírala, i když je tato pozice nyní prázdná.

Zprávy jsou uloženy ve frontě jedním ze dvou způsobů:

- FIFO v rámci priority (MSPRIO), nebo
- FIFO bez ohledu na prioritu (MSFIFO)

Atribut fronty **MsgDeliverySequence** označuje, která metoda se použije (podrobnosti viz “Atributy pro fronty” na stránce 1360).

Má-li fronta hodnotu *MsgDeliverySequence* MSPRIO a do fronty je doručena zpráva s vyšší prioritou, než je priorita, na kterou aktuálně ukazuje kurzor procházení, není tato zpráva během aktuálního procházení fronty pomocí GMBRWN nalezena. Lze jej nalézt pouze po resetování kurzoru procházení pomocí GMBRWF (nebo opětovným otevřením fronty).

V případě potřeby lze později použít volbu GMMUC s voláním MQGET bez procházení, aby se zpráva odebrala z fronty.

Kurzor procházení není přesunut pomocí volání nonbrowse MQGET s použitím stejného popisovače *HOBJ* .

Spolu s touto volbou lze zadat volbu GMLK, která způsobí, že procházená zpráva bude uzamčena.

GMBRWN může být uvedeno s jakoukoli platnou kombinací voleb GM* a MO*, které řídí zpracování zpráv ve skupinách a segmentech logických zpráv.

Je-li zadána hodnota GMLOGO, jsou zprávy procházeny v logickém pořadí. Je-li tato volba vynechána, jsou zprávy procházeny ve fyzickém pořadí. Je-li uvedeno GMBRWF, je možné přepínat mezi logickým a fyzickým pořadem, ale následná volání MQGET používající GMBRWN musí procházet frontu ve stejném pořadí jako poslední volání, které určilo GMBRWF pro popisovač fronty. Volání selže s kódem příčiny RC2259 , pokud není tato podmínka splněna.

Poznámka: Pokud je volání MQGET použito k procházení za koncem skupiny zpráv (nebo logickou zprávou, která není ve skupině), není-li zadán parametr GMLOGO, je třeba věnovat zvláštní pozornost. Pokud například poslední zpráva ve skupině předchází první zprávě ve skupině ve frontě pomocí GMBRWN pro procházení za koncem skupiny, uvedení MOSEQN s hodnotou *MDSEQ* nastavenou na 1 (pro nalezení první zprávy další skupiny) vrátí první zprávu ve skupině, která již procházela. K tomu může dojít okamžitě, nebo několik volání MQGET později (pokud existují skupiny, které zasahují).

Možnost nekonečné smyčky lze vyhnout otevřením fronty dvakrát pro procházení:

- Pomocí prvního popisovače můžete procházet pouze první zprávu v každé skupině.
- Pomocí druhého popisovače můžete procházet pouze zprávy v rámci specifické skupiny.

- Pomocí voleb MO* přesuňte druhý kurzor procházení na pozici prvního kurzoru procházení před procházením zpráv ve skupině.
- Nepoužívejte GMBRWN k procházení za koncem skupiny.

Informace o skupině a segmentu, které správce front uchovává pro volání MQGET, která procházejí zprávy ve frontě, jsou odděleny od informací o skupině a segmentu, které uchovává pro volání MQGET, která odebírají zprávy z fronty.

Tato volba není platná s žádnou z následujících voleb:

- GMBRWF
- GMBRWC
- GMMUC
- GMSYP
- GMPSYP
- GMUNLK

Také se jedná o chybu, pokud nebyla fronta otevřena pro procházení.

GMBRWC
Procházet zprávu pod kurzorem procházení.

Tato volba způsobí, že zpráva, na kterou ukazuje kurzor procházení, bude načtena nedestruktivně, bez ohledu na volby MO* uvedené v poli *GMMO* v MQGMO.

Zpráva, na kterou ukazuje kurzor procházení, je ta, která byla naposledy načtena pomocí volby GMBRWF nebo GMBRWN. Volání selže, pokud pro tuto frontu nebylo od jejího otevření zadáno žádné z těchto volání, nebo pokud zpráva, která byla pod kurzorem procházení, byla od té doby načtena destruktivně.

Pozice kurzoru procházení se tímto voláním nezmění.

Volbu GMMUC pak lze v případě potřeby použít s voláním MQGET bez procházení, aby se zpráva odebrala z fronty.

Kurzor procházení není přesunut voláním MQGET bez procházení s použitím stejného popisovače *HOBJ* . Nepřesunuje se ani pomocí volání procházení MQGET, které vrací kód dokončení CCFAIL nebo kód příčiny RC2080.

Pokud je GMBRWC uvedeno s GMLK:

- Pokud již existuje zamknutá zpráva, musí být zpráva pod kurzorem, aby byla vrácena bez odemknutí a opětovného zamknutí; zpráva zůstane uzamčena.
- Pokud neexistuje žádná zamčená zpráva, zpráva pod kurzorem procházení (pokud existuje) je uzamčena a vrácena do aplikace; pokud není žádná zpráva pod kurzorem procházení, volání selže.

Pokud je GMBRWC uvedeno bez GMLK:

- Pokud je již zpráva uzamčena, musí být ta, která je pod kurzorem. Tato zpráva se vrátí aplikaci a poté se odemkne. Vzhledem k tomu, že zpráva je nyní odemknuta, neexistuje žádná záruka, že ji lze znovu procházet nebo načíst destruktivně (může být načtena destruktivně jinou aplikací, která získává zprávy z fronty).
- Pokud neexistuje žádná zamknutá zpráva, je zpráva pod kurzorem procházení (pokud existuje) vrácena aplikaci; pokud pod kurzorem procházení není žádná zpráva, volání selže.

Je-li GMCMPM uveden s GMBRWC, kurzor procházení musí identifikovat zprávu s polem *MDOFF* v MQMD, které je nula. Není-li tato podmínka splněna, volání se nezdaří s kódem příčiny RC2246 .

Informace o skupině a segmentu, které správce front uchovává pro volání MQGET, která procházejí zprávy ve frontě, jsou odděleny od informací o skupině a segmentu, které uchovává pro volání MQGET, která odebírají zprávy z fronty.

Tato volba není platná s žádnou z následujících voleb:

- GMBRWF
- GMBRWN
- GMMUC
- GMSYP
- GMPSYP
- GMUNLK

Také se jedná o chybu, pokud nebyla fronta otevřena pro procházení.

GMMUC
Získat zprávu pod kurzorem procházení.

Tato volba způsobí načtení zprávy, na kterou ukazuje kurzor procházení, bez ohledu na volby MO* uvedené v poli *GMMO* v MQGMO. Zpráva je odebrána z fronty.

Zpráva, na kterou ukazuje kurzor procházení, je ta, která byla naposledy načtena pomocí volby GMBRWF nebo GMBRWN.

Je-li GMCMPM uveden s GMMUC, kurzor procházení musí identifikovat zprávu s polem *MDOFF* v MQMD, které je nula. Není-li tato podmínka splněna, volání se nezdaří s kódem příčiny RC2246 .

Tato volba není platná s žádnou z následujících voleb:

- GMBRWF
- GMBRWC
- GMBRWN
- GMUNLK

Také se jedná o chybu, pokud nebyla fronta otevřena pro procházení i pro vstup. Pokud kurzor procházení aktuálně neukazuje na zprávu s možností načtení, volání MQGET vrátí chybu.

Volby zámku: Následující volby se týkají zamykání zpráv ve frontě:

GMLK
Zamknout zprávu.

Tato volba uzamkne procházenou zprávu, aby se tato zpráva stala neviditelnou pro jakýkoli jiný manipulátor otevřený pro frontu. Volbu lze zadat pouze v případě, že je zadána také jedna z následujících voleb:

- GMBRWF
- GMBRWN
- GMBRWC

Pro každý popisovač fronty může být uzamčena pouze jedna zpráva, ale může se jednat o logickou zprávu nebo fyzickou zprávu:

- Je-li zadána volba GMCMPM, všechny segmenty zpráv, které tvoří logickou zprávu, jsou uzamčeny pro manipulátor fronty (jsou-li všechny ve frontě a jsou-li k dispozici pro načtení).
- Není-li parametr GMCMPM zadán, je pro manipulátor fronty uzamknuta pouze jedna fyzická zpráva. Pokud se tato zpráva stane segmentem logické zprávy, uzamčený segment zabrání jiným aplikacím, které používají GMCMPM k načtení nebo procházení logické zprávy.

Uzamčená zpráva je vždy zpráva pod kurzorem procházení a zprávu lze odebrat z fronty pozdějším voláním MQGET, které určuje volbu GMMUC. Jiná volání MQGET používající manipulátor fronty mohou také odebrat zprávu (například volání, které určuje identifikátor zamknuté zprávy).

Pokud volání vrátí kód dokončení CCFAIL nebo CCWARN s kódem příčiny RC2080, žádná zpráva se nezamkne.

Pokud se aplikace rozhodne zprávu z fronty neodebrat, zámek se uvolní takto:

- Je-li pro tento manipulátor zadáno jiné volání MQGET s parametrem GMBRWF nebo GMBRWN (s parametrem GMLK nebo bez něj), zpráva se odemkne, pokud je volání dokončeno s parametrem CCOK nebo CCWARN, ale zůstane uzamčena, pokud je volání dokončeno s parametrem CCFAIL. Platí však následující výjimky:
  - Zpráva není odemknuta, pokud je vrácena hodnota CCWARN s RC2080.
  - Zpráva je odemknuta, pokud je vrácena hodnota CCFAIL s RC2033.

  Je-li také uveden GMLK, vrácená zpráva je uzamčena. Není-li GMLK uveden, po volání není žádná zamknutá zpráva.

  Je-li uveden GMWT a není-li okamžitě k dispozici žádná zpráva, dojde k odemčení původní zprávy před začátkem čekání (za předpokladu, že volání je jinak bez chyby).
- Další volání MQGET pro tento popisovač s GMBRWC (bez GMLK); zpráva se odemkne, pokud se volání dokončí s CCOK nebo CCWARN, ale zůstane zamčená, pokud se volání dokončí s CCFAIL. Platí však následující výjimka:
  - Zpráva není odemknuta, pokud je vrácena hodnota CCWARN s RC2080.
- Probíhá další volání MQGET pro tento manipulátor s GMUNLK.
- Zadání volání MQCLOSE pro tento manipulátor (buď explicitně, nebo implicitně ukončením aplikace).

K zadání této volby není vyžadována žádná speciální volba otevření, jiná než OOBRW, která je potřebná k zadání doprovodné volby procházení.

Tato volba není platná s žádnou z následujících voleb:

- GMSYP
- GMPSYP
- GMUNLK

GMUNLK
Odemknout zprávu.

Zpráva, která má být odemknuta, musí být dříve uzamčena voláním MQGET s volbou GMLK. Pokud pro tento manipulátor není uzamčena žádná zpráva, volání se dokončí s CCWARN a RC2209 .

Parametry **MSGDSC**, **BUFLEN**, **BUFFER**a **DATLEN** nejsou kontrolovány nebo pozměněny, pokud je zadán parametr GMUNLK. V souboru *BUFFER*není vrácena žádná zpráva.

Pro zadání této volby není vyžadována žádná speciální volba otevření (ačkoli OOBRW je potřeba k vydání požadavku na zámek na prvním místě).

Tato volba není platná s žádnými volbami kromě následujících:

- GMNWT
- GMNSYP

Obě tyto volby se předpokládají bez ohledu na to, zda jsou zadány, či nikoli.

Volby dat zprávy: Následující volby se vztahují ke zpracování dat zprávy při čtení zprávy z fronty:

GMATM
Povolit oříznutí dat zprávy.

Pokud je vyrovnávací paměť zpráv příliš malá na to, aby zadržela celou zprávu, umožní tato volba volání MQGET zaplnit vyrovnávací paměť tak velkou část zprávy, jak může vyrovnávací paměť zadržet, vydat kód pro dokončení varování a dokončit zpracování. To znamená:

- Při procházení zpráv je kurzor procházení rozšířen na vrácenou zprávu.
- Při odebírání zpráv je vrácená zpráva odebrána z fronty.
- Pokud nedojde k žádné jiné chybě, vrátí se kód příčiny RC2079 .

Bez této volby je vyrovnávací paměť stále naplněna tak velkou část zprávy, jak je možné ji zadržet, je vydán varovný kód dokončení, ale zpracování není dokončeno. To znamená:

- Při procházení zpráv není kurzor procházení rozšířen.
- Při odebírání zpráv není zpráva odebrána z fronty.
- Pokud nedojde k žádné jiné chybě, vrátí se kód příčiny RC2080 .

GMCONV
Převést data zprávy.

Tato volba požaduje, aby byla data aplikace ve zprávě převedena tak, aby byla v souladu s hodnotami *MDCSI* a *MDENC* uvedenými v parametru **MSGDSC** volání MQGET před zkopírováním dat do parametru **BUFFER** .

Pole *MDFMT* zadané při vložení zprávy předpokládá proces převodu, aby identifikoval povahu dat ve zprávě. Převod dat zprávy provádí správce front pro vestavěné formáty a uživatelská procedura pro jiné formáty.

- Pokud je převod úspěšně proveden, pole *MDCSI* a *MDENC* zadaná v parametru **MSGDSC** se při návratu z volání MQGET nezmění.
- Pokud převod nelze úspěšně provést (ale volání MQGET jinak skončí bez chyby), data zprávy se vrátí nepřevedená a pole *MDCSI* a *MDENC* v souboru *MSGDSC* se nastaví na hodnoty pro nepřevedenou zprávu. V tomto případě je kód dokončení CCWARN.

V obou případech tedy tato pole popisují identifikátor znakové sady a kódování dat zprávy vrácených v parametru **BUFFER** .

Seznam názvů formátů, pro které správce front provádí převod, naleznete v poli *MDFMT* popsaném v tématu “MQMD (deskriptor zprávy) na IBM i” na stránce 1105 .

Volby skupin a segmentů: Následující volby se vztahují ke zpracování zpráv ve skupinách a segmentech logických zpráv. Tyto definice mohou být užitečné při pochopení voleb:

Fyzická zpráva
Jedná se o nejmenší jednotku informací, kterou lze umístit do fronty nebo z ní odebrat. Často odpovídá informacím zadaným nebo načteným v rámci jednoho volání MQPUT, MQPUT1nebo MQGET. Každá fyzická zpráva má svůj vlastní deskriptor zpráv (MQMD). Obecně se fyzické zprávy rozlišují podle různých hodnot pro identifikátor zprávy (pole*MDMID* v deskriptoru MQMD), ačkoli to správce front nevynucuje.

Logická zpráva
Toto je jedna jednotka informací o aplikaci. Při absenci systémových omezení by logická zpráva byla stejná jako fyzická zpráva. Ale tam, kde jsou logické zprávy velké, mohou systémová omezení způsobit, že je vhodné nebo nezbytné rozdělit logickou zprávu do dvou nebo více fyzických zpráv, nazývaných segmenty.

Logická zpráva, která byla segmentována, se skládá ze dvou nebo více fyzických zpráv, které mají stejný nenulový identifikátor skupiny (pole*MDGID* v deskriptoru MQMD) a stejné pořadové číslo zprávy (pole*MDSEQ* v deskriptoru MQMD). Segmenty jsou odlišeny odlišnými hodnotami pro posunutí segmentu (pole*MDOFF* v MQMD), které poskytuje posunutí dat ve fyzické zprávě od začátku dat v logické zprávě. Protože každý segment je fyzickou zprávou, segmenty v logické zprávě mají obvykle různé identifikátory zpráv.

Logická zpráva, která nebyla segmentována, ale pro kterou byla segmentace povolena odesílající aplikací, má také nenulový identifikátor skupiny, ačkoli v tomto případě existuje pouze jedna fyzická zpráva s tímto identifikátorem skupiny, pokud logická zpráva nepatří do skupiny zpráv. Logické zprávy, pro které byla segmentace blokována odesílající aplikací, mají identifikátor skupiny s hodnotou null (GINONE), pokud logická zpráva nepatří do skupiny zpráv.

Skupina zpráv
Jedná se o sadu jedné nebo více logických zpráv, které mají stejný nenulový identifikátor skupiny. Logické zprávy ve skupině jsou rozlišeny různými hodnotami pro pořadové číslo zprávy, což je celé

číslo v rozsahu 1 až n, kde n je počet logických zpráv ve skupině. Je-li jedna nebo více logických zpráv segmentováno, je ve skupině více než n fyzických zpráv.

GMLOGO
Zprávy ve skupinách a segmentech logických zpráv jsou vraceny v logickém pořadí.

Tato volba řídí pořadí, ve kterém jsou zprávy vraceny následnými voláními MQGET pro popisovač fronty. Volba musí být uvedena na každém z těchto volání, aby měla efekt.

Je-li pro následná volání MQGET pro manipulátor fronty zadána hodnota GMLOGO, jsou zprávy ve skupinách vraceny v pořadí určeném pořadovými čísly zpráv a segmenty logických zpráv jsou vraceny v pořadí určeném jejich posuny segmentů. Toto pořadí se může lišit od pořadí, ve kterém se tyto zprávy a segmenty vyskytují ve frontě.

Poznámka: Určení GMLOGO nemá žádné nepříznivé důsledky na zprávy, které nepatří do skupin a nejsou segmenty. Ve skutečnosti se s takovými zprávami zachází tak, jako by každá z nich patřila do skupiny zpráv, která se skládá pouze z jedné zprávy. Proto je naprosto bezpečné určit GMLOGO při načítání zpráv z front, které mohou obsahovat směs zpráv ve skupinách, segmentech zpráv a nesegmentovaných zprávách, které nejsou ve skupinách.

Chcete-li vrátit zprávy v požadovaném pořadí, správce front uchová informace o skupině a segmentu mezi následnými voláními MQGET. Tyto informace identifikují aktuální skupinu zpráv a aktuální logickou zprávu pro popisovač fronty, aktuální pozici ve skupině a logickou zprávu a zda jsou zprávy načítány v rámci pracovní jednotky. Vzhledem k tomu, že správce front tyto informace uchovává, nemusí aplikace před každým voláním MQGET nastavit informace o skupině a segmentu. Konkrétně to znamená, že aplikace nemusí nastavovat pole *MDGID*, *MDSEQ*a *MDOFF* v deskriptoru MQMD. Aplikace však musí při každém volání správně nastavit volbu GMSYP nebo GMNSYP.

Když je fronta otevřena, neexistuje žádná aktuální skupina zpráv a žádná aktuální logická zpráva. Skupina zpráv se stane aktuální skupinou zpráv, když volání MQGET vrátí zprávu s příznakem MFMIG. Při zadání GMLOGO pro následná volání zůstává tato skupina aktuální skupinou, dokud se nevrátí zpráva, která má:

- MFLMIG bez MFSEG (to znamená, že poslední logická zpráva ve skupině není segmentovaná), nebo
- MFLMIG s MFLSEG (to znamená, že vrácená zpráva je posledním segmentem poslední logické zprávy ve skupině).

Je-li vrácena taková zpráva, dojde k ukončení skupiny zpráv a po úspěšném dokončení tohoto volání MQGET již není aktuální skupina. Podobně se logická zpráva stane aktuální logickou zprávou, když volání MQGET vrátí zprávu, která má příznak MFSEG, a tato logická zpráva se ukončí, když je vrácena zpráva, která má příznak MFLSEG.

Nejsou-li zadána žádná kritéria výběru, vrátí následná volání MQGET zprávy pro první skupinu zpráv ve frontě (ve správném pořadí), poté zprávy pro druhou skupinu zpráv atd. až do doby, kdy nebudou k dispozici žádné další zprávy. Je možné vybrat konkrétní vrácené skupiny zpráv uvedením jedné nebo více následujících voleb v poli *GMMO* :

- MOMSGI
- MOCORI
- MOGRPI

Tyto volby jsou však platné pouze v případě, že neexistuje žádná aktuální skupina zpráv nebo logická zpráva. Viz pole *GMMO* popsané v tomto tématu.

Tabulka 702 na stránce 1085 zobrazuje hodnoty polí *MDMID*, *MDCID*, *MDGID*, *MDSEQ*a *MDOFF* , které správce front hledá při pokusu o nalezení zprávy, která má být vrácena při volání MQGET. To platí jak pro odebrání zpráv z fronty, tak pro procházení zpráv ve frontě. Sloupce v tabulce mají následující význam:

PROTOKOLNÍ ZÁZNAM
Označuje, zda je volba GMLOGO uvedena ve volání.

Aktuální grp
Označuje, zda před voláním existuje aktuální skupina zpráv.

Aktuální zpráva protokolu
Označuje, zda před voláním existuje aktuální logická zpráva.

Ostatní sloupce
Zobrazit hodnoty, které správce front hledá. "Předchozí" označuje hodnotu vrácenou pro pole v předchozí zprávě pro popisovač fronty.

Tabulka 702. Volby MQGET související se zprávami ve skupinách a segmentech logických zpráv

Volby, které zadáte	**Stav skupiny a zprávy protokolu před voláním**		**Hodnoty, které správce front hledá**				
LOG ORD	**Cur grp**	**Aktuální zpráva protokolu**	*MDMID*	*MDCID*	*MDGID*	*MDSEQ*	*MDOFF*
Ano	Ne	Ne	řízený subjektem *GMMO*	řízený subjektem *GMMO*	řízený subjektem *GMMO*	1	0
Ano	Ne	Ano	Libovolný identifikátor zprávy	Libovolný identifikátor korelace	Předchozí identifikátor skupiny	1	Předchozí offset + předchozí délka segmentu
Ano	Ano	Ne	Libovolný identifikátor zprávy	Libovolný identifikátor korelace	Předchozí identifikátor skupiny	Předchozí pořadové číslo + 1	0
Ano	Ano	Ano	Libovolný identifikátor zprávy	Libovolný identifikátor korelace	Předchozí identifikátor skupiny	Předchozí pořadové číslo	Předchozí offset + předchozí délka segmentu
Ne	buď	buď	řízený subjektem *GMMO*	řízený subjektem *GMMO*	řízený subjektem *GMMO*	řízený subjektem *GMMO*	řízený subjektem *GMMO*

Je-li ve frontě přítomno více skupin zpráv, které jsou vhodné pro návrat, jsou skupiny vráceny v pořadí určeném pozicí ve frontě prvního segmentu první logické zprávy v každé skupině (tj. fyzické zprávy, které mají pořadová čísla zpráv 1, a offsety 0, určují pořadí, ve kterém jsou vhodné skupiny vráceny).

Volba GMLOGO ovlivňuje jednotky práce takto:

- Pokud je první logická zpráva nebo segment ve skupině načten v rámci pracovní jednotky, všechny ostatní logické zprávy a segmenty ve skupině musí být načteny v rámci pracovní jednotky, pokud je použit stejný manipulátor fronty. Nemusí však být načteny v rámci stejné pracovní jednotky. To umožňuje rozdělit skupinu zpráv skládající se z mnoha fyzických zpráv na dvě nebo více po sobě jdoucích jednotek práce pro popisovač fronty.
- Není-li první logická zpráva nebo segment ve skupině načten v rámci pracovní jednotky, nelze v rámci pracovní jednotky načíst žádnou jinou logickou zprávu a segmenty ve skupině, pokud je použit stejný manipulátor fronty.

Nejsou-li tyto podmínky splněny, volání MQGET se nezdaří s kódem příčiny RC2245 .

Je-li zadána volba GMLOGO, nesmí být hodnota MQGMO zadaná ve volání MQGET menší než hodnota GMVER2a hodnota MQMD nesmí být menší než hodnota MDVER2. Není-li tato podmínka splněna, volání selže s kódem příčiny RC2256 nebo RC2257 , podle potřeby.

Není-li pro následná volání MQGET pro manipulátor fronty zadána hodnota GMLOGO, budou zprávy vráceny bez ohledu na to, zda patří do skupin zpráv nebo zda se jedná o segmenty logických zpráv. To znamená, že zprávy nebo segmenty z určité skupiny nebo logické zprávy mohou být vráceny mimo pořadí, nebo mohou být promíchány se zprávami nebo segmenty z jiných skupin nebo logických zpráv, nebo se zprávami, které nejsou ve skupinách a nejsou segmenty. V této situaci jsou konkrétní zprávy, které jsou vráceny následnými voláními MQGET, řízeny volbami MO* uvedenými v těchto voláních (podrobnosti o těchto volbách viz pole *GMMO* popsané v části "MQGMO (Volby Get-message) na systému IBM i" na stránce 1073).

Jedná se o techniku, kterou lze použít k restartování skupiny zpráv nebo logické zprávy uprostřed po selhání systému. Když se systém restartuje, aplikace může nastavit pole *MDGID*, *MDSEQ*, *MDOFF*a *GMMO* na odpovídající hodnoty a pak spustit volání MQGET s GMSYP nebo GMNSYP nastavenou podle potřeby, ale bez uvedení GMLOGO. Pokud je toto volání úspěšné, správce front zachová informace o skupině a segmentu a následná volání MQGET používající tento manipulátor fronty mohou určit GMLOGO jako normální.

Informace o skupině a segmentu, které správce front uchovává pro volání MQGET, jsou odděleny od informací o skupině a segmentu, které uchovává pro volání MQPUT. Kromě toho správce front uchovává samostatné informace pro:

- Volání MQGET, která odebírají zprávy z fronty.
- Volání MQGET, která procházejí zprávy ve frontě.

Pro jakýkoli daný popisovač fronty může aplikace směšovat volání MQGET, která určují GMLOGO, s voláními MQGET, která nikoli, ale je třeba poznamenat následující body:

- Není-li zadán parametr GMLOGO, každé úspěšné volání MQGET způsobí, že správce front nastaví uložené informace o skupině a segmentu na hodnoty odpovídající vrácené zprávě. Tato volba nahradí existující informace o skupině a segmentu uchovávané správcem front pro daný manipulátor fronty. Změní se pouze informace odpovídající akci volání (procházet nebo odebrat).
- Není-li uveden GMLOGO, volání neselže, pokud existuje aktuální skupina zpráv nebo logická zpráva; volání však může být úspěšné s kódem dokončení CCWARN. Tabulka 703 na stránce 1086 zobrazuje různé případy, které mohou nastat. V těchto případech, pokud kód dokončení není CCOK, je kód příčiny jeden z následujících:
  - RC2241
  - RC2242
  - RC2245

  Poznámka: Správce front nekontroluje informace o skupině a segmentu při procházení fronty nebo při zavírání fronty, která byla otevřena pro procházení, ale nikoli pro vstup; v těchto případech je kód dokončení vždy CCOK (nepředpokládá se žádné jiné chyby).

Tabulka 703. Výsledek, když volání MQGET nebo MQCLOSE není konzistentní s informacemi o skupině a segmentu

Aktuální volání je	Předchozí volání bylo MQGET s GMLOGO	Předchozí volání bylo MQGET bez GMLOGO
MQGET s GMLOGO	CCFAIL	CCFAIL
MQGET bez GMLOGO	CCWARN (varování)	CCOK
MQCLOSE s neukončenou skupinou nebo logickou zprávou	CCWARN (varování)	CCOK

Aplikacím, které jednoduše chtějí načíst zprávy a segmenty v logickém pořadí, se doporučuje zadat GMLOGO, protože se jedná o nejjednodušší volbu, kterou lze použít. Tato volba zbavuje aplikaci potřeby spravovat informace o skupině a segmentu, protože správce front tyto informace spravuje. Specializované aplikace však mohou vyžadovat větší kontrolu, než jakou poskytuje volba GMLOGO, a toho lze dosáhnout neurčením této volby. Pokud se tak stane, aplikace se musí ujistit, že pole *MDMID*, *MDCID*, *MDGID*, *MDSEQ*a *MDOFF* v MQMD a volby MO* v *GMMO* v MQGMO jsou správně nastaveny před každým voláním MQGET.

Například aplikace, která chce předávat přijaté fyzické zprávy bez ohledu na to, zda jsou tyto zprávy ve skupinách nebo segmentech logických zpráv, by neměla určovat GMLOGO. Důvodem je skutečnost, že v komplexní síti s více cestami mezi odesílajícími a přijímajícími správci front mohou fyzické zprávy přicházet mimo pořadí. Nezadáte-li GMLOGO a odpovídající PMLOGO ve volání MQPUT, může aplikace předání načíst a předat každou fyzickou zprávu ihned po jejím doručení, aniž by musela čekat na další zprávu v logickém pořadí.

GMLOGO může být specifikováno s jakoukoli jinou volbou GM* a s různými volbami MO* za vhodných okolností.

GMCMPM

Lze načíst pouze úplné logické zprávy.

Tato volba určuje, že volání MQGET může vrátit pouze úplnou logickou zprávu. Je-li logická zpráva segmentována, správce front znovu sestaví segmenty a vrátí aplikaci úplnou logickou zprávu. Skutečnost, že logická zpráva byla segmentována, není pro aplikaci, která ji načítá, zřejmá.

Poznámka: Jedná se o jedinou volbu, která způsobí, že správce front znovu sestaví segmenty zpráv. Není-li uveden, segmenty jsou vráceny aplikaci jednotlivě, pokud jsou přítomny ve frontě (a splňují ostatní kritéria výběru uvedená ve volání MQGET). Aplikace, které nechtějí přijímat jednotlivé segmenty, by proto měly vždy specifikovat GMCMPM.

Chcete-li použít tuto volbu, musí aplikace poskytnout vyrovnávací paměť, která je dostatečně velká, aby pojmula celou zprávu, nebo uvést volbu GMATM.

Pokud fronta obsahuje segmentované zprávy s chybějícími segmenty (například proto, že byly zpožděny v síti a dosud nebyly přijaty), uvedení GMCMPM zabrání načtení segmentů, které patří k neúplným logickým zprávám. Tyto segmenty zpráv však stále přispívají k hodnotě atributu fronty **CurrentQDepth** ; to znamená, že nemusí existovat žádné načítatelné logické zprávy, i když je hodnota *CurrentQDepth* větší než nula.

V případě trvalých zpráv může správce front znovu sestavit segmenty pouze v rámci pracovní jednotky:

- Pokud volání MQGET pracuje v rámci uživatelem definované jednotky práce, použije se tato jednotka práce. Pokud se volání nezdaří v rámci procesu opětovného sestavení, správce front obnoví ve frontě všechny segmenty, které byly odebrány během opětovného sestavení. Selhání však nebrání úspěšnému potvrzení transakce.
- Pokud volání pracuje mimo uživatelem definovanou jednotku práce a neexistuje žádná uživatelem definovaná jednotka práce, správce front vytvoří jednotku práce pouze po dobu trvání volání. Je-li volání úspěšné, správce front potvrdí transakci automaticky (aplikace to nemusí provést). Pokud volání selže, správce front odvolá transakci.
- Pokud volání pracuje mimo uživatelem definovanou jednotku práce, ale existuje uživatelem definovaná jednotka práce, nemůže správce front provést opětovné sestavení. Pokud zpráva nevyžaduje opětovné sestavení, volání může být stále úspěšné. Pokud však zpráva vyžaduje opětovné sestavení, volání selže s kódem příčiny RC2255 .

V případě přechodných zpráv správce front nevyžaduje, aby byla k provedení opětovného sestavení k dispozici jednotka práce.

Každá fyzická zpráva, která je segmentem, má svůj vlastní deskriptor zprávy. Pro segmenty tvořící jedinou logickou zprávu je většina polí v deskriptoru zprávy stejná pro všechny segmenty v logické zprávě-obvykle se liší pouze pole *MDMID*, *MDOFF*a *MDMFL* v jednotlivých segmentech logické zprávy. Pokud je však segment umístěn do fronty nedoručených zpráv ve zprostředkujícím

správci front, obslužná rutina DLQ načte zprávu uvádějící volbu GMCONV, což může vést ke změně znakové sady nebo kódování segmentu. Pokud obslužná rutina DLQ úspěšně odešle segment po cestě, může mít segment znakovou sadu nebo kódování, které se liší od ostatních segmentů v logické zprávě, když segment nakonec dorazí do cílového správce front.

Logická zpráva skládající se ze segmentů, ve kterých se pole *MDCSI*, *MDENC*nebo obě liší, nemůže být znovu sestavena správcem front do jediné logické zprávy. Místo toho správce front znovu sestaví a vrátí prvních několik po sobě jdoucích segmentů na začátku logické zprávy, které mají stejné identifikátory znakové sady a kódování, a volání MQGET se dokončí s kódem dokončení CCWARN a kódem příčiny RC2243 nebo RC2244 podle potřeby. K tomu dochází bez ohledu na to, zda je zadán parametr GMCONV. Chcete-li načíst zbývající segmenty, musí aplikace znovu zadat volání MQGET bez volby GMCMPM a načíst segmenty jeden po druhém. GMLOGO lze použít k načtení zbývajících segmentů v pořadí.

Je také možné, aby aplikace, která vkládá segmenty, nastavila jiná pole v deskriptoru zprávy na hodnoty, které se v jednotlivých segmentech liší. To však nemá žádnou výhodu, pokud přijímající aplikace používá GMCMPM k načtení logické zprávy. Když správce front znovu sestaví logickou zprávu, vrátí v deskriptoru zprávy hodnoty z deskriptoru zprávy pro první segment; jedinou výjimkou je pole *MDMFL* , které správce front nastaví tak, aby označilo, že znovu sestavená zpráva je jediným segmentem.

Je-li pro zprávu sestavy zadána volba GMCMPM, provede správce front speciální zpracování. Správce front zkontroluje frontu, aby zjistil, zda se ve frontě nacházejí všechny zprávy sestavy daného typu týkající se různých segmentů v logické zprávě. Pokud ano, lze je načíst jako jednu zprávu zadáním GMCMPM. Aby to bylo možné, zprávy sestavy musí být generovány správcem front nebo agentem MCA, který podporuje segmentaci, nebo musí původní aplikace požadovat alespoň 100 bajtů dat zprávy (tj. musí být zadány příslušné volby RO* D nebo RO* F). Pokud je pro segment přítomno méně dat aplikace, chybějící bajty jsou nahrazeny hodnotami null ve vrácené zprávě sestavy.

Je-li GMCMPM uveden s GMMUC nebo GMBRWC, kurzor procházení musí být umístěn na zprávě s polem *MDOFF* v MQMD, které má hodnotu 0. Není-li tato podmínka splněna, volání se nezdaří s kódem příčiny RC2246 .

GMCMPM znamená GMASGA, který proto nemusí být specifikován.

GMCMPM může být uveden s jakoukoli jinou možností GM* kromě GMPSYP a s jakoukoli z možností MO* kromě MOOFFS.

GMAMSA
Všechny zprávy ve skupině musí být k dispozici.

Tato volba určuje, že zprávy ve skupině budou k dispozici pro načtení pouze v případě, že jsou k dispozici všechny zprávy ve skupině. Pokud fronta obsahuje skupiny zpráv s některými chybějícími zprávami (například proto, že byly zpožděny v síti a dosud nebyly doručeny), uvedení GMAMSA zabrání načtení zpráv, které patří do neúplných skupin. Tyto zprávy však stále přispívají k hodnotě atributu fronty **CurrentQDepth** ; to znamená, že nemusí existovat žádné načítatelné skupiny zpráv, i když **CurrentQDepth** je větší než nula. Pokud nejsou k dispozici žádné další zprávy, které by bylo možné načíst, kód příčiny RC2033 se vrátí po uplynutí určeného intervalu čekání (pokud existuje).

Zpracování GMAMSA závisí na tom, zda je také specifikováno GMLOGO:

- Jsou-li zadány obě volby, ovlivní GMAMSA pouze v případě, že neexistuje žádná aktuální skupina nebo logická zpráva. Pokud existuje aktuální skupina nebo logická zpráva, GMAMSA se ignoruje. To znamená, že GMAMSA může zůstat zapnutý při zpracování zpráv v logickém pořadí.
- Pokud je GMAMSA uvedeno bez GMLOGO, GMAMSA má vždy efekt. To znamená, že volba musí být vypnuta po odebrání první zprávy ve skupině z fronty, aby bylo možné odebrat zbývající zprávy ve skupině.

Úspěšné dokončení volání MQGET s uvedením GMAMSA znamená, že v době, kdy bylo volání MQGET vydáno, byly všechny zprávy ve skupině ve frontě. Mějte však na paměti, že ostatní

aplikace jsou stále schopny odebrat zprávy ze skupiny (skupina není uzamčena pro aplikaci, která načte první zprávu ve skupině).

Není-li tato volba uvedena, zprávy patřící do skupin lze načíst, i když je skupina neúplná.

GMAMSA znamená GMASGA, který proto nemusí být specifikován.

GMAMSA lze zadat s jakoukoli jinou volbou GM* a s jakoukoli z voleb MO*.

GMASGA
Všechny segmenty v logické zprávě musí být k dispozici.

Tato volba určuje, že segmenty v logické zprávě budou k dispozici pro načtení pouze v případě, že jsou k dispozici všechny segmenty v logické zprávě. Pokud fronta obsahuje segmentované zprávy s chybějícími segmenty (například proto, že byly zpožděny v síti a dosud nebyly přijaty), uvedení GMASGA zabrání načtení segmentů, které patří k neúplným logickým zprávám. Tyto segmenty však stále přispívají k hodnotě atributu fronty **CurrentQDepth** ; to znamená, že nemusí existovat žádné načítatelné logické zprávy, i když je hodnota **CurrentQDepth** větší než nula. Pokud nejsou k dispozici žádné další zprávy, které by bylo možné načíst, kód příčiny RC2033 se vrátí po uplynutí určeného intervalu čekání (pokud existuje).

Zpracování GMASGA závisí na tom, zda je také specifikováno GMLOGO:

- Jsou-li zadány obě volby, GMASGA má efekt pouze v případě, že neexistuje žádná aktuální logická zpráva. Pokud existuje aktuální logická zpráva, GMASGA se ignoruje. To znamená, že GMASGA může zůstat zapnutý při zpracování zpráv v logickém pořadí.
- Pokud je GMASGA specifikován bez GMLOGO, GMASGA má vždy efekt. To znamená, že volba musí být vypnuta po odebrání prvního segmentu v logické zprávě z fronty, aby bylo možné odebrat zbývající segmenty v logické zprávě.

Není-li tato volba uvedena, segmenty zpráv lze načíst i v případě, že je logická zpráva neúplná.

Zatímco GMCMPM i GMASGA vyžadují, aby byly všechny segmenty k dispozici dříve, než je možné některý z nich načíst, první z nich vrátí úplnou zprávu, zatímco druhá umožňuje, aby byly segmenty načteny jeden po druhém.

Je-li pro zprávu sestavy zadána volba GMASGA, provede správce front speciální zpracování. Správce front kontroluje frontu a zjišťuje, zda pro každý ze segmentů, které tvoří úplnou logickou zprávu, existuje alespoň jedna zpráva sestavy. Pokud existuje, podmínka GMASGA je splněna. Správce front však nekontroluje typ přítomných zpráv sestavy, a proto může ve zprávách sestavy existovat směs typů sestav souvisejících se segmenty logické zprávy. V důsledku toho úspěch GMASGA neznamená, že GMCMPM uspěje. Je-li pro segmenty konkrétní logické zprávy přítomna směs typů sestav, musí být tyto zprávy sestavy načteny jeden po druhém.

GMASGA lze zadat s jakoukoli jinou volbou GM* a s jakoukoli z voleb MO*.

Výchozí volba: Pokud není vyžadována žádná z popsaných voleb, lze použít následující volbu:

GMNONE
Nejsou uvedeny žádné volby.

Tuto hodnotu lze použít k označení, že nebyly zadány žádné další volby; všechny volby předpokládají své výchozí hodnoty. GMNONE je definován jako pomocná dokumentace programu; není zamýšleno, aby tato volba byla použita s jinou, ale protože její hodnota je nula, nelze takové použití zjistit.

Počáteční hodnota pole *GMOPT* je GMNWT.

GMRE1 (řetězec 1 bajtových znaků)

Vyhrazeno.

Toto je vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER2.

GMRL (10místné celé číslo se znaménkem)

Délka vrácených dat zprávy (bajty).

Jedná se o výstupní pole nastavené správcem front na délku (v bajtech) dat zprávy vrácených voláním MQGET v parametru **BUFFER** . Pokud správce front tuto schopnost nepodporuje, je parametr *GMRL* nastaven na hodnotu RLUNDF.

Když jsou zprávy převáděny mezi kódováním nebo znakovými sadami, data zprávy mohou někdy měnit velikost. Při návratu z volání MQGET:

- Pokud *GMRL* není RLUNDF, je počet bajtů vrácených dat zprávy dán *GMRL*.
- Má-li parametr *GMRL* hodnotu RLUNDF, je počet bajtů vrácených dat zprávy obvykle dán hodnotou menší z hodnot *BUFLEN* a *DATLEN*, ale může být menší než tato hodnota, pokud je volání MQGET dokončeno s kódem příčiny RC2079 . Pokud k tomu dojde, nevýznamné bajty v parametru **BUFFER** jsou nastaveny na hodnoty null.

Je definována následující speciální hodnota:

RLUNDF
Délka vrácených dat není definována.

Počáteční hodnota tohoto pole je RLUNDF. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER3.

GMRQN (řetězec 48bajtových znaků)

Vyřešený název cílové fronty.

Jedná se o výstupní pole nastavené správcem front na lokální název fronty, ze které byla zpráva načtena, jak je definováno pro lokálního správce front. Liší se od názvu použitého k otevření fronty, pokud:

- Byla otevřena alias fronta (v takovém případě je vrácen název lokální fronty, do které byl alias převeden), nebo
- Byla otevřena modelová fronta (v takovém případě je vrácen název dynamické lokální fronty).

Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

GMRS2 (1bajtový znakový řetězec)

Vyhrazeno.

Toto je vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER4.

GMSEG (1bajtový znakový řetězec)

Příznak označující, zda je pro načtenou zprávu povolena další segmentace.

Má jednu z následujících hodnot:

SEGIHB
Segmentace není povolena.

SEGALW-pracovní
Segmentace je povolena.

Toto je výstupní pole. Počáteční hodnota tohoto pole je SEGIHB. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER2.

GMSG1 (10místné celé číslo se znaménkem)

Signál.

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

GMSG2 (10místné celé číslo se znaménkem)

Identifikátor signálu.

Toto je vyhrazené pole; jeho hodnota není významná.

GMSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

GMSIDV
Identifikátor pro strukturu voleb získání zprávy.

Toto pole je vždy vstupní pole. Počáteční hodnota tohoto pole je GMSIDV.

GMSST (1bajtový znakový řetězec)

Příznak označující, zda je načtená zpráva segmentem logické zprávy.

Má jednu z následujících hodnot:

SSNSEG
Zpráva není segmentem.

SSSEG
Zpráva je segmentem, ale není posledním segmentem logické zprávy.

SSLSEG
Zpráva je posledním segmentem logické zprávy.

Toto je také vrácená hodnota, pokud se logická zpráva skládá pouze z jednoho segmentu.

Toto pole je výstupní pole. Počáteční hodnota tohoto pole je SSNSEG. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER2.

GMTOK (16bajtový bitový řetězec)

Token zprávy.

Toto je vyhrazené pole; jeho hodnota není významná. Je definována následující speciální hodnota:

MTKNON
Žádný token zprávy.

Hodnota je binární nula pro délku pole.

Délka tohoto pole je dána hodnotou LNMTOK. Počáteční hodnota tohoto pole je MTKNON. Toto pole je ignorováno, pokud je hodnota *GMVER* menší než GMVER3.

GMVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

GMVER1
Version-1 struktura voleb get-message.

GMVER2
Version-2 struktura voleb get-message.

GMVER3
Version-3 struktura voleb get-message.

GMVER4
Version-4 struktura voleb get-message.

Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

GMVERC
Aktuální verze struktury voleb get-message.

Toto pole je vždy vstupní pole. Počáteční hodnota tohoto pole je GMVER1.

GMVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

GMVER1
Version-1 struktura voleb get-message.

GMVER2
Version-2 struktura voleb get-message.

GMVER3
Version-3 struktura voleb get-message.

GMVER4
Version-4 struktura voleb get-message.

Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

GMVERC
Aktuální verze struktury voleb get-message.

Toto pole je vždy vstupní pole. Počáteční hodnota tohoto pole je GMVER1.

GMWI (10místné celé číslo se znaménkem)

Interval čekání.

Jedná se o přibližnou dobu vyjádřenou v milisekundách, po kterou volání MQGET čeká na doručení vhodné zprávy (tj. zpráva splňující kritéria výběru uvedená v parametru **MSGDSC** volání MQGET; další podrobnosti viz pole *MDMID* popsané v tématu “MQMD (deskriptor zprávy) na IBM i” na stránce 1105). Pokud po uplynutí této doby nedorazila žádná vhodná zpráva, volání se dokončí s CCFAIL a kódem příčiny RC2033.

GMWI se používá s volbou GMWT. Není-li tato volba zadána, bude ignorována. Je-li zadána hodnota *GMWI* , musí být větší nebo rovna nule nebo musí být zadána následující speciální hodnota:

WIULIM
Neomezený interval čekání.

Počáteční hodnota tohoto pole je 0.

Počáteční hodnoty

Tabulka 704. Pole v MQGMO

Název pole	Název konstanty	Hodnota konstanty
GMSID	GMSIDV	'GMO¬'
GMVER	GMVER1	1
GMOPT	GMNWT	0
GMWI	Není	0
GMSG1	Není	0
GMSG2	Není	0
GMRQN	Není	Mezery
GMMO	MOMSGI + MOCORI	3
GMGST	GSNIG	''
GMSST	SSNSEG	''

Tabulka 704. Pole v MQGMO (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
GMSEG	SEGIHB	''
GMRE1	Není	''
GMTOK	MTKNON	Hodnoty null
GMRL	RLUNDF	-1
GMRS2	Není	''
GMMH	HMNONE	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQGMO Structure
D*
D* Structure identifier
D  GMSID                  1      4    INZ('GMO ')
D* Structure version number
D  GMVER                  5      8I 0 INZ(1)
D* Options that control the action ofMQGET
D  GMOPT                  9     12I 0 INZ(0)
D* Wait interval
D  GMWI                  13     16I 0 INZ(0)
D* Signal
D  GMSG1                 17     20I 0 INZ(0)
D* Signal identifier
D  GMSG2                 21     24I 0 INZ(0)
D* Resolved name of destination queue
D  GMRQN                 25     72    INZ
D* Options controlling selection criteriaused for MQGET
D  GMMO                  73     76I 0 INZ(3)
D* Flag indicating whether messageretrieved is in a group
D  GMGST                 77     77    INZ(' ')
D* Flag indicating whether messageretrieved is a segment of a
D* logicalmessage
D  GMSST                 78     78    INZ(' ')
D* Flag indicating whether furthersegmentation is allowed for themessage
D* retrieved
D  GMSEG                 79     79    INZ(' ')
D* Reserved
D  GMRE1                 80     80    INZ
D* Message token
D  GMTOK                 81     96    INZ(X'00000000000000-
D                                     0000000000000000')
D* Length of message data returned(bytes)
D  GMRL                  97    100I 0 INZ(-1)
D* Reserved
D  GMRS2                101    104I 0 INZ(0)
D* Message handle
D  GMMH                 105    112I 0 INZ(0)
```

IBM i MQIIH (IMS záhlaví informací) na IBM i

Struktura MQIIH popisuje informace, které musí být přítomny na začátku zprávy odeslané do mostu IMS přes IBM MQ for z/OS.

Přehled

Název formátu: FMIMS.

Znaková sada a kódování: Pro znakovou sadu a kódování používané pro strukturu MQIIH a data zpráv aplikace platí zvláštní podmínky:

- Aplikace, které se připojují ke správci front vlastnícímu frontu mostu IMS , musí poskytovat strukturu MQIIH, která je ve znakové sadě a kódování správce front. Důvodem je skutečnost, že v tomto případě není proveden převod dat struktury MQIIH.
- Aplikace, které se připojují k jiným správcům front, mohou poskytovat strukturu MQIIH, která se nachází v libovolné z podporovaných znakových sad a kódování. Převod MQIIH provádí přijímající agent kanálu zpráv připojený ke správci front, který vlastní frontu mostu IMS .

  Poznámka: Existuje jedna výjimka. Pokud správce front, který vlastní frontu mostu IMS , používá CICS pro distribuované fronty, musí být MQIIH ve znakové sadě a kódování správce front, který je vlastníkem fronty mostu IMS .
- Data zprávy aplikace následující po struktuře MQIIH musí být ve stejné znakové sadě a kódování jako struktura MQIIH. Pole *IICSI* a *IIENC* ve struktuře MQIIH nelze použít k určení znakové sady a kódování dat zprávy aplikace.

  Uživatelská procedura převodu dat musí být poskytnuta uživatelem pro převod dat zprávy aplikace, pokud data nejsou jedním z vestavěných formátů podporovaných správcem front.

- "Ověření přístupových štítků pro aplikace mostu IMS" na stránce 1094
- "Pole" na stránce 1094
- "Počáteční hodnoty" na stránce 1097
- "Deklarace RPG" na stránce 1098

Ověření přístupových štítků pro aplikace mostu IMS

Nyní je možné, aby administrátoři produktu IBM MQ zadali název aplikace, který se má použít pro ověření přístupových štítků, pro aplikace mostu IMS . Chcete-li to provést, název aplikace je uveden jako nový atribut PTKTAPPL pro definici objektu STGCLASS jako alfanumerický řetězec o délce 1 až 8 znaků.

Prázdná hodnota znamená, že ověření se provádí stejně jako v předchozích vydáních produktu IBM MQ, to znamená, že v požadavku na ověření neproudí žádný název aplikace a místo toho se použije hodnota MVSxxxx to.

Hodnota 1-8 alfanumerických znaků musí dodržovat pravidla pro názvy aplikací přístupového lístku, jak je popsáno v publikacích RACF .

IBM MQ Administrátoři a RACF se musí dohodnout na platných názvech aplikací, které se mají použít. Administrátor produktu RACF musí vytvořit profil ve třídě PTKTDATA, který poskytne přístup pro čtení ID uživatelů všech aplikací, kterým má být udělen přístup. Administrátor produktu IBM MQ musí vytvořit nebo změnit požadované definice STGCLASS, které určují název aplikace, který se má použít pro ověření pomocí přístupového lístku.

Související informace naleznete v příručce *Script (MQSC) Command Reference*.

Pole

Struktura MQIIH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

IIAUT (8bajtový znakový řetězec)

RACF heslo nebo přístupový prvek.

Tato volba je nepovinná. Je-li určena, používá se spolu s ID uživatele v kontextu zabezpečení MQMD k sestavení tokenu UTOKEN, který je odeslán do produktu IMS za účelem poskytnutí kontextu zabezpečení. Není-li uvedeno, použije se ID uživatele bez ověření. To závisí na nastavení přepínačů RACF , které mohou vyžadovat přítomnost ověřovatele.

Toto se ignoruje, pokud je první bajt prázdný nebo má hodnotu null. Lze použít následující speciální hodnotu:

IAUNON
Bez ověření.

Délka tohoto pole je dána hodnotou LNAUTH. Počáteční hodnota tohoto pole je IAUNON.

IICMT (1bajtový znakový řetězec)

Režim potvrzení.

Další informace o režimech potvrzení IMS viz *OTMA Reference* . Hodnota musí být jedna z následujících:

ICMCTS
Potvrdit a odeslat.

Tento režim znamená dvojité řazení výstupu do fronty, ale kratší dobu obsazenosti oblasti. Rychlé cesty a konverzační transakce nemohou být spuštěny s tímto režimem.

ICMSTC
Odeslat a poté potvrdit.

Počáteční hodnota tohoto pole je ICMCTS.

IICSI (10místné celé číslo se znaménkem)

Vyhrazeno.

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

IIENC (10místné celé číslo se znaménkem)

Vyhrazeno.

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

IIFLG (10místné celé číslo se znaménkem)

Příznaky.

Hodnota musí být:

IINONE
Žádné příznaky.

Počáteční hodnota tohoto pole je IINONE.

IIFMT (8bajtový znakový řetězec)

IBM MQ název formátu dat, která následují za MQIIH.

Tato volba určuje název formátu IBM MQ dat, která následují za strukturou MQIIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *MDFMT* v MQMD.

Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

IILEN (10místné celé číslo se znaménkem)

Délka struktury MQIIH.

Hodnota musí být:

IILEN1
Délka struktury záhlaví informací IMS .

Počáteční hodnota tohoto pole je IILEN1.

IILTO (8bajtový znakový řetězec)

Přepis logického terminálu.

Toto je umístěno v poli IO PCB. Je volitelný; pokud není uveden, použije se název TPIPE. Ignoruje se, pokud je první bajt prázdný, nebo má hodnotu null.

Délka tohoto pole je dána hodnotou LNLTOV. Počáteční hodnota tohoto pole je 8 prázdných znaků.

IIMMN (8bajtový znakový řetězec)

Název mapy služeb formátu zpráv.

Toto je umístěno v poli IO PCB. Tato položka není povinná. Na vstupu představuje MID, na výstupu představuje MOD. Ignoruje se, pokud je první bajt prázdný nebo má hodnotu null.

Délka tohoto pole je dána LNMFMN. Počáteční hodnota tohoto pole je 8 prázdných znaků.

IIRFM (8bajtový znakový řetězec)

IBM MQ formátovat název zprávy odpovědi.

Jedná se o název formátu IBM MQ zprávy odpovědi, která bude odeslána jako odpověď na aktuální zprávu. Pravidla pro kódování jsou stejná jako pro pole *MDFMT* v deskriptoru MQMD.

Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

IIRSV (1bajtový znakový řetězec)

Vyhrazeno.

Toto je vyhrazené pole; musí být prázdné.

IISEC (1bajtový znakový řetězec)

Rozsah zabezpečení.

To označuje požadované zpracování zabezpečení IMS . Jsou definovány tyto hodnoty:

ISSCHK
Zkontrolujte rozsah zabezpečení.

AEE je postaven v řídicí oblasti, ale ne v závislé oblasti.

ISSFUL
Úplný rozsah zabezpečení.

V řídicí oblasti je sestaveno prostředí ACEE uložené v mezipaměti a v závislé oblasti je sestaveno prostředí ACEE, které není uloženo v mezipaměti. Pokud používáte ISSFUL, musíte se ujistit, že ID uživatele, pro které je sestaveno ACEE, má přístup k prostředkům používaným v závislé oblasti.

Pokud ISSCHK a ISSFUL nejsou uvedeny pro toto pole, předpokládá se ISSCHK.

Počáteční hodnota tohoto pole je ISSCHK.

IISID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

IISIDV
Identifikátor pro strukturu záhlaví informací IMS .

Počáteční hodnota tohoto pole je IISIDV.

IITID (16bajtový bitový řetězec)

Identifikátor instance transakce.

Toto pole je používáno výstupními zprávami z produktu IMS , takže je ignorováno na prvním vstupu. Je-li parametr *IITST* nastaven na hodnotu ITSIC, musí být uveden v dalším vstupu a ve všech následných vstupech, aby produkt IMS mohl korelovat zprávy se správnou konverzací. Lze použít následující speciální hodnotu:

ITINON
Žádné ID instance transakce.

Délka tohoto pole je dána hodnotou LNTIID. Počáteční hodnota tohoto pole je ITINON.

IITST (1bajtový znakový řetězec)

Stav transakce.

Označuje stav konverzace IMS . Tato volba je při prvním vstupu ignorována, protože neexistuje žádná konverzace. Na následných vstupech označuje, zda je konverzace aktivní nebo ne. Na výstupu je nastavena pomocí IMS. Hodnota musí být jedna z následujících:

ITSIC
V rozhovoru.

ITSNIC
Ne v konverzaci.

ITSARC
Vrátit stavová data transakce v architektonické podobě.

Tato hodnota se používá pouze s příkazem IMS `/DISPLAY TRAN` . Způsobí, že se data stavu transakce vrátí ve formě architektury IMS místo ve formě znaků. Další podrobnosti viz Psaní IMS transakčních programů prostřednictvím IBM MQ .

Počáteční hodnota tohoto pole je ITSNIC.

IIVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

IIVER1
Číslo verze pro strukturu záhlaví informací IMS .

Následující konstanta určuje číslo verze aktuální verze:

IIVERC
Aktuální verze struktury záhlaví informací IMS .

Počáteční hodnota tohoto pole je IIVER1.

Počáteční hodnoty

Tabulka 705. Pole v MQIIH

Název pole	Název konstanty	Hodnota konstanty
IISID	IISIDV	'IIH¬'
IIVER	IIVER1	1
IILEN	IILEN1	84
IIENC	Není	0
IICSI	Není	0
IIFMT	FMNONE	Mezery
IIFLG	IINONE	0
IILTO	Není	Mezery
IIMMN	Není	Mezery
IIRFM	FMNONE	Mezery
IIAUT	IAUNON	Mezery
IITID	ITINON	Hodnoty null
IITST	ITSNIC	''

Tabulka 705. Pole v MQIIH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
IICMT	ICMCTS	'0'
IISEC	ISSCHK	'C'
IIRSV	Není	''

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQIIH Structure
D*
D* Structure identifier
D  IISID                 1      4    INZ('IIH ')
D* Structure version number
D  IIVER                 5      8I 0 INZ(1)
D* Length of MQIIH structure
D  IILEN                 9     12I 0 INZ(84)
D* Reserved
D  IIENC                13     16I 0 INZ(0)
D* Reserved
D  IICSI                17     20I 0 INZ(0)
D* MQ format name of data that followsMQIIH
D  IIFMT                21     28    INZ('        ')
D* Flags
D  IIFLG                29     32I 0 INZ(0)
D* Logical terminal override
D  IILTO                33     40    INZ
D* Message format services map name
D  IIMMN                41     48    INZ
D* MQ format name of reply message
D  IIRFM                49     56    INZ('        ')
D* RACF password or passticket
D  IIAUT                57     64    INZ('        ')
D* Transaction instance identifier
D  IITID                65     80    INZ(X'00000000000000-
D                                    000000000000000000')
D* Transaction state
D  IITST                81     81    INZ(' ')
D* Commit mode
D  IICMT                82     82    INZ('0')
D* Security scope
D  IISEC                83     83    INZ('C')
D* Reserved
D  IIRSV                84     84    INZ
```

IBM i MQIMPO (Volby vlastnosti dotazové zprávy) na systému IBM i

Struktura MQIMPO umožňuje aplikacím určit volby, které řídí, jak jsou zjišťovány vlastnosti zpráv.

Přehled

Účel: Struktura je vstupní parametr pro volání MQINQMP.

Znaková sada a kódování: Data v objektu MQIMPO musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1099
- "Počáteční hodnoty" na stránce 1104
- "Deklarace RPG" na stránce 1104

Pole

Struktura MQIMPO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

IPOPT (10místné celé číslo se znaménkem)

Následující volby řídí akci MQINQMP. Můžete uvést jednu nebo více těchto voleb. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace). Jsou zaznamenány neplatné kombinace voleb; všechny ostatní kombinace jsou platné.

Volby dat hodnoty: Následující volby se vztahují ke zpracování dat hodnoty, když je vlastnost načtena ze zprávy.

IPCVAL
Tato volba vyžaduje, aby byla hodnota vlastnosti převedena tak, aby byla v souladu s hodnotami *IPREQCSI* a *IPREQENC* určenými před voláním MQINQMP vrátí hodnotu vlastnosti v oblasti *Value* .

- Pokud je převod úspěšný, pole *IPRETCSI* a *IPRETENC* jsou nastavena na stejnou hodnotu jako pole *IPREQCSI* a *IPREQENC* při návratu z volání MQINQMP.
- Pokud se převod nezdaří, ale volání MQINQMP se jinak dokončí bez chyby, hodnota vlastnosti se vrátí nepřevedená.

  Pokud je vlastnost řetězec, pole *IPRETCSI* a *IPRETENC* jsou nastavena na znakovou sadu a kódování nepřevedeného řetězce.

  Kód dokončení je v tomto případě CCWARN s kódem příčiny RC2466. Kurzor vlastnosti je rozšířen na vrácenou vlastnost.

Pokud se hodnota vlastnosti během převodu rozbalí a překročí velikost parametru **Value** , hodnota se vrátí nepřevedená s kódem dokončení CCFAIL; kód příčiny je nastaven na RC2469.

Parametr **DataLength** volání MQINQMP vrací délku, na kterou by byla hodnota vlastnosti převedena, aby mohla aplikace určit velikost vyrovnávací paměti potřebnou pro přizpůsobení převedené hodnoty vlastnosti. Kurzor vlastnosti je nezměněn.

Tato volba také vyžaduje, aby:

- Pokud název vlastnosti obsahuje zástupný znak a
- Pole *IPRETNAMECHRP* je inicializováno s adresou nebo offsetem pro vrácené jméno,

pak je vrácený název převeden tak, aby odpovídal hodnotám *IPREQCSI* a *IPREQENC* .

- Pokud je převod úspěšný, pole *VSCCSID* pro *IPRETNAMECHRP* a kódování vráceného názvu se nastaví na vstupní hodnotu *IPREQCSI* a *IPREQENC*.
- Pokud se převod nezdaří, ale volání MQINQMP se jinak dokončí bez chyby nebo varování, vrácený název se nepřevede. Kód dokončení je v tomto případě CCWARN s kódem příčiny RC2492.

  Kurzor vlastnosti je rozšířen na vrácenou vlastnost. RC2466 se vrátí, pokud se hodnota i název nepřevedou.

Pokud se vrácený název během převodu rozšíří a překročí velikost pole *VSBufsize* v souboru *RequestedName*, vrácený řetězec zůstane nepřevedený s kódem dokončení CCFAIL a kód příčiny je nastaven na RC2465.

Pole *VSLength* struktury MQCHARV vrací délku, na kterou by byla hodnota vlastnosti převedena, aby mohla aplikace určit velikost vyrovnávací paměti potřebnou pro uložení převedené hodnoty vlastnosti. Kurzor vlastnosti je nezměněn.

IPCTYP
Tato volba vyžaduje, aby byla hodnota vlastnosti převedena z aktuálního datového typu na datový typ určený parametrem **Type** volání MQINQMP.

- Je-li převod úspěšný, parametr **Type** se při návratu volání MQINQMP nezmění.
- Pokud se převod nezdaří, ale volání MQINQMP se jinak dokončí bez chyby, volání se nezdaří s příčinou RC2470. Kurzor vlastnosti je nezměněn.

Pokud převod datového typu způsobí, že se hodnota během převodu rozbalí a převedená hodnota překročí velikost parametru **Value** , hodnota se vrátí nepřevedená s kódem dokončení CCFAIL a kód příčiny je nastaven na RC2469.

Parametr **DataLength** volání MQINQMP vrací délku, na kterou by byla hodnota vlastnosti převedena, aby mohla aplikace určit velikost vyrovnávací paměti potřebnou pro přizpůsobení převedené hodnoty vlastnosti. Kurzor vlastnosti je nezměněn.

Pokud hodnota parametru **Type** volání MQINQMP není platná, volání selže s příčinou RC2473.

Pokud požadovaný převod datového typu není podporován, volání selže s příčinou RC2470. Podporovány jsou následující převody datových typů:

Tabulka 706. Podporované převody datových typů

Datový typ vlastnosti	Podporované cílové datové typy
Počet TYPOBOLŮ	TYPSTR, TYPI8, TYPI16, TYPI32, TYPI64
TYPBST	TYPSTR
TYPI8	TYPSTR, TYPI16, TYPI32, TYPI64
TYPI16	TYPSTR, TYPI32, TYPI64
TYPI32	TYPSTR, TYPI64
TYPI64	TYPSTR
TYPF32	TYPSTR, TYPF64
TYPF64	TYPSTR
TYPSTR	TYPBOL, TYPI8, TYPI16, TYPI32, TYPI64, TYPF32, TYPF64
TYPNUL	Není

Obecná pravidla pro podporované převody jsou následující:

- Číselné hodnoty vlastností lze převést z jednoho datového typu na jiný za předpokladu, že během převodu nebudou ztracena žádná data.

  Například hodnotu vlastnosti s datovým typem TYPI32 lze převést na hodnotu s datovým typem TYPI64, ale nelze ji převést na hodnotu s datovým typem TYPI16.
- Hodnotu vlastnosti libovolného datového typu lze převést na řetězec.
- Hodnotu vlastnosti řetězce lze převést na jakýkoli jiný datový typ za předpokladu, že je řetězec pro převod správně naformátován. Pokud se aplikace pokusí převést hodnotu řetězcové vlastnosti, která není správně naformátována, IBM MQ vrátí kód příčiny RC2472.
- Pokud se aplikace pokusí o převod, který není podporován, IBM MQ vrátí kód příčiny RC2470.

Specifická pravidla pro převod hodnoty vlastnosti z jednoho datového typu na jiný jsou následující:

- Při převodu hodnoty vlastnosti TYPBOL na řetězec se hodnota TRUE převede na řetězec "TRUE" a hodnota false se převede na řetězec "FALSE".
- Při převodu hodnoty vlastnosti TYPBOL na číselný datový typ se hodnota TRUE převede na hodnotu jedna a hodnota FALSE se převede na nulu.
- Při převodu hodnoty vlastnosti řetězce na hodnotu TYPBOL je řetězec "TRUE" nebo "1" převeden na hodnotu TRUE a řetězec "FALSE" nebo "0" je převeden na hodnotu FALSE.

  Všimněte si, že výrazy "TRUE" a "FALSE" nerozlišují velká a malá písmena.

  Žádný jiný řetězec nelze převést; IBM MQ vrací kód příčiny RC2472.

- Při převodu hodnoty vlastnosti řetězce na hodnotu s datovým typem TYPI8, TYPI16, TYPI32 nebo TYPI64musí mít řetězec následující formát:

```
[blanks][sign]digits
```

Význam komponent řetězce je následující:

blanks
Volitelné úvodní prázdné znaky

sign
Volitelný znak plus (+) nebo znak minus (-).

digits
Souvislá posloupnost číselných znaků (0-9). Musí být uveden alespoň jeden znak číslice.

Po posloupnosti číselných znaků může řetězec obsahovat další znaky, které nejsou číselnými znaky, ale převod se zastaví, jakmile je dosaženo prvního z těchto znaků. Předpokládá se, že řetězec představuje desetinné celé číslo.

IBM MQ vrací kód příčiny RC2472 , pokud není řetězec správně naformátován.

- Při převodu hodnoty vlastnosti řetězce na hodnotu s datovým typem TYPF32 nebo TYPF64musí mít řetězec následující formát:

```
[blanks][sign]digits[.digits][e_char[e_sign]e_digits]
```

Význam komponent řetězce je následující:

blanks
Volitelné úvodní prázdné znaky

sign
Volitelný znak plus (+) nebo znak minus (-).

digits
Souvislá posloupnost číselných znaků (0-9). Musí být uveden alespoň jeden znak číslice.

e_char
Exponent, který je buď "E", nebo "e".

e_sign
Volitelný znak plus (+) nebo minus (-) pro exponent.

e_digits
Souvislá posloupnost číselných znaků (0-9) pro exponent. Pokud řetězec obsahuje znak exponentu, musí být uveden alespoň jeden znak číslice.

Po posloupnosti číselných znaků nebo volitelných znaků reprezentujících exponent může řetězec obsahovat jiné znaky, které nejsou číselnými znaky, ale převod se zastaví, jakmile je dosaženo prvního z těchto znaků. Předpokládá se, že řetězec představuje desetinné číslo s pohyblivou řádovou čárkou s exponentem, který je mocninou 10.

IBM MQ vrací kód příčiny RC2472 , pokud není řetězec správně naformátován.

- Při převodu hodnoty číselné vlastnosti na řetězec je hodnota převedena na řetězcovou reprezentaci hodnoty jako desetinné číslo, nikoli řetězec obsahující pro tuto hodnotu znak ASCII. Například celé číslo 65 se převede na řetězec "65", nikoli na řetězec "A".
- Při převodu hodnoty vlastnosti bajtového řetězce na řetězec je každý bajt převeden na dva hexadecimální znaky, které představují bajt. Například bajtové pole {0xF1, 0x12, 0x00, 0xFF} je převedeno na řetězec "F11200FF".

IPQLEN
Zadejte dotaz na typ a délku hodnoty vlastnosti. Délka je vrácena v parametru **DataLength** volání MQINQMP. Hodnota vlastnosti není vrácena.

Je-li zadána vyrovnávací paměť *ReturnedName* , je pole *VSLength* struktury MQCHARV vyplněno délkou názvu vlastnosti. Název vlastnosti není vrácen.

Volby iterace: Následující volby se týkají iterace nad vlastnostmi s použitím názvu se zástupným znakem.

IPINQF
Dotázat se na první vlastnost, která odpovídá zadanému názvu. Po tomto volání se na vrácené vlastnosti ustanoví kurzor.

Toto je výchozí hodnota.

Volbu IPINQC lze následně použít s voláním MQINQMP, je-li vyžadováno, k opětovnému dotazu na stejnou vlastnost.

Všimněte si, že existuje pouze jeden kurzor vlastnosti; proto, pokud název vlastnosti zadaný ve volání MQINQMP změní kurzor na reset.

Tato volba není platná s žádnou z následujících voleb:

IPINQN
IPINQC-nestandardní

IPINQN
Dotazuje se na další vlastnost, která odpovídá zadanému názvu, a pokračuje v hledání z kurzoru vlastnosti. Kurzor je rozšířen na vrácenou vlastnost.

Pokud se jedná o první volání MQINQMP pro uvedený název, vrátí se první vlastnost, která odpovídá zadanému názvu.

Volbu IPINQC lze v případě potřeby následně použít s voláním MQINQMP, aby se znovu dotazovala na stejnou vlastnost.

Pokud byla vlastnost pod kurzorem odstraněna, vrátí MQINQMP další odpovídající vlastnost následující po té, která byla odstraněna.

Je-li přidána vlastnost, která odpovídá zástupnému znaku, zatímco probíhá iterace, tato vlastnost může nebo nemusí být vrácena během dokončení iterace. Vlastnost je vrácena, jakmile se iterace restartuje pomocí IPINQF.

Vlastnost odpovídající zástupnému znaku, který byl odstraněn během probíhající iterace, není vrácena po jejím odstranění.

Tato volba není platná s žádnou z následujících voleb:

IPINQF
IPINQC-nestandardní

IPINQC-nestandardní
Načtěte hodnotu vlastnosti, na kterou ukazuje kurzor vlastnosti. Vlastnost, na kterou ukazuje kurzor vlastnosti, je ta, která byla naposledy dotazována pomocí volby IPINQF nebo IPINQN.

Kurzor vlastnosti je resetován při opětném použití manipulátoru zprávy, při zadání manipulátoru zprávy v poli *MsgHandle* MQGMO ve volání MQGET nebo při zadání manipulátoru zprávy v polích *OriginalMsgHandle* nebo *NewMsgHandle* struktury MQPMO ve volání MQPUT.

Pokud je tato volba použita, když kurzor vlastnosti ještě nebyl zaveden, nebo pokud byla vlastnost, na kterou ukazoval kurzor vlastnosti, odstraněna, volání selže s kódem dokončení CCFAIL a příčinou RC2471.

Tato volba není platná s žádnou z následujících voleb:

IPINQF
IPINQN

Pokud není požadována žádná z dříve popsaných voleb, lze použít následující volbu:

IPNONE
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

IPNONE pomáhá s dokumentací programu; není zamýšleno, aby tato volba byla použita s jinou, ale protože její hodnota je nula, nelze takové použití zjistit.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je IPINQF.

IPREQCSI (10místné celé číslo se znaménkem)

Znaková sada, na kterou má být hodnota dotazované vlastnosti převedena, pokud je hodnota znakový řetězec. Toto je také znaková sada, do které se má *ReturnedName* převést, když je uvedeno IPCVAL nebo IPCTYP.

Počáteční hodnota tohoto pole je CSAPL.

IPREQENC (10místné celé číslo se znaménkem)

Toto je kódování, do kterého se má převést hodnota dotazované vlastnosti, když je uvedeno IPCVAL nebo IPCTYP.

Počáteční hodnota tohoto pole je ENNAT.

IPRE1 (10místné celé číslo se znaménkem)

Toto je vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak.

IPRETCSI (10místné celé číslo se znaménkem)

Na výstupu se jedná o znakovou sadu vrácené hodnoty, pokud parametr **Type** volání MQINQMP je TYPSTR.

Pokud je uvedena volba IPCVAL a konverze byla úspěšná, pole *ReturnedCCSID* , při návratu, je stejná hodnota jako předaná hodnota.

Počáteční hodnota tohoto pole je nula.

IPRETENC (10místné celé číslo se znaménkem)

Na výstupu se jedná o kódování vrácené hodnoty.

Pokud je uvedena volba IPCVAL a konverze byla úspěšná, pole *ReturnedEncoding* , při návratu, je stejná hodnota jako předaná hodnota.

Počáteční hodnota tohoto pole je ENNAT.

IPRETNAMCHRP (10místné celé číslo se znaménkem)

Skutečný název dotazovaného objektu.

Na vstupu lze vyrovnávací paměť řetězců předat pomocí pole *VSPtr* nebo *VSOffset* struktury MQCHARV. Délka vyrovnávací paměti řetězců je určena pomocí pole *VSBufsize* struktury MQCHARV.

Při návratu z volání MQINQMP je vyrovnávací paměť řetězců dokončena s názvem vlastnosti, která byla zjišťována, za předpokladu, že vyrovnávací paměť řetězců byla dostatečně dlouhá, aby plně obsahovala název. Pole *VSLength* struktury MQCHARV je vyplněno délkou názvu vlastnosti. Pole *VSCCSID* struktury MQCHARV je vyplněno, aby označilo znakovou sadu vráceného názvu, bez ohledu na to, zda se nezdařil převod názvu.

Toto je vstupní/výstupní pole. Počáteční hodnota tohoto pole je MQCHARV_DEFAULT.

IPSID (10místné celé číslo se znaménkem)

Jedná se o identifikátor struktury. Hodnota musí být:

IPSIDV
Identifikátor pro strukturu voleb vlastností dotazové zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je IPSIDV.

IPTYP (10místné celé číslo se znaménkem)

Řetězcová reprezentace datového typu vlastnosti.

Pokud byla vlastnost určena v záhlaví MQRFH2 a atribut MQRFH2 `dt` nebyl rozpoznán, lze toto pole použít k určení datového typu vlastnosti. *TypeString* je vráceno v kódované znakové sadě 1208 (UTF-8) a je to prvních osm bajtů hodnoty atributu `dt` vlastnosti, kterou se nepodařilo rozpoznat.

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je nulový řetězec v programovacím jazyku C a 8 prázdných znaků v jiných programovacích jazycích.

IPVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury. Hodnota musí být:

IPVER1
Číslo verze pro strukturu voleb vlastností dotazové zprávy.

Následující konstanta určuje číslo verze aktuální verze:

IPVERC-bezpečnostní zařízení
Aktuální verze struktury voleb vlastností dotazové zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je IPVER1.

Počáteční hodnoty

Tabulka 707. Pole v MQIPMO

Název pole	Název konstanty	Hodnota konstanty
IPSID	IPSIDV	'IMPO'
IPVER	IPVER1	1
IPOPT	IPINQF	
IPREQENC	ENNAT	
IPREQCSI	CSAPL	
IPRETENC	ENNAT	
IPRETCSI	0	
IPRE1	0	
IPRETNAMCHRP		
IPTYP		mezery

Deklarace RPG

```
D* MQIMPO Structure
D*
D*
D* Structure identifier
D IPSID         1   4  INZ('IMPO')
D*
```

```
D* Structure version number
D IPVER         5   8I 0 INZ(1)
D*
** Options that control the action of
D* MQINQMP
D IPOPT         9   12I 0 INZ(0)
D*
D* Requested encoding of Value
D IPREQENC       13   16I 0 INZ(273)
D*
** Requested character set identifier
D* of Value
D IPREQCSI       17   20I 0 INZ(-3)
D*
D* Returned encoding of Value
D IPRETENC       21   24I 0 INZ(273)
D*
** Returned character set identifier of
D* Value
D IPRETCSI       25   28I 0 INZ(0)
D*
D* Reserved
D IPRE1         29   32I 0 INZ(0)
D*
D* Returned property name
D* Address of variable length string
D IPRETNAMCHRP     33   48*  INZ(*NULL)
D* Offset of variable length string
D IPRETNAMCHRO     49   52I 0 INZ(0)
D* Size of buffer
D IPRETNAMVSBS     53   56I 0 INZ(-1)
D* Length of variable length string
D IPRETNAMCHRL     57   60I 0 INZ(0)
D* CCSID of variable length string
D IPRETNAMCHRC     61   64I 0 INZ(-3)
D*
D* Property data type as a string
D IPTYP         65   72  INZ
```

IBM i MQMD (deskriptor zprávy) na IBM i

Přehled

Účel: Struktura MQMD obsahuje řídicí informace, které doprovázejí data aplikace, když zpráva prochází mezi odesílající a přijímající aplikací. Struktura je vstupní/výstupní parametr pro volání MQGET, MQPUT a MQPUT1 .

Verze: Aktuální verze MQMD je MDVER2. Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi MQMD podporovanou prostředím, ale s počáteční hodnotou pole MDVER nastavenou na hodnotu MDVER1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole MDVER na číslo verze požadované verze.

Deklarace pro strukturu version-1 je k dispozici s názvem MQMD1.

Znaková sada a kódování: Data v deskriptoru MQMD musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódováním lokálního správce front daným kódem ENNAT. Pokud je však aplikace spuštěna jako IBM MQ MQI client, musí být struktura ve znakové sadě a kódování klienta.

Pokud odesílající a přijímající správci front používají různé znakové sady nebo kódování, data v MQMD se převedou automaticky. Převod deskriptoru MQMD aplikací není nutný.

- “Použití různých verzí MQMD” na stránce 1106
- “kontext zprávy” na stránce 1106
- “Vypršení zprávy” na stránce 1106
- “Pole” na stránce 1107
- “Počáteční hodnoty” na stránce 1146
- “Deklarace RPG” na stránce 1147

Použití různých verzí MQMD

version-2 MQMD je obecně ekvivalentní s použitím version-1 MQMD a předponou dat zprávy se strukturou MQMDE. Pokud však všechna pole ve struktuře MQMDE mají své výchozí hodnoty, lze MQMDE vynechat. version-1 MQMD plus MQMDE se používají tak, jak je popsáno dále v této části.

- Pokud u volání MQPUT a MQPUT1 aplikace poskytuje MQMD version-1 , může aplikace volitelně přidat k datům zprávy předponu MQMDE a nastavit pole MDFMT v MQMD na hodnotu FMMDE tak, aby označovala přítomnost MQMDE. Pokud aplikace neposkytne prostředí MQMDE, správce front předpokládá výchozí hodnoty pro pole v prostředí MQMDE.

  Poznámka: Několik polí, která existují ve volání MQMD version-2 , ale nikoli version-1 MQMD, jsou vstupní/výstupní pole ve voláních MQPUT a MQPUT1 . Správce front však nevrací žádné hodnoty v ekvivalentních polích ve výstupu MQMDE z volání MQPUT a MQPUT1 . Pokud aplikace tyto výstupní hodnoty vyžaduje, musí použít MQMD version-2 .
- Pokud aplikace ve volání MQGET poskytuje MQMD version-1 , správce front před zprávu vrácenou s MQMDE opatřuje předponou, ale pouze v případě, že jedno nebo více polí v MQMDE má jinou než výchozí hodnotu. Pole MDFMT v deskriptoru MQMD bude mít hodnotu FMMDE, která bude indikovat přítomnost MQMDE.

Výchozí hodnoty, které správce front použil pro pole v prostředí MQMDE, jsou stejné jako počáteční hodnoty těchto polí, viz Tabulka 709 na stránce 1146.

Nachází-li se zpráva v přenosové frontě, jsou některá pole v deskriptoru MQMD nastavena na konkrétní hodnoty. Podrobnosti naleznete v části “MQXQH (záhlaví přenosové fronty) na IBM i” na stránce 1239 .

kontext zprávy

Určitá pole v deskriptoru MQMD obsahují kontext zprávy. Obvykle:

- *Kontext identity* souvisí s aplikací, která původně vložila zprávu.
- *Kontext původu* souvisí s aplikací, která naposledy vložila zprávu.
- *Uživatelský kontext* souvisí s aplikací, která původně vložila zprávu.

Tyto dvě aplikace mohou být stejné, ale mohou být také různé aplikace (například když je zpráva předána z jedné aplikace do druhé).

Ačkoli identita a původní kontext mají obvykle dříve popsané významy, obsah obou typů polí kontextu v deskriptoru MQMD ve skutečnosti závisí na volbách PM*, které jsou určeny při vložení zprávy. V důsledku toho nemusí kontext identity nutně souviset s aplikací, která zprávu původně vložila, a kontext původu nemusí nutně souviset s aplikací, která zprávu vložila naposledy-závisí na návrhu sady aplikací.

Existuje jedna třída aplikace, která nikdy nemění kontext zprávy, konkrétně agent kanálu zpráv (MCA). Adaptéry MCA, které přijímají zprávy od vzdálených správců front, používají volbu kontextu PMSETA pro volání MQPUT nebo MQPUT1 . To umožňuje přijímajícímu MCA zachovat přesně kontext zprávy, který byl spolu se zprávou z odesílajícího MCA. Výsledkem však je, že původní kontext nesouvisí s aplikací, která naposledy vložila zprávu (přijímající MCA), ale místo toho souvisí s dřívější aplikací, která vložila zprávu (případně samotnou původní aplikaci).

Další informace viz Kontext zprávy.

Vypršení zprávy

Zprávy, které vypršely v načtené frontě (fronta, která byla otevřena), jsou automaticky odebrány z fronty v přiměřené době po uplynutí jejich platnosti. Některé další nové funkce tohoto vydání produktu IBM MQ mohou vést k tomu, že načtené fronty budou skenovány méně často než v předchozí verzi produktu, avšak zprávy s vypršenou platností v načtených frontách budou vždy odebrány v přiměřené lhůtě po vypršení jejich platnosti.

Pole

Struktura MQMD obsahuje následující pole; pole jsou popsána v abecedním pořadí:

MDACC (32bitový bitový řetězec)

Účtovací token.

Toto je část *kontextu identity* zprávy. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Produkt *MDACC* umožňuje aplikaci, aby způsobila, že práce vykonaná v důsledku zprávy bude řádně zpoplatněna. Správce front považuje tyto informace za řetězec bitů a nekontroluje jejich obsah.

Když správce front vygeneruje tyto informace, nastaví se takto:

- První bajt pole je nastaven na délku účetních informací přítomných v následujících bajtech; tato délka je v rozsahu od nuly do 30 a je uložena v prvním bajtu jako binární celé číslo.
- Druhý a následující bajty (jak jsou uvedeny v poli délky) jsou nastaveny na informace o účtování odpovídající prostředí.
  - z/OS V systému z/OS jsou informace o účtování nastaveny na:
    - V případě dávky z/OS se jedná o účtovací informace z karty JES JOB nebo z příkazu JES ACCT v kartě EXEC (čárkové oddělovače jsou změněny na X'FF '). Tato informace je v případě potřeby oříznuta na 31 bajtů.
    - V případě TSO se jedná o číslo účtu uživatele.
    - V případě operačního systému CICSse jedná o identifikátor jednotky práce LU 6.2 (UEPUOWDS) (26 bajtů).
    - V případě systému IMSse jedná o 8znakový název PSB zřetězený s 16znakovým tokenem zotavení IMS .
  - IBM i V systému IBM ijsou účtovací informace nastaveny na účtovací kód úlohy.
  - Linux AIX V systému AIX and Linuxjsou účtovací informace nastaveny na číselný identifikátor uživatele ve znacích ASCII.
  - Windows V systému Windowsjsou informace o účtování nastaveny na identifikátor zabezpečení Windows NT (SID) v komprimovaném formátu. Identifikátor SID jedinečně identifikuje identifikátor uživatele uložený v poli *MDUID* . Je-li identifikátor SID uložen v poli *MDACC* , je vynechán 6bajtový identifikátor autority (který se nachází ve třetím a následných bajtech identifikátoru SID). Pokud je například identifikátor SID Windows NT dlouhý 28 bajtů, do pole *MDACC* se uloží 22 bajtů informací o identifikátoru SID.
- Poslední bajt je nastaven na typ evidenčního tokenu jedna z následujících hodnot:

  Akumulační nádrže
  CICS Identifikátor LUOW.

  ATTDOS-operační systém
  Výchozí účtovací token PC DOS.

  ATTWNT
  Windows identifikátor zabezpečení.

  ATT400
  IBM i token evidence.

  Akumulační nádrže ATTUNX
  AIX and Linux číselný identifikátor.

  ATTUSR pro uživatele
  Token evidence definovaný uživatelem.

  Akumulační nádrže
  Neznámý typ účtovacího tokenu.

Typ evidenčního tokenu je nastaven na explicitní hodnotu pouze v následujících prostředích:

- 

AIX
- 

IBM i
- 

Windows

a pro systém IBM MQ MQI clients připojený k těmto systémům.

V jiných prostředích je typ evidenčního tokenu nastaven na hodnotu ATTUNK. V těchto prostředích lze pole MDPAT použít k odpočtu typu přijatého účtovacího tokenu.

- Všechny ostatní bajty jsou nastaveny na binární nulu.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PMO** uvedeno PMSETI nebo PMSETA. Není-li zadána hodnota PMSETI ani PMSETA, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor MDACC , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude to hodnota MDACC , která je uchována se zprávou, pokud je zachována (viz popis PMRET v části “MQPMO (Vložit-volby zpráv) na IBM i” na stránce 1168 , kde jsou další podrobnosti o zachovaných publikacích), ale nepoužívá se jako MDACC , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání MDACC ve všech publikacích, které jim byly odeslány. Pokud zpráva nemá žádný kontext, pole je zcela binární nula.

Toto je výstupní pole pro volání MQGET.

Toto pole nepodléhá žádnému překladu založenému na znakové sadě správce front-pole je považováno za řetězec bitů a nikoli za řetězec znaků.

Správce front neprovádí žádnou činnost s informacemi v tomto poli. Aplikace musí interpretovat informace, pokud chce tyto informace použít pro účely účtování.

Pro pole *MDACC* lze použít následující speciální hodnotu:

ACNONE
Není uveden žádný token evidence.

Hodnota je binární nula pro délku pole.

Délka tohoto pole je dána LNACCT. Počáteční hodnota tohoto pole je ACNONE.

MDAID (32bajtový znakový řetězec)

Údaje o aplikaci týkající se identity.

Toto je část *kontextu identity* zprávy. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

MDAID je informace, která je definována sadou aplikací, a lze ji použít k poskytnutí dalších informací o zprávě nebo jejím původci. Správce front považuje tyto informace za znaková data, ale nedefinuje jejich formát. Když správce front generuje tyto informace, jsou zcela prázdné.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PMO** uvedeno PMSETI nebo PMSETA. Je-li přítomen znak null, správce front převede hodnotu null a všechny následující znaky na mezery. Není-li zadána hodnota PMSETI ani PMSETA, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor MDAID , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude to hodnota MDAID , která je uchována se zprávou, pokud je uchována (viz popis PMRET, kde jsou další podrobnosti o zachovaných publikacích), ale nepoužívá se jako MDAID , když je zpráva odeslána jako publikování odběratelům,

protože poskytují hodnotu k přepsání MDAID ve všech publikacích, která jim byla odeslána. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou LNAIDD. Počáteční hodnota tohoto pole je 32 prázdných znaků.

MDAOD (4bajtový znakový řetězec)

Údaje o žádosti týkající se původu.

Toto je část *původního kontextu* zprávy. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

MDAOD je informace definovaná sadou aplikací, kterou lze použít k poskytnutí dalších informací o původu zprávy. Může být například nastaven aplikacemi, které jsou spuštěny s vhodným oprávněním uživatele, aby označily, zda jsou data identity důvěryhodná.

Správce front považuje tyto informace za znaková data, ale nedefinuje jejich formát. Když správce front generuje tyto informace, jsou zcela prázdné.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PMO** uvedeno PMSETA. Jakékoli informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota PMSETA, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor MDAOD , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude to hodnota MDAOD , která je uchována se zprávou, pokud je uchována (viz popis PMRET, kde jsou další podrobnosti o zachovaných publikacích), ale nepoužívá se jako MDAOD , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání MDAOD ve všech publikacích, která jim byla odeslána. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou LNAORD. Počáteční hodnota tohoto pole je 4 prázdné znaky.

MDBOC (10místné celé číslo se znaménkem)

Počítadlo vrácení.

Jedná se o počet případů, kdy byla zpráva dříve vrácena voláním MQGET jako součást pracovní jednotky a následně vrácena zpět. Poskytuje se jako pomůcka pro aplikaci při zjišťování chyb zpracování, které jsou založeny na obsahu zprávy. Tento počet vylučuje volání MQGET, která uváděla některou z voleb GMBRW*.

Přesnost tohoto počtu je ovlivněna atributem fronty **HardenGetBackout** ; viz “Atributy pro fronty” na stránce 1360.

Toto je výstupní pole pro volání MQGET. Pro volání MQPUT a MQPUT1 se ignoruje. Počáteční hodnota tohoto pole je 0.

MDCID (24bajtový bitový řetězec)

Identifikátor korelace.

Jedná se o bajtový řetězec, který může aplikace použít k přiřazení jedné zprávy k jiné nebo k přiřazení zprávy k jiné práci, kterou aplikace provádí. Identifikátor korelace je trvalou vlastností zprávy a přetrvává po restartování správce front. Vzhledem k tomu, že identifikátor korelace je bajtový řetězec a nikoli znakový řetězec, není identifikátor korelace převeden mezi znakovými sadami, pokud zpráva přechází z jednoho správce front do jiného.

Pro volání MQPUT a MQPUT1 může aplikace zadat libovolnou hodnotu. Správce front odešle tuto hodnotu se zprávou a doručí ji aplikaci, která vydá požadavek na získání zprávy.

Pokud aplikace uvádí PMNCID, správce front vygeneruje jedinečný identifikátor korelace, který se odešle se zprávou, a také se vrátí odesílající aplikaci na výstupu z volání MQPUT nebo MQPUT1 .

Tento generovaný identifikátor korelace je uchováván spolu se zprávou, pokud je zachován, a používá se jako identifikátor korelace, když je zpráva odeslána jako publikování odběratelům, kteří zadali hodnotu CINONE v poli SDCID v MQSD předaném při volání MQSUB.

Další podrobnosti o zachovaných publikacích naleznete v části "MQPMO (Vložit-volby zpráv) na IBM i" na stránce 1168 .

Když správce front nebo agent kanálu zpráv vygeneruje zprávu sestavy, nastaví pole MDCID způsobem určeným v poli MDREP původní zprávy, buď ROCMTC, nebo ROPCI. Aplikace, které generují zprávy sestav, by to měly také provést.

Pro volání MQGET je MDCID jedním z pěti polí, které lze použít k výběru konkrétní zprávy, která má být načtena z fronty. Podrobnosti o tom, jak zadat hodnoty pro toto pole, naleznete v popisu pole MDMID .

Zadání parametru CINONE jako identifikátoru korelace má stejný efekt jako nezadání parametru MOCORI, tj. jakýkoli identifikátor korelace se bude shodovat.

Je-li v parametru **GMO** ve volání MQGET zadána volba GMMUC, bude toto pole ignorováno.

Při návratu z volání MQGET je pole MDCID nastaveno na identifikátor korelace vrácené zprávy (pokud existuje).

Lze použít následující speciální hodnoty:

CINONE
Není uveden žádný identifikátor korelace.

Hodnota je binární nula pro délku pole.

CINEWS
Zpráva je začátek nové relace.

Tato hodnota je rozpoznána produktem CICS bridge jako označení začátku nové relace, tj. začátku nové posloupnosti zpráv.

Pro volání MQGET se jedná o vstupní/výstupní pole. Pro volání MQPUT a MQPUT1 se jedná o vstupní pole, pokud není uvedeno PMNCID, a výstupní pole, pokud je uvedeno PMNCID. Délka tohoto pole je dána hodnotou LNCID. Počáteční hodnota tohoto pole je CINONE.

MDCSI (10místné celé číslo se znaménkem)

Určuje identifikátor znakové sady znakových dat ve zprávě.

Poznámka: Znaková data v deskriptoru MQMD a další datové struktury produktu IBM MQ , které jsou parametry volání, musí být ve znakové sadě správce front. Tento atribut je definován atributem **CodedCharSetId** správce front. Podrobnosti o tomto atributu naleznete v části "Atributy pro správce front v systému IBM i" na stránce 1391 .

Lze použít následující speciální hodnoty:

CSQM
Identifikátor znakové sady správce front.

Znaková data ve zprávě jsou ve znakové sadě správce front.

Při volání MQPUT a MQPUT1 správce front změní tuto hodnotu v deskriptoru MQMD odeslaném spolu se zprávou na identifikátor znakové sady správce front. Výsledkem je, že hodnota CSQM není nikdy vrácena voláním MQGET.

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data ve zprávě jsou ve stejné znakové sadě jako tato struktura; jedná se o znakovou sadu správce front. (Pouze pro MQMD má CSINHT stejný význam jako CSQM).

Správce front změní tuto hodnotu v deskriptoru MQMD odeslanou spolu se zprávou na skutečný identifikátor znakové sady deskriptoru MQMD. Za předpokladu, že nedojde k žádné chybě, není hodnota CSINHT vrácena voláním MQGET.

CSINHT nelze použít, pokud je hodnota pole MDPAT v MQMD ATBRKR.

CSEMBD
Vložený identifikátor znakové sady.

Znaková data ve zprávě jsou ve znakové sadě s identifikátorem, který je obsažen v samotných datech zprávy. V datech zprávy může být vložen libovolný počet identifikátorů znakových sad, které se vztahují na různé části dat. Tato hodnota musí být použita pro zprávy PCF, které obsahují data ve směsi znakových sad. Zprávy PCF mají název formátu FMPCF.

Tuto hodnotu zadejte pouze u volání MQPUT a MQPUT1 . Pokud je zadán ve volání MQGET, zabrání převodu zprávy.

Ve voláních MQPUT a MQPUT1 změní správce front hodnoty CSQM a CSINHT v deskriptoru MQMD odeslaném se zprávou, jak bylo popsáno dříve, ale nezmění hodnotu MQMD určenou ve volání MQPUT nebo MQPUT1 . Na uvedené hodnotě se neprovádí žádná další kontrola.

Aplikace, které načítají zprávy, by měly porovnat toto pole s hodnotou, kterou aplikace očekává; pokud se hodnoty liší, aplikace bude možná muset převést znaková data ve zprávě.

Je-li ve volání MQGET zadána volba GMCONV, jedná se o vstupní/výstupní pole. Hodnota uvedená aplikací je identifikátor kódované znakové sady, na který by měla být v případě potřeby převedena data zprávy. Pokud je konverze úspěšná nebo zbytečná, hodnota se nezmění (kromě toho, že se hodnota CSQM nebo CSINHT převede na skutečnou hodnotu). Pokud je převod neúspěšný, hodnota po volání MQGET představuje identifikátor kódované znakové sady nepřevedené zprávy, která je vrácena aplikaci.

Jinak se jedná o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je CSQM.

MDENC (10místné celé číslo se znaménkem)

Číselné kódování dat zprávy.

Tato volba určuje číselné kódování číselných dat ve zprávě; nevztahuje se na číselná data v samotné struktuře MQMD. Číselné kódování definuje reprezentaci používanou pro binární celá čísla, pakovaná desetinná celá čísla a čísla s pohyblivou řádovou čárkou.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je pole platné. Je definována následující speciální hodnota:

ENNAT
Nativní kódování počítače.

Kódování je výchozí pro programovací jazyk a počítač, na kterém je aplikace spuštěna.

Poznámka: Hodnota této konstanty závisí na programovacím jazyku a prostředí. Z tohoto důvodu musí být aplikace kompilovány pomocí souborů záhlaví, makra, COPY nebo INCLUDE odpovídajících prostředí, ve kterém bude aplikace spuštěna.

Aplikace, které vkládají zprávy, by měly obvykle uvádět ENNAT. Aplikace, které načítají zprávy, by měly porovnat toto pole s hodnotou ENNAT; pokud se hodnoty liší, aplikace může potřebovat převést číselná data ve zprávě. Pomocí volby GMCONV lze požadovat, aby správce front převedl zprávu v rámci zpracování volání MQGET.

Je-li ve volání MQGET zadána volba GMCONV, jedná se o vstupní/výstupní pole. Hodnota určená aplikací je kódování, do kterého by měla být data zprávy v případě potřeby převedena. Pokud je převod úspěšný nebo zbytečný, hodnota se nezmění. Pokud je převod neúspěšný, hodnota po volání MQGET představuje kódování nepřevedené zprávy, která je vrácena aplikaci.

V jiných případech se jedná o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je ENNAT.

MDEXP (10místné celé číslo se znaménkem)

Životnost zprávy.

Jedná se o časové období vyjádřené v desetinách sekundy, nastavené aplikací, která vkládá zprávu. Zpráva se stane způsobilou k vyřazení, pokud nebyla odebrána z cílové fronty před uplynutím této doby.

Hodnota je snížena tak, aby odrážela čas, který zpráva stráví v cílové frontě, a také v jakýchkoli intermediačních přenosových frontách, pokud je vložení do vzdálené fronty. Může být také dekrementován agenty kanálů zpráv, aby odrážel doby přenosu, pokud jsou významné. Podobně může aplikace, která předává tuto zprávu do jiné fronty, snížit hodnotu, je-li to nutné, pokud tuto zprávu uchovala po dlouhou dobu. Čas vypršení platnosti je však považován za přibližný a hodnota nemusí být snížena, aby odrážela malé časové intervaly.

Když je zpráva načtena aplikací pomocí volání MQGET, představuje pole MDEXP množství původního času vypršení platnosti, které stále zbývá.

Po uplynutí doby vypršení platnosti zprávy je vhodné ji správce front zrušit. V aktuálních implementacích je zpráva vyřazena, když dojde k volání MQGET pro procházení nebo neprocházení, které by vrátilo zprávu, pokud by ještě nevypršela její platnost. Například volání MQGET bez procházení s polem GMMO v sadě MQGMO na čtení MONONE z řazené fronty FIFO způsobí, že všechny zprávy s vypršenou platností budou vyřazeny až do první zprávy s vypršenou platností. V případě fronty seřazené podle priority stejné volání zahodí zprávy s vypršenou platností s vyšší prioritou a zprávy se stejnou prioritou, které dorazily do fronty před první nevypršenou zprávou.

Aplikaci se nikdy nevrátí zpráva, jejíž platnost vypršela (buď procházením, nebo voláním MQGET bez procházení), takže hodnota v poli MDEXP deskriptoru zprávy po úspěšném volání MQGET je buď větší než nula, nebo speciální hodnota EIULIM.

Je-li zpráva vložena do vzdálené fronty, může její platnost vypršet (a může být vyřazena), zatímco je ve střední přenosové frontě, než zpráva dosáhne cílové fronty.

Sestava se vygeneruje, když je zpráva s vypršenou platností vyřazena, pokud zpráva zadala jednu z voleb sestavy ROEXP*. Není-li zadána žádná z těchto voleb, žádná taková sestava se nevygeneruje; zpráva se po tomto časovém období již nepovažuje za relevantní (možná proto, že ji později zpráva nahradila).

Jakýkoli jiný program, který vyřadí zprávy na základě času vypršení platnosti, musí také odeslat odpovídající zprávu sestavy, pokud byla požadována.

Poznámka:

1. Pokud je vložena zpráva s MDEXP časem nula, volání MQPUT nebo MQPUT1 se nezdaří s kódem příčiny RC2013; v tomto případě se nevygeneruje žádná zpráva sestavy.
2. Vzhledem k tomu, že zpráva s časem vypršení platnosti, který uplynul, nemusí být ve skutečnosti vyřazena až později, mohou se ve frontě nacházet zprávy, které překročily čas vypršení platnosti, a které proto nejsou vhodné pro načtení. Tyto zprávy se přesto započítávají do počtu zpráv ve frontě pro všechny účely, včetně spouštění hloubky.
3. Sestava vypršení platnosti je generována, pokud je požadována, když je zpráva skutečně vyřazena, ne když se stane způsobilou pro vyřazení.
4. Vyřazení vypršelé zprávy a generování sestavy vypršení platnosti, je-li požadováno, nejsou nikdy součástí jednotky práce aplikace, a to ani v případě, že zpráva byla naplánována pro vyřazení v důsledku volání MQGET fungujících v rámci jednotky práce.
5. Pokud je zpráva s téměř vypršenou platností načtena voláním MQGET v rámci pracovní jednotky a jednotka práce je následně vrácena zpět, může být zpráva způsobilá k vyřazení, než bude možné ji znovu načíst.
6. Pokud je zpráva s téměř vypršenou platností uzamknuta voláním MQGET s GMLK, může se stát, že zpráva bude vyřazena, než ji bude možné načíst voláním MQGET s GMMUC; kód příčiny RC2034 je vrácen při tomto následném volání MQGET, pokud k tomu dojde.
7. Když je načtena zpráva požadavku s dobou vypršení platnosti větší než nula, aplikace může při odesílání zprávy odpovědi provést jednu z následujících akcí:
   - Zkopírujte zbývající čas vypršení platnosti ze zprávy požadavku do zprávy odpovědi.

- Nastavte čas vypršení platnosti ve zprávě odpovědi na explicitní hodnotu větší než nula.
- Nastavte čas vypršení platnosti ve zprávě odpovědi na EIULIM.

Akce, která se má provést, závisí na návrhu sady aplikací. Výchozí akcí pro vkládání zpráv do fronty nedoručených zpráv by však mělo být zachování zbývajícího času vypršení platnosti zprávy a její snížení.

8. Zprávy spouštěče jsou vždy generovány s EIULIM.
9. Zpráva (obvykle v přenosové frontě), která má název MDFMT FMXQH, má druhý deskriptor zprávy v MQXQH. Proto má k sobě přidružena dvě pole MDEXP . V tomto případě je třeba poznamenat následující dodatečné body:
   - Když aplikace vloží zprávu do vzdálené fronty, správce front umístí zprávu na počátku do lokální přenosové fronty a před data zprávy aplikace umístí strukturu MQXQH. Správce front nastaví hodnoty dvou polí MDEXP tak, aby byly stejné jako hodnoty určené aplikací.

     Pokud aplikace vkládá zprávu přímo do lokální přenosové fronty, musí data zprávy již začínat strukturou MQXQH a název formátu musí být FMXQH (ale správce front to nevynucuje). V tomto případě nemusí aplikace nastavit hodnoty těchto dvou polí MDEXP tak, aby byly stejné. (Správce front nekontroluje, zda pole MDEXP v rámci MQXQH obsahuje platnou hodnotu, nebo zda jsou data zprávy dostatečně dlouhá, aby je bylo možné zahrnout.)
   - Když je zpráva s názvem MDFMT FMXQH načtena z fronty (bez ohledu na to, zda se jedná o normální nebo přenosovou frontu), správce front sníží obě tato pole MDEXP o dobu strávenou čekáním na frontu. Pokud data zprávy nejsou dostatečně dlouhá pro zahrnutí pole MDEXP do MQXQH, dojde k chybě.
   - Správce front používá pole MDEXP v samostatném deskriptoru zprávy (tj. nikoli v deskriptoru zprávy vloženém ve struktuře MQXQH) k testování, zda je zpráva vhodná pro vyřazení.
   - Pokud se počáteční hodnoty dvou polí MDEXP lišily, je možné, že čas MDEXP v odděleném deskriptoru zprávy, když je zpráva načtena, je větší než nula (takže zpráva není vhodná pro vyřazení), zatímco čas podle pole MDEXP v MQXQH uplynul. V tomto případě je pole MDEXP v MQXQH nastaveno na nulu.

Je rozpoznána následující speciální hodnota:

EIULIM
Neomezená životnost.

Zpráva má neomezenou dobu vypršení platnosti.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je EIULIM.

MDFB (10místné celé číslo se znaménkem)

Zpětná vazba nebo kód příčiny.

Používá se spolu se zprávou typu MTRPRT k označení povahy sestavy a je smysluplná pouze s tímto typem zprávy. Pole může obsahovat jednu z hodnot FB* nebo jednu z hodnot RC*. Kódy zpětné vazby jsou seskupeny takto:

FBNONE
Nebyla poskytnuta žádná zpětná vazba.

Doporučení FBSFST
Nejnižší hodnota pro zpětnou vazbu generovanou systémem.

Doporučení FBSLST
Nejvyšší hodnota pro zpětnou vazbu generovanou systémem.

Rozsah systémem generovaných kódů zpětné vazby FBSFST prostřednictvím FBSLST zahrnuje obecné kódy zpětné vazby uvedené dále v tomto oddíle (FB*) a také kódy příčiny (RC*), které se mohou vyskytnout, když zprávu nelze vložit do cílové fronty.

FBAFST
Nejnižší hodnota pro zpětnou vazbu generovanou aplikací.

FBALST
Nejvyšší hodnota pro zpětnou vazbu generovanou aplikací.

Aplikace, které generují zprávy sestav, by neměly používat kódy zpětné vazby v rozsahu systému (jiném než FBQUIT), pokud nechtějí simulovat zprávy sestav generované správcem front nebo agentem kanálu zpráv.

U volání MQPUT nebo MQPUT1 musí být zadaná hodnota buď FBNONE, nebo musí být v rozsahu systému nebo aplikace. Tato volba je zaškrtnuta bez ohledu na hodnotu parametru `MDMT`.

Obecné kódy zpětné vazby:

Doporučení FBCOA
Potvrzení o příjezdu do cílové fronty (viz ROCOA).

Doporučení FBCOD
Potvrzení o doručení na přijímající žádost (viz ROCOD).

Doporučení FBEXP
Platnost zprávy vypršela.

Zpráva byla zrušena, protože nebyla odebrána z cílové fronty před uplynutím doby platnosti.

FBPAN
Pozitivní oznámení akce (viz ROPAN).

FBNAN
Negativní oznámení akce (viz RONAN).

Doporučení FBQUIT
Aplikace by měla skončit.

Tuto možnost může použít plánovací program pracovní zátěže k řízení počtu instancí aplikačního programu, které jsou spuštěny. Odeslání zprávy MTRPRT s tímto kódem zpětné vazby do instance aplikačního programu označuje této instanci, že by měla zastavit zpracování. Dodržování této konvence je však záležitostí aplikace; správce front ji nevynucuje.

IMS-bridge feedback codes: Když most IMS obdrží nenulový kód IMS-OTMA, most IMS převede kód chyby z hexadecimálního na desetinný, přičte hodnotu FBIERR (300) a umístí výsledek do pole `MDFB` zprávy odpovědi. To má za následek, že kód zpětné vazby má hodnotu v rozsahu FBIFST (301) až FBILST (399), když došlo k chybě IMS-OTMA.

Most IMS může generovat následující kódy zpětné vazby:

FBDLZ
Délka dat nula.

V datech aplikace zprávy byla nulová délka segmentu.

FBDLN
Délka dat je záporná.

Délka segmentu byla v datech aplikace zprávy záporná.

FBDLTB
Délka dat je příliš velká.

Délka segmentu byla příliš velká v datech aplikace zprávy.

FBBUFO
Přetečení vyrovnávací paměti.

Hodnota jednoho z polí délky způsobí, že data přetečou vyrovnávací paměť zpráv.

FBLOB1
Délka v chybě o jeden.

Hodnota jednoho z polí délky byla jeden bajt příliš krátká.

Doporučení FBIIH
Struktura MQIIH není platná nebo chybí.

Pole MDFMT v deskriptoru MQMD určuje hodnotu FMIMS, zpráva však nezačíná platnou strukturou MQIIH.

Doporučení FBNAFI
ID uživatele není autorizováno pro použití v produktu IMS.

ID uživatele obsažené v deskriptoru zprávy MQMD nebo heslo obsažené v poli IIAUT ve struktuře MQIIH selhalo při ověření provedeném mostem IMS . V důsledku toho nebyla zpráva předána do adresáře IMS.

Doporučení FBIERR
Funkce IMSvrátila neočekávanou chybu.

Funkce IMSvrátila neočekávanou chybu. Další informace o chybě naleznete v protokolu chyb systému IBM MQ , na kterém je umístěn most IMS .

Doporučení FBIFST
Nejnižší hodnota pro zpětnou vazbu vygenerovanou produktem IMS.

IMS-generované kódy zpětné vazby zabírají rozsah FBIFST (300) až FBILST (399). Samotný smyslový kód IMS-OTMA je MDFB minus FBIERR.

Doporučení FBILST
Nejvyšší hodnota pro zpětnou vazbu generovanou produktem IMS.

CICS-bridge feedback codes: Produkt CICS bridgemůže generovat následující kódy zpětné vazby:

Doporučení FBCAAB
Aplikace byla nedokončena.

Aplikační program uvedený ve zprávě byl neukončen. Tento kód zpětné vazby se vyskytuje pouze v poli DLREA struktury MQDLH.

Doporučení FBCANS
Aplikaci nelze spustit.

Došlo k selhání EXEC CICS LINK pro aplikační program uvedený ve zprávě. Tento kód zpětné vazby se vyskytuje pouze v poli DLREA struktury MQDLH.

FBCBRF
Funkce CICS bridge byla nestandardně ukončena bez dokončení normálního zpracování chyb.

FBCCSE
Identifikátor znakové sady není platný.

Doporučení FBCIHE
Struktura záhlaví informací CICS chybí nebo není platná.

Doporučení FBCCAE
Délka CICS commarea není platná.

Doporučení FBCCIE
Identifikátor korelace není platný.

FBCDLQ
Fronta nedoručených zpráv není k dispozici.

Úloha CICS bridge nemohla zkopírovat odpověď na tento požadavek do fronty nedoručených zpráv. Požadavek byl odvolán.

Doporučení FBCENE
Kódování není platné.

Doporučení FBCINE
Produkt CICS bridge zjistil neočekávanou chybu.

Tento kód zpětné vazby se vyskytuje pouze v poli DLREA struktury MQDLH.

Doporučení FBCNTA
Identifikátor uživatele není autorizován nebo heslo není platné.

Tento kód zpětné vazby se vyskytuje pouze v poli DLREA struktury MQDLH.

FBCUBO
Jednotka práce byla odvolána.

Jednotka práce byla odvolána z jednoho z následujících důvodů:

- Při zpracování jiného požadavku v rámci stejné pracovní jednotky bylo zjištěno selhání.
- Během probíhající jednotky práce se vyskytlo nestandardního ukončení CICS .

FBCUWE
Pole řízení jednotky práce CIUOW není platné.

MQ kódy příčiny: Pro zprávy sestavy výjimek obsahuje MDFB kód příčiny MQ . Mezi možné kódy příčiny patří:

RC2051
(2051, X'803 ') Volání Put bylo pro frontu zablokováno.

RC2053
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

RC2035
(2035, X'7F3') Není autorizováno pro přístup.

RC2056
(2056, X'808 ') Není k dispozici místo na disku pro frontu.

RC2048
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

RC2031
(2031, X'7EF') Délka zprávy pro správce front je větší než maximum.

RC2030
(2030, X'7EE') Délka zprávy je větší než maximum pro frontu.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je FBNONE.

MDFMT (8bajtový znakový řetězec)

Název formátu dat zprávy.

Toto je jméno, které může odesilatel zprávy použít k označení povahy dat ve zprávě příjemci. Všechny znaky, které jsou obsaženy ve znakové sadě správce front, mohou být zadány pro název, ale doporučuje se, aby byl název omezen na následující:

- Velká písmena A až Z
- Číselné číslice 0 až 9

Pokud jsou použity jiné znaky, nemusí být možné přeložit název mezi znakovými sadami odesílajících a přijímajících správců front.

Název by měl být doplněn mezerami na délku pole, nebo znakem null použitým k ukončení názvu před koncem pole; s hodnotou null a dalšími znaky se zachází jako s mezerami. Neuvádějte název s úvodními nebo vloženými mezerami. Pro volání MQGET vrátí správce front název vyplněný mezerami na délku pole.

Správce front nekontroluje, zda název odpovídá dříve popsaným doporučením.

Názvy začínající "MQ" velkými, malými a smíšenými písmeny mají význam definovaný správcem front. Pro vlastní formáty byste neměli používat názvy začínající těmito písmeny. Vestavěné formáty správce front jsou:

FMNONE
Žádný název formátu.

Povaha dat není definována. To znamená, že data nelze převést, když je zpráva načtena z fronty pomocí volby GMCONV.

Je-li ve volání MQGET zadána hodnota GMCONV a znaková sada nebo kódování dat ve zprávě se liší od znakové sady nebo kódování dat v parametru **MSGDSC** , bude zpráva vrácena s následujícími kódy příčiny a dokončení (bez dalších chyb):

- Kód dokončení CCWARN a kód příčiny RC2110 , pokud jsou data FMNONE na začátku zprávy.
- Kód dokončení CCOK a kód příčiny RCNONE, pokud se data FMNONE nachází na konci zprávy (tj. před jednou nebo více strukturami záhlaví MQ). Struktury záhlaví MQ jsou v tomto případě převedeny na požadovanou znakovou sadu a kódování.

FMADMN
Zpráva požadavku/odpovědi příkazového serveru.

Zpráva je požadavek příkazového serveru nebo zpráva odpovědi v programovatelném formátu příkazu (PCF). Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV. Další informace o použití zpráv programovatelného formátu příkazů naleznete v tématu Použití programovatelných formátů příkazů.

FMCICS
Záhlaví informací CICS .

Data zprávy začínají záhlavím informací CICS MQCIH, za nímž následují data aplikace. Název formátu dat aplikace je dán polem CIFMT ve struktuře MQCIH.

FMCMD1
Napište 1 zprávu s odpovědí na příkaz.

Zpráva je zpráva odpovědi příkazového serveru MQSC obsahující počet objektů, kód dokončení a kód příčiny. Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMCMD2
Napište 2 zprávu s odpovědí na příkaz.

Zpráva je zpráva odpovědi příkazového serveru MQSC obsahující informace o požadovaném objektu (objektech). Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMDLH
Hlavička nedoručovací zprávy.

Data zprávy začínají záhlavím nedoručených zpráv MQDLH. Data z původní zprávy bezprostředně následují strukturu MQDLH. Název formátu původních dat zprávy je dán polem DLFMT ve struktuře MQDLH; podrobnosti o této struktuře viz “MQDLH (záhlaví nedoručených dopisů) na IBM i” na stránce 1061 . Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

Sestavy COA a COD nejsou generovány pro zprávy, které mají MDFMT FMDLH.

FMDH
Záhlaví distribučního seznamu.

Data zprávy začínají záhlavím distribučního seznamu MQDH; to zahrnuje pole záznamů MQOR a MQPMR. Za záhlavím rozdělovníku mohou následovat další data. Formát případných dalších dat je dán polem DHFMT ve struktuře MQDH; podrobnosti o této struktuře viz “MQDH (záhlaví distribuce) na IBM i” na stránce 1057 . Zprávy ve formátu FMDH lze převést, je-li ve volání MQGET zadána volba GMCONV.

FMEVNT
Zpráva události.

Zpráva je zpráva události produktu MQ , která hlásí událost, k níž došlo. Zprávy událostí mají stejnou strukturu jako programovatelné příkazy; další informace o této struktuře viz Struktury

příkazů a odezev. Chcete-li získat informace o událostech, prohlédněte si téma Monitorování událostí.

Zprávy událostí Version-1 lze převést, pokud je ve volání MQGET určena volba GMCONV.

FMIMS
Záhlaví informací IMS .

Data zprávy začínají záhlaví informací IMS MQIIH, za nímž následují data aplikace. Název formátu dat aplikace je dán polem *IIFMT* ve struktuře MQIIH. Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMIMVS
Řetězec proměnné IMS .

Zpráva je řetězec proměnné IMS , což je řetězec ve tvaru `llzzccc`, kde:

`ll`
je 2bajtové pole délky určující celkovou délku položky řetězce proměnné IMS . Tato délka se rovná délce `ll` (2 bajty) plus délce `zz` (2 bajty) plus délce samotného znakového řetězce. `ll` je dvoubajtové binární celé číslo v kódování uvedeném v poli `MDENC` .

`zz`
je 2bajtové pole obsahující příznaky, které jsou důležité pro IMS. `zz` je bajtový řetězec skládající se ze dvou 1bajtových bitových řetězcových polí a je přenášen beze změny z odesilatele na příjemce (to znamená, že `zz` nepodléhá žádnému převodu).

`ccc`
je znakový řetězec s proměnnou délkou obsahující `ll-4` znaků. `ccc` je ve znakové sadě uvedené v poli `MDCSI` .

Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMMDE
Rozšíření deskriptoru zpráv.

Data zprávy začínají příponou MQMDE deskriptoru zprávy a jsou volitelně následována dalšími daty (obvykle daty zprávy aplikace). Název formátu, znaková sada a kódování dat, která následují za MQMDE, jsou dány poli `MEFMT`, `MECSI`a `MEENC` v MQMDE. Podrobnosti o této struktuře viz “MQMDE (rozšíření deskriptoru zpráv) na IBM i” na stránce 1148 . Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

Prostředek FMPCF
Uživatelsky definovaná zpráva v programovatelném formátu příkazů (PCF).

Zpráva je uživatelem definovaná zpráva, která je v souladu se strukturou zprávy PCF (programmable command format). Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV. Další informace o použití zpráv programovatelného formátu příkazů naleznete v tématu Použití programovatelných formátů příkazů .

FMRMH
Záhlaví referenční zprávy.

Data zprávy začínají záhlavím referenční zprávy MQRMH a jsou volitelně následována dalšími daty. Název formátu, znaková sada a kódování dat jsou dány poli `RMFMT`, `RMCSI`a `RMENC` v MQRMH. Podrobnosti o této struktuře viz “MQRMH (záhlaví referenční zprávy) na IBM i” na stránce 1194 . Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMRFH
Pravidla a formátování záhlaví.

Data zprávy začínají pravidly a formátováním záhlaví MQRFH a jsou volitelně následována dalšími daty. Název formátu, znaková sada a kódování dat (pokud existují) jsou dány poli `RFFMT`, `RFCSI`a `RFENC` v MQRFH. Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMRFH2
Pravidla a formátování záhlaví verze 2.

Data zprávy začínají pravidly version-2 a formátováním záhlaví MQRFH2a jsou volitelně následována dalšími daty. Název formátu, znaková sada a kódování volitelných dat (jsou-li nějaká) jsou dány poli RF2FMT, RF2CSIa RF2ENC v MQRFH2. Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMSTR
Zpráva se skládá výhradně ze znaků.

Data zprávy aplikace mohou být buď SBCS řetězec (jednobajtová znaková sada), nebo DBCS řetězec (dvoubajtová znaková sada). Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMTM
Zpráva spouštěče.

Zpráva je zpráva spouštěče popsaná strukturou MQTM. Podrobnosti o této struktuře viz “MQTM-zpráva spouštěče” na stránce 1229 . Zprávy tohoto formátu lze převést, pokud je ve volání MQGET zadána volba GMCONV.

FMWIH
Záhlaví informací o práci.

Data zprávy začínají záhlavím MQWIH, za nímž následují data aplikace. Název formátu dat aplikace je dán polem WIFMT ve struktuře MQWIH.

FMXQH
Záhlaví přenosové fronty.

Data zprávy začínají záhlavím přenosové fronty MQXQH. Data z původní zprávy bezprostředně následují strukturu MQXQH. Název formátu původních dat zprávy je dán polem MDFMT ve struktuře MQMD, která je součástí záhlaví přenosové fronty MQXQH. Podrobnosti o této struktuře viz “MQXQH (záhlaví přenosové fronty) na IBM i” na stránce 1239 .

Sestavy COA a COD nejsou generovány pro zprávy, které mají MDFMT FMXQH.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

MDGID (24bajtový bitový řetězec)

Identifikátor skupiny.

Jedná se o bajtový řetězec, který se používá k identifikaci konkrétní skupiny zpráv nebo logické zprávy, ke které fyzická zpráva patří. MDGID se také používá, pokud je pro zprávu povolena segmentace. Ve všech těchto případech má MDGID nenulovou hodnotu a v poli MDMFL je nastaven jeden nebo více následujících příznaků:

- MFMIG
- MFLMIG
- MFSEG
- MFLSEG-rozšíření
- MFSEGA

Pokud není nastaven žádný z těchto příznaků, MDGID má speciální hodnotu null GINONE.

Toto pole nemusí být nastaveno aplikací ve volání MQPUT nebo MQGET, pokud:

- Ve volání MQPUT je zadána hodnota PMLOGO.
- Ve volání MQGET není zadán parametr MOGRPI.

Zvažte použití těchto volání pro zprávy, které nejsou zprávami sestavy. Pokud však aplikace vyžaduje větší kontrolu nebo volání je MQPUT1, musí aplikace zajistit, aby byl parametr MDGID nastaven na odpovídající hodnotu.

Skupiny zpráv a segmenty lze správně zpracovat pouze v případě, že je identifikátor skupiny jedinečný. Z tohoto důvodu by aplikace neměly generovat své vlastní identifikátory skupin; místo toho by aplikace měly provést jednu z následujících možností:

- Je-li uveden PMLOGO, správce front automaticky vygeneruje jedinečný identifikátor skupiny pro první zprávu ve skupině nebo segmentu logické zprávy a použije tento identifikátor skupiny pro zbývající zprávy ve skupině nebo segmentech logické zprávy, takže aplikace nemusí provádět žádné speciální akce. Zvažte použití tohoto postupu.
- Není-li zadána hodnota PMLOGO, měla by aplikace požádat správce front o vygenerování identifikátoru skupiny nastavením parametru MDGID na hodnotu GINONE při prvním volání MQPUT nebo MQPUT1 pro zprávu ve skupině nebo segmentu logické zprávy. Identifikátor skupiny vrácený správcem front na výstupu z tohoto volání by měl být poté použit pro zbývající zprávy ve skupině nebo segmentech logické zprávy. Pokud skupina zpráv obsahuje segmentované zprávy, musí být pro všechny segmenty a zprávy ve skupině použit stejný identifikátor skupiny.

  Není-li zadána hodnota PMLOGO, lze zprávy ve skupinách a segmentech logických zpráv vkládat v libovolném pořadí (například v opačném pořadí), avšak identifikátor skupiny musí být přidělen prvním voláním MQPUT nebo MQPUT1 , které je vydáno pro kteroukoli z těchto zpráv.

Na vstupu volání MQPUT a MQPUT1 používá správce front hodnotu uvedenou v části PMOPT. Ve výstupu volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána spolu se zprávou, pokud je otevřený objekt jedinou frontou a nikoli distribučním seznamem, ale ponechá jej beze změny, pokud je otevřený objekt distribučním seznamem. V druhém případě, pokud aplikace potřebuje znát vygenerované identifikátory skupin, musí aplikace poskytnout záznamy MQPMR obsahující pole PRGID .

Při vstupu do volání MQGET používá správce front hodnotu popsanou v části Tabulka 1. Ve výstupu volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Je definována následující speciální hodnota:

GINONE
Není uveden žádný identifikátor skupiny.

Hodnota je binární nula pro délku pole. Jedná se o hodnotu, která se používá pro zprávy, které nejsou ve skupinách, nejsou segmenty logických zpráv a pro které není segmentace povolena.

Délka tohoto pole je dána hodnotou LNGID. Počáteční hodnota tohoto pole je GINONE. Toto pole se ignoruje, pokud je MDVER menší než MDVER2.

MDMFL (10místné celé číslo se znaménkem)

Příznaky zprávy.

Jedná se o příznaky, které určují atributy zprávy nebo řídí její zpracování. Vlajky jsou rozděleny do následujících kategorií:

- Příznak segmentace
- Příznaky stavu

Ty jsou popsány na oplátku.

Příznaky segmentace: Pokud je zpráva pro frontu příliš velká, pokus o vložení zprávy do fronty obvykle selže. Segmentace je technika, při které správce front nebo aplikace rozdělí zprávu na menší části nazývané segmenty a umístí jednotlivé segmenty do fronty jako samostatnou fyzickou zprávu. Aplikace, která načte zprávu, může buď načíst segmenty jeden po druhém, nebo požádat správce front o opětovné sestavení segmentů do jedné zprávy, která je vrácena voláním MQGET. Toho je dosaženo zadáním volby GMCMPM pro volání MQGET a dodáním vyrovnávací paměti, která je dostatečně velká, aby pojmula celou zprávu. (Podrobnosti o volbě GMCMPM naleznete v části “MQGMO (Volby

Get-message) na systému IBM i" na stránce 1073 .) K segmentaci zprávy může dojít v odesílajícím správci front, v intermediačním správci front nebo v cílovém správci front.

Chcete-li řídit segmentaci zprávy, můžete určit jednu z následujících možností:

MFSEGI
Segmentace byla zablokována.

Tato volba brání tomu, aby správce front rozdělil zprávu na segmenty. Je-li uvedeno pro zprávu, která je již segmentem, tato volba zabrání rozdělení segmentu na menší segmenty.

Hodnota tohoto příznaku je binární nula. Toto nastavení je výchozí.

MFSEGA
Segmentace je povolena.

Tato volba umožňuje správci front rozdělit zprávu na segmenty. Je-li uvedeno pro zprávu, která je již segmentem, tato volba umožňuje rozdělení segmentu na menší segmenty. MFSEGA lze nastavit bez nastavení MFSEG nebo MFLSEG.

Když správce front segmentuje zprávu, zapne správce front příznak MFSEG v kopii deskriptoru MQMD, který je odeslán s každým segmentem, ale nezmění nastavení těchto příznaků v deskriptoru MQMD poskytnutém aplikací ve volání MQPUT nebo MQPUT1 . Pro poslední segment v logické zprávě správce front také zapne příznak MFLSEG v deskriptoru MQMD, který je odeslán spolu se segmentem.

Poznámka: Při vložení zpráv s MFSEGA, ale bez PMLOGO, je třeba věnovat pozornost. Pokud je zpráva:

- Nejedná se o segment a
- Není ve skupině a
- Nepředávají se,

aplikace musí pamatovat na resetování pole MDGID na hodnotu GINONE před každým voláním MQPUT nebo MQPUT1 , aby pro každou zprávu správce front vygeneroval jedinečný identifikátor skupiny. Pokud se tak nestane, nesouvisející zprávy mohou neúmyslně skončit se stejným identifikátorem skupiny, což může vést k následnému chybnému zpracování. Další informace o tom, kdy musí být pole MDGID resetováno, naleznete v popisech pole MDGID a volby PMLOGO.

Správce front rozdělí zprávy do segmentů podle potřeby, aby se zajistilo, že se segmenty (plus data záhlaví, která mohou být vyžadována) vejde do fronty. Existuje však dolní limit velikosti segmentu generovaného správcem front a pouze poslední segment vytvořený ze zprávy může být menší než tento limit. (Dolní limit velikosti segmentu generovaného aplikací je jeden bajt.) Segmenty generované správcem front mohou mít nestejnou délku. Správce front zpracuje zprávu následujícím způsobem:

- Uživatelem definované formáty jsou rozděleny na hranice, které jsou násobky 16 bajtů. To znamená, že správce front nebude generovat segmenty, které jsou menší než 16 bajtů (jiné než poslední segment).
- Vestavěné formáty jiné než FMSTR jsou rozděleny v bodech odpovídajících povaze přítomných dat. Správce front však nikdy nerozdělí zprávu uprostřed struktury záhlaví produktu MQ . To znamená, že segment obsahující jedinou strukturu záhlaví produktu MQ nemůže správce front dále rozdělit, a v důsledku toho je minimální možná velikost segmentu pro tuto zprávu větší než 16 bajtů.

  Druhý nebo novější segment vygenerovaný správcem front začíná jedním z následujících:

  – Struktura záhlaví MQ
  – Spuštění dat zprávy aplikace
  – Částečný průchod daty zprávy aplikace
- FMSTR je rozdělen bez ohledu na povahu přítomných dat (SBCS, DBCS nebo smíšený SBCS/DBCS). Je-li řetězec DBCS nebo smíšený SBCS/DBCS, může to vést k segmentům, které nelze

převést z jedné znakové sady na jinou. Správce front nikdy nerozdělí zprávy FMSTR na segmenty, které jsou menší než 16 bajtů (jiné než poslední segment).

- Pole MDFMT, MDCSIa MDENC v deskriptoru MQMD jednotlivých segmentů jsou správcem front nastavena tak, aby správně popisovala data přítomná na začátku segmentu. Název formátu bude buď název vestavěného formátu, nebo název formátu definovaného uživatelem.
- Pole MDREP v deskriptoru MQMD segmentů s hodnotou MD0FF větší než nula se mění takto:
  - Pro každý typ sestavy platí, že pokud je volba sestavy RO* D, ale segment nemůže obsahovat žádné z prvních 100 bajtů uživatelských dat (tj. dat následujících po případných strukturách záhlaví MQ), volba sestavy se změní na RO*.

Správce front dodržuje předchozí pravidla, ale jinak nepředvídatelně rozděluje zprávy; neprovádějte předpoklady o tom, kde je zpráva rozdělena.

Pro trvalé zprávy může správce front provádět segmentaci pouze v rámci pracovní jednotky:

- Pokud volání MQPUT nebo MQPUT1 pracuje v rámci uživatelem definované jednotky práce, použije se tato jednotka práce. Dojde-li k selhání volání v rámci procesu segmentace, správce front odebere všechny segmenty, které byly v důsledku selhání volání umístěny do fronty. Selhání však nebrání úspěšnému potvrzení transakce.
- Pokud volání pracuje mimo uživatelem definovanou jednotku práce a neexistuje žádná uživatelem definovaná jednotka práce, správce front vytvoří jednotku práce pouze po dobu trvání volání. Je-li volání úspěšné, správce front potvrdí transakci automaticky (aplikace to nemusí provést). Pokud volání selže, správce front odvolá transakci.
- Pokud volání pracuje mimo uživatelem definovanou jednotku práce, ale existuje uživatelem definovaná jednotka práce, nemůže správce front provést segmentaci. Pokud zpráva nevyžaduje segmentaci, volání může být stále úspěšné. Pokud však zpráva vyžaduje segmentaci, volání selže s kódem příčiny RC2255.

Pro přechodné zprávy nevyžaduje správce front, aby byla k dispozici jednotka práce, aby bylo možné provést segmentaci.

Zvláštní pozornost je třeba věnovat převodu dat zpráv, které mohou být segmentovány:

- Pokud převod dat provádí pouze přijímající aplikace ve volání MQGET a aplikace určuje volbu GMCMPM, uživatelské proceduře pro převod dat bude předána úplná zpráva pro uživatelskou proceduru, která má být převedena, a skutečnost, že zpráva byla segmentována, nebude pro uživatelskou proceduru zřejmá.
- Pokud přijímající aplikace načte jeden segment v daném okamžiku, bude vyvolána uživatelská procedura převodu dat, aby převedla jeden segment v daném okamžiku. Výstup tedy musí být schopen převést data v segmentu nezávisle na datech v kterémkoli z ostatních segmentů.

  Je-li povaha dat ve zprávě taková, že libovolná segmentace dat na 16bajtových hranicích může mít za následek segmenty, které nelze převést uživatelskou procedurou, nebo formát je FMSTR a znaková sada je DBCS nebo smíšená SBCS/DBCS, měla by odesílající aplikace sama vytvořit a vložit segmenty, přičemž uvede MFSEGI pro potlačení další segmentace. Tímto způsobem může odesílající aplikace zajistit, aby každý segment obsahoval dostatečné informace, které umožní uživatelské proceduře pro převod dat úspěšně převést segment.
- Je-li pro agenta MCA (sender channel agent) zadán převod odesilatele, agent MCA převede pouze zprávy, které nejsou segmenty logických zpráv; agent MCA se nikdy nepokusí převést zprávy, které jsou segmenty.

Tento příznak je vstupním příznakem volání MQPUT a MQPUT1 a výstupním příznakem volání MQGET. Při druhém volání správce front také zopakuje hodnotu příznaku do pole GMSEG v MQGMO.

Počáteční hodnota tohoto příznaku je MFSEGI.

Příznaky stavu: Jedná se o příznaky, které označují, zda fyzická zpráva patří do skupiny zpráv, zda se jedná o segment logické zprávy, obojí, nebo obojí. Ve volání MQPUT nebo MQPUT1 lze zadat jednu nebo více následujících možností nebo je lze vrátit voláním MQGET:

MFMIG
Zpráva je členem skupiny.

MFLMIG
Zpráva je poslední logická zpráva ve skupině.

Je-li tento příznak nastaven, správce front zapne MFMIG v kopii MQMD, která je odeslána se zprávou, ale nezmění nastavení těchto příznaků v MQMD poskytnutém aplikací ve volání MQPUT nebo MQPUT1 .

Je platné, aby skupina sestával pouze z jedné logické zprávy. Pokud se jedná o tento případ, je nastavena hodnota MFLMIG, ale pole MDSEQ má hodnotu jedna.

MFSEG
Zpráva je segmentem logické zprávy.

Je-li MFSEG zadán bez MFLSEG, délka dat zprávy aplikace v segmentu (s výjimkou délky případných struktur záhlaví MQ) musí být alespoň jedna. Je-li délka nula, volání MQPUT nebo MQPUT1 selže s kódem příčiny RC2253.

MFLSEG-rozšíření
Zpráva je posledním segmentem logické zprávy.

Je-li tento příznak nastaven, správce front zapne MFSEG v kopii MQMD, která je odeslána se zprávou, ale nezmění nastavení těchto příznaků v MQMD poskytnutém aplikací ve volání MQPUT nebo MQPUT1 .

Je platné, aby logická zpráva sestával pouze z jednoho segmentu. Pokud se jedná o tento případ, je nastavena hodnota MFLSEG, ale pole MDOFF má hodnotu nula.

Když je uvedeno MFLSEG, je přípustné, aby délka dat zprávy aplikace v segmentu (s výjimkou délek libovolných struktur záhlaví, které mohou být přítomny) byla nula.

Aplikace musí zajistit správné nastavení těchto příznaků při vkládání zpráv. Pokud je zadána hodnota PMLOGO nebo byla zadána v předchozím volání MQPUT pro manipulátor fronty, musí být nastavení příznaků konzistentní s informacemi o skupině a segmentu uchovávanými správcem front pro manipulátor fronty. Následující podmínky platí pro následná volání MQPUT pro manipulátor fronty při zadání PMLOGO:

- Pokud neexistuje žádná aktuální skupina nebo logická zpráva, všechny tyto příznaky (a jejich kombinace) jsou platné.
- Jakmile je uvedeno MFMIG, musí zůstat zapnuté, dokud není uvedeno MFLMIG. Volání selže s kódem příčiny RC2241 , pokud není tato podmínka splněna.
- Jakmile je uvedeno MFSEG, musí zůstat zapnuté, dokud není uvedeno MFLSEG. Volání selže s kódem příčiny RC2242 , pokud není tato podmínka splněna.
- Jakmile je MFSEG uvedeno bez MFMIG, MFMIG musí zůstat vypnutý, dokud není uvedeno MFLSEG. Volání selže s kódem příčiny RC2242 , pokud není tato podmínka splněna.

Tabulka 1 zobrazuje platné kombinace příznaků a hodnoty použité pro různá pole.

Jedná se o vstupní příznaky pro volání MQPUT a MQPUT1 a výstupní příznaky pro volání MQGET. Při druhém volání správce front také zopakuje hodnoty příznaků do polí GMGST a GMSST v produktu MQGMO.

Výchozí příznaky: Pomocí následujícího příkazu lze určit, že zpráva má výchozí atributy:

MFNONE
Žádné příznaky zprávy (výchozí atributy zprávy).

To brání segmentaci a označuje, že zpráva není ve skupině a není segmentem logické zprávy. MFNONE je definován jako pomocná programová dokumentace. Není zamýšleno, aby byl tento příznak použit s jinými, ale protože jeho hodnota je nula, nelze takové použití zjistit.

Pole MDMFL je rozděleno na dílčí pole; podrobnosti viz "Volby sestavy a příznaky zpráv na IBM i" na stránce 1425.

Počáteční hodnota tohoto pole je MFNONE. Toto pole se ignoruje, pokud je MDVER menší než MDVER2.

MDMID (24bajtový bitový řetězec)

Identifikátor zprávy.

Jedná se o bajtový řetězec, který se používá k odlišení jedné zprávy od druhé. Obecně platí, že žádné dvě zprávy by neměly mít stejný identifikátor zprávy, ačkoli to správce front nepovoluje. Identifikátor zprávy je trvalou vlastností zprávy a přetrvává po restartování správce front. Vzhledem k tomu, že identifikátor zprávy je bajtový řetězec a nikoli znakový řetězec, není identifikátor zprávy převeden mezi znakovými sadami, pokud zpráva přechází z jednoho správce front do jiného.

Pro volání MQPUT a MQPUT1 platí, že pokud aplikace zadá hodnotu MINONE nebo PMNMID, správce front při vložení zprávy vygeneruje jedinečný identifikátor zprávy a umístí jej do deskriptoru zprávy odeslaného se zprávou. Správce front také vrací tento identifikátor zprávy v deskriptoru zprávy, který náleží odesílající aplikaci. Aplikace může tuto hodnotu použít k zaznamenávání informací o konkrétních zprávách a k odpovídání na dotazy z jiných částí aplikace.

MDMID generovaný správcem front se skládá z 4bajtového identifikátoru produktu (AMQ¬ nebo CSQ¬ v ASCII nebo EBCDIC, kde ¬ představuje jeden prázdný znak), následovaný implementací jedinečného řetězce specifickou pro produkt. V produktu IBM MQ obsahuje prvních 12 znaků názvu správce front a hodnotu odvozenou z hodin systému. Všichni správci front, kteří mohou komunikovat, proto musí mít názvy, které se liší v prvních 12 znacích, aby bylo zajištěno, že identifikátory zpráv jsou jedinečné. Schopnost generovat jedinečný řetězec také závisí na tom, aby systémové hodiny nebyly měněny dozadu. Chcete-li vyloučit možnost, aby identifikátor zprávy generovaný správcem front duplikoval identifikátor generovaný aplikací, měla by se aplikace vyvarovat generování identifikátorů s počátečními znaky v rozsahu A až I v kódu ASCII nebo EBCDIC (X'41 'až X'49' a X'C1'až X'C9'). Aplikaci však není bráněno v generování identifikátorů s počátečními znaky v těchto rozsazích.

Pokud je zpráva vkládána do tématu, správce front generuje jedinečné identifikátory zpráv podle potřeby pro každou publikovanou zprávu. Pokud je identifikátor PMNMID určen aplikací, správce front vygeneruje jedinečný identifikátor zprávy, který se má vrátit při výstupu. Pokud aplikace zadá hodnotu MINONE, hodnota pole MDMID v deskriptoru MQMD se při návratu z volání nezmění.

Další podrobnosti o zachovaných publikacích naleznete v popisu tabulky PMRET v části PMOPT .

Pokud je zpráva vkládána do distribučního seznamu, správce front vygeneruje jedinečné identifikátory zpráv podle potřeby, ale hodnota pole MDMID v MQMD se při návratu z volání nezmění, a to i v případě, že byly zadány hodnoty MINONE nebo PMNMID. Pokud aplikace potřebuje znát identifikátory zpráv generované správcem front, musí poskytnout záznamy MQPMR obsahující pole PRMID .

Odesílající aplikace může také určit konkrétní hodnotu pro identifikátor zprávy jinou než MINONE; tím se zastaví generování jedinečného identifikátoru zprávy správcem front. Aplikace, která předává zprávu, může tento prostředek použít k šíření identifikátoru zprávy původní zprávy.

Správce front sám toto pole nepoužívá, kromě následujících:

- Generovat jedinečnou hodnotu, je-li požadována, jak bylo popsáno dříve
- Dodat hodnotu do aplikace, která vydá požadavek na získání pro zprávu
- Zkopírujte hodnotu do pole MDCID libovolné zprávy sestavy, kterou generuje o této zprávě (v závislosti na volbách MDREP).

Když správce front nebo agent kanálu zpráv vygeneruje zprávu sestavy, nastaví pole MDMID způsobem uvedeným v poli MDREP původní zprávy, buď RONMI, nebo ROPMI. Aplikace, které generují zprávy sestav, by to měly také provést.

Pro volání MQGET je MDMID jedním z pěti polí, které lze použít k výběru konkrétní zprávy, která má být načtena z fronty. Volání MQGET obvykle vrací další zprávu ve frontě, ale pokud je vyžadována konkrétní zpráva, lze ji získat zadáním jednoho nebo více pěti kritérií výběru v libovolné kombinaci; tato pole jsou:

- MDMID
- MDCID

- MDGID
- MDSEQ
- MDOFF

Aplikace nastaví jedno nebo více z těchto polí na požadované hodnoty a poté nastaví odpovídající volby shody MO* v poli GMMO v produktu MQGMO, aby označila, že by tato pole měla být použita jako kritéria výběru. Kandidáty na načtení jsou pouze zprávy, které mají uvedené hodnoty v těchto polích. Výchozí hodnotou pro pole GMMO (pokud není aplikací změněno) je shoda jak s identifikátorem zprávy, tak s identifikátorem korelace.

Za normálních okolností je vrácená zpráva první zprávou ve frontě, která splňuje kritéria výběru. Ale pokud je uvedeno GMBRWN, vrácená zpráva je další zpráva, která splňuje kritéria výběru; skenování této zprávy začíná zprávou následující aktuální pozici kurzoru.

Poznámka: Fronta je skenována sekvenčně pro zprávu, která splňuje kritéria výběru, takže doba načtení bude pomalejší, než když nejsou uvedena žádná kritéria výběru, zejména pokud musí být skenováno mnoho zpráv před nalezením vhodné zprávy.

Další informace o tom, jak se kritéria výběru používají v různých situacích, viz Tabulka 1 .

Uvedení MINONE jako identifikátoru zprávy má stejný efekt jako neuvedení MOMSGI, to znamená, že jakýkoli identifikátor zprávy se bude shodovat.

Toto pole je ignorováno, pokud je v parametru **GMO** ve volání MQGET zadána volba GMMUC.

Při návratu z volání MQGET je pole MDMID nastaveno na identifikátor vrácené zprávy (pokud existuje).

Lze použít následující speciální hodnotu:

DOLNÉ
Není uveden žádný identifikátor zprávy.

Hodnota je binární nula pro délku pole.

Toto je vstupní/výstupní pole pro volání MQGET, MQPUT a MQPUT1 . Délka tohoto pole je dána hodnotou LNMID. Počáteční hodnota tohoto pole je MINONE.

MDMT (10místné celé číslo se znaménkem)

Typ zprávy.

Označuje typ zprávy. Typy zpráv jsou seskupeny takto:

MTSFST
Nejnižší hodnota pro typy zpráv definované systémem.

MTSLST
Nejvyšší hodnota pro typy zpráv definované systémem.

Následující hodnoty jsou momentálně definovány v rozsahu systému:

MTDGRM
Zpráva nevyžadující odpověď.

Zpráva je taková, která nevyžaduje odpověď.

MTRQST
Zpráva vyžadující odpověď.

Zpráva je zpráva, která vyžaduje odpověď.

Název fronty, do které se má odeslat odpověď, musí být uveden v poli MDRQ . Pole MDREP označuje, jak mají být nastaveny hodnoty MDMID a MDCID odpovědi.

MTRPLY
Odpovězte na dřívější zprávu požadavku.

Zpráva je odpovědí na dřívější zprávu požadavku (MTRQST). Zpráva by měla být odeslána do fronty označené polem MDRQ zprávy požadavku. Pole MDREP požadavku by mělo být použito k řízení nastavení MDMID a MDCID odpovědi.

Poznámka: Správce front nevynucuje vztah požadavek-odezva. Jedná se o odpovědnost aplikace.

MTRPRT
Zpráva sestavy.

Zpráva vykazuje očekávaný nebo neočekávaný výskyt, obvykle související s jinou zprávou (například byla přijata zpráva požadavku, která obsahovala neplatná data). Zpráva by měla být odeslána do fronty označené polem MDRQ deskriptoru původní zprávy. Pole MDFB by mělo být nastaveno tak, aby označovala povahu sestavy. Pole MDREP původní zprávy lze použít k řízení způsobu nastavení MDMID a MDCID zprávy sestavy.

Zprávy sestav generované správcem front nebo agentem kanálu zpráv jsou vždy odesílány do fronty MDRQ s nastavenými poli MDFB a MDCID , jak bylo popsáno výše.

Další hodnoty v rámci rozsahu systému mohou být definovány v budoucích verzích rozhraní MQI a jsou přijímány voláními MQPUT a MQPUT1 bez chyb.

Lze také použít hodnoty definované aplikací. Musí být v tomto rozsahu:

MTAFST
Nejnižší hodnota pro typy zpráv definované aplikací.

MTALST
Nejvyšší hodnota pro typy zpráv definované aplikací.

Pro volání MQPUT a MQPUT1 musí být hodnota MDMT buď v rozsahu definovaném systémem, nebo v rozsahu definovaném aplikací; pokud není, volání selže s kódem příčiny RC2029.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MTDGRM.

MDOFF (10místné celé číslo se znaménkem)

Posunutí dat ve fyzické zprávě od začátku logické zprávy.

Jedná se o posun v bajtech dat ve fyzické zprávě od začátku logické zprávy, jejíž část tvoří data. Tato data se nazývají *segment*. Posun je v rozsahu 0 až 999 999 999 999. Fyzická zpráva, která není segmentem logické zprávy, má offset nula.

Toto pole nemusí být nastaveno aplikací ve volání MQPUT nebo MQGET, pokud:

- Ve volání MQPUT je zadána hodnota PMLOGO.
- Ve volání MQGET není uveden MOOFFS.

Toto jsou doporučené způsoby použití těchto volání pro zprávy, které nejsou zprávami sestavy. Pokud však aplikace nesplňuje tyto podmínky nebo volání je MQPUT1, musí aplikace zajistit, aby byl parametr MDOFF nastaven na odpovídající hodnotu.

Při vstupu pro volání MQPUT a MQPUT1 používá správce front hodnotu popsanou v tabulce 1. Ve výstupu volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána se zprávou.

Pro zprávu sestavy vykazující segment logické zprávy se pole MDOLN (za předpokladu, že není OLUNDF) používá k aktualizaci offsetu v informacích segmentu uchovávaných správcem front.

Při vstupu do volání MQGET používá správce front hodnotu popsanou v části Tabulka 1. Ve výstupu volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Počáteční hodnota tohoto pole je nula. Toto pole se ignoruje, pokud je MDVER menší než MDVER2.

MDOLN (10místné celé číslo se znaménkem)

Délka původní zprávy.

Toto pole má význam pouze pro zprávy sestavy, které jsou segmenty. Uvádí délku segmentu zprávy, ke kterému se zpráva sestavy vztahuje; neuvádí délku logické zprávy, jejíž část tvoří segment, ani délku dat ve zprávě sestavy.

Poznámka: Při generování zprávy sestavy pro zprávu, která je segmentem, zkopírují správce front a agent kanálu zpráv do MQMD pro zprávu sestavy pole MDGID, MDSEQ, MDOFFa *MDMFL*z původní zprávy. Výsledkem je, že zpráva sestavy je také segmentem. Aplikacím, které generují zprávy sestavy, se doporučuje provést totéž a zajistit správné nastavení pole MDOLN .

Je definována následující speciální hodnota:

OLUNDF
Původní délka zprávy není definována.

MDOLN je vstupní pole pro volání MQPUT a MQPUT1 , ale hodnota poskytnutá aplikací je přijata pouze za určitých okolností:

- Pokud je vkládaná zpráva segmentem a je také zprávou sestavy, správce front přijme zadanou hodnotu. Hodnota musí být:
  - Větší než nula, pokud segment není posledním segmentem
  - Není menší než nula, pokud je segment posledním segmentem
  - Ne méně než délka dat přítomných ve zprávě

  Nejsou-li tyto podmínky splněny, volání selže s kódem příčiny RC2252.
- Pokud je vkládaná zpráva segmentem, ale ne zprávou sestavy, správce front pole ignoruje a místo toho použije délku dat zprávy aplikace.
- Ve všech ostatních případech správce front ignoruje pole a místo toho použije hodnotu OLUNDF.

Toto je výstupní pole pro volání MQGET.

Počáteční hodnota tohoto pole je OLUNDF. Toto pole se ignoruje, pokud je MDVER menší než MDVER2.

MDPAN (28bajtový znakový řetězec)

Název aplikace, která vložila zprávu.

Toto je část *původního kontextu* zprávy. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Formát parametru MDPAN závisí na hodnotě parametru MDPAT.

Je-li toto pole nastaveno správcem front (tj. pro všechny volby kromě PMSETA), je nastaveno na hodnotu, která je určena prostředím:

- z/OS V systému z/OSpoužívá správce front:
  - V případě dávky z/OS se jedná o osmiznakový název úlohy z karty JES JOB.
  - V případě TSO se jedná o 7znakový identifikátor uživatele TSO.
  - V případě systému CICSse jedná o osmiznakovou aplikaci následovanou čtyřznakovým identifikátorem tranid.
  - V případě systému IMSse jedná o 8znakový identifikátor systému IMS následovaný 8znakovým názvem PSB.
  - V případě XCF se jedná o 8znakový název skupiny XCF následovaný 16znakovým názvem člena XCF.
  - V případě zprávy generované správcem front se jedná o prvních 28 znaků názvu správce front.
  - V případě distribuovaného řazení do fronty bez produktu CICSse jedná o osmiznakový název úlohy inicializátoru kanálu následovaný osmiznakovým názvem modulu, který se vkládá do fronty nedoručených zpráv, za nímž následuje osmiznakový identifikátor úlohy.

– Pro zpracování vazeb jazyka MQSeries Java s IBM MQ for z/OS 8znakovým názvem úlohy adresního prostoru vytvořeného pro prostředí z/OS UNIX System Services . Obvykle se jedná o identifikátor uživatele TSO s připojeným jedním číselným znakem.

Každý název nebo názvy jsou vyplněny vpravo mezerami, stejně jako jakákoli mezera ve zbytku pole. Pokud existuje více než jeden název, není mezi nimi žádný oddělovač.

- Windows V systémech PC DOS a Windows používá správce front:
  – V případě aplikace CICS se jedná o název transakce CICS .
  – V případě aplikací jiných nežCICS se jedná o 28 znaků zcela vpravo od úplného názvu spustitelného souboru.
- IBM i V systému IBM ipoužívá správce front úplný název úlohy.
- Linux AIX V systému AIX and Linuxpoužívá správce front:
  – V případě aplikace CICS se jedná o název transakce CICS .
  – V případě aplikací, které nejsou typuCICS , 14 znaků zcela vpravo od úplného názvu spustitelného souboru, pokud je k dispozici pro správce front, a mezery jinak (například v systému AIX).
- V systému VSE/ESApoužívá správce front osmiznakovou aplikaci následovanou čtyřznakovým identifikátorem tranid.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PMO** uvedeno PMSETA. Jakékoli informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota PMSETA, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou LNPAN. Počáteční hodnota tohoto pole je 28 prázdných znaků.

MDPAT (10místné celé číslo se znaménkem)

Typ aplikace, která vložila zprávu.

Toto je část **původního kontextu** zprávy. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Produkt *MDPAT* může mít jeden z následujících standardních typů. Uživatelsky definované typy mohou být také použity, ale měly by být omezeny na hodnoty v rozsahu ATUFST až ATULST.

ATAIX-operační systém
Aplikace AIX (stejná hodnota jako ATUNIX).

ATBRKR
Zprostředkovatel.

ATCICS:
CICS transakce.

ATCICB
CICS bridge.

ATVSE
CICS/VSE transakce.

ATDOS-operační systém
Aplikace IBM MQ MQI client na PC DOS.

ATDQM-řízení jakosti
Agent distribuovaného správce front.

ATGUAR (PRŮVODCE)
Aplikace Tandem Guardian (stejná hodnota jako ATNSK).

ATIMS-řízení
IMS .

ATIMSB
Most IMS .

ATJAVA-více
Java.

ATMVS
Aplikace MVS nebo TSO (stejná hodnota jako ATZOS).

ATNOTE:
Lotus Notes Aplikace agenta.

ATNSK
Tandemová aplikace jádra NonStop .

AT390
Aplikace OS/390 (stejná hodnota jako ATZOS).

AT400
IBM i .

ATQM-řízení jakosti
Správce front.

Systém ATUNIX
UNIX .

ATVOS
Stratus VOS aplikace.

ATWIN
16bitová aplikace Windows .

ATWINT
32bitová aplikace Windows .

ATXCF-vyhledávání
XCF.

ATZOS
z/OS .

ATDEF
Výchozí typ aplikace.

Jedná se o výchozí typ aplikace pro platformu, na které je aplikace spuštěna.

Poznámka: Hodnota této konstanty je specifická pro dané prostředí.

ATUNK-neschválené
Neznámý typ aplikace.

Tuto hodnotu lze použít k označení, že typ aplikace je neznámý, i když jsou přítomny jiné informace o kontextu.

ATUFST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

ATULST
Nejvyšší hodnota pro typ aplikace definovaný uživatelem.

Může se také vyskytnout následující speciální hodnota:

ATNCON
Ve zprávě nejsou obsaženy žádné informace o kontextu.

Tato hodnota je nastavena správcem front při vložení zprávy bez kontextu (tj. je zadána volba kontextu PMNOC).

Po načtení zprávy lze pro tuto hodnotu testovat MDPAT , aby se rozhodlo, zda má zpráva kontext (doporučuje se, aby aplikace MDPAT nebyla nikdy nastavena na ATNCON aplikací používající PMSETA, pokud jsou některá z ostatních kontextových polí neprázdná).

ATSIB
Označuje zprávu pocházející z jiného produktu systému zpráv IBM MQ a doručenou přes most SIB (Service Integration Bus).

Když správce front vygeneruje tyto informace v důsledku vložení aplikace, pole se nastaví na hodnotu, která je určena prostředím.


Všimněte si, že v systému IBM ije pole nastaveno na hodnotu AT400; správce front nikdy nepoužívá ATCICS v systému IBM i.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PMO** uvedeno PMSETA. Není-li zadána hodnota PMSETA, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor MDPAT , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude to hodnota MDPAT , která je uchována se zprávou, pokud je uchována (viz popis PMRET, kde jsou další podrobnosti o zachovaných publikacích), ale nepoužívá se jako MDPAT , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání MDPAT ve všech publikacích, která jim byla odeslána. Pokud zpráva nemá žádný kontext, pole je nastaveno na ATNCON.

Toto je výstupní pole pro volání MQGET. Počáteční hodnota tohoto pole je ATNCON.

MDPD (8bajtový znakový řetězec)

Datum, kdy byla zpráva vložena.

Toto je část *původního kontextu* zprávy. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Formát použitý pro datum, kdy je toto pole generováno správcem front, je:

- RRRRMMDD

kde znaky představují:

yyyy
rok (čtyři číslice)

MM
měsíc v roce (01 až 12)

DD
den v měsíci (01 až 31)

Greenwichský střední čas (GMT) se používá pro pole MDPD a MDPT pod podmínkou, že systémové hodiny jsou přesně nastaveny na GMT.

Pokud byla zpráva vložena jako součást pracovní jednotky, datum je datum, kdy byla zpráva vložena, a ne datum, kdy byla pracovní jednotka potvrzena.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PMO** uvedeno PMSETA. Obsah pole není kontrolován správcem front s tím rozdílem, že veškeré informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota PMSETA, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor MDPD , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude to hodnota MDPD , která je uchována se zprávou, pokud je uchována (viz popis PMRET, kde jsou další podrobnosti o zachovaných publikacích), ale nepoužívá se jako MDPD , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání MDPD ve všech publikacích, která jim byla odeslána. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou LNPDAT. Počáteční hodnota tohoto pole je 8 prázdných znaků.

MDPER (10místné celé číslo se znaménkem)

Perzistence zpráv.

Označuje, zda zpráva přežívá selhání systému a restartování správce front. Pro volání MQPUT a MQPUT1 musí být hodnota jedna z následujících:

PEPER
Zpráva je trvalá.

To znamená, že zpráva přežije selhání systému a restartuje správce front. Jakmile je zpráva vložena a pracovní jednotka putteru je potvrzena (pokud je zpráva vložena jako součást pracovní jednotky), je zpráva uchována v pomocné paměti. Zůstává tam, dokud není zpráva odebrána z fronty a dokud není potvrzena jednotka práce getter (pokud je zpráva načtena jako součást jednotky práce).

Je-li do vzdálené fronty odeslána trvalá zpráva, mechanizmus uložit-a-předat se používá k zadržení zprávy v každém správci front na trase do místa určení, dokud není známo, že zpráva dorazila do dalšího správce front.

Trvalé zprávy nelze umístit na:

- Dočasné dynamické fronty
- Sdílené fronty, ve kterých je úroveň struktury prostředku Coupling Facility nižší než tři nebo ve kterých není struktura prostředku Coupling Facility obnovitelná.

Trvalé zprávy lze umisťovat do trvalých dynamických front, předdefinovaných front a sdílených front, kde úroveň struktury prostředku Coupling Facility je 3 a prostředek CF je obnovitelný.

PENPER
Zpráva není trvalá.

To znamená, že zpráva obvykle nepřežije selhání systému nebo restartování správce front. To platí i v případě, že je během restartování správce front v pomocné paměti nalezena neporušená kopie zprávy.

Ve zvláštním případě sdílených front dočasné zprávy *přežijí* restarty správců front ve skupině sdílení front, ale nepřežijí selhání prostředku Coupling Facility používaného pro ukládání zpráv ve sdílených frontách.

PEQDEF
Zpráva má výchozí perzistenci.

- Jedná-li se o frontu klastru, je perzistence zprávy převzata z atributu **DefPersistence** definovaného ve správci cílové fronty, který vlastní konkrétní instanci fronty, do níž je zpráva umístěna. Obvykle mají všechny instance fronty klastru stejnou hodnotu pro atribut **DefPersistence** , i když to není nařízeno.

  Hodnota **DefPersistence** se zkopíruje do pole *MDPER* , když se zpráva umístí do cílové fronty. Pokud se **DefPersistence** následně změní, zprávy, které již byly umístěny do fronty, nebudou ovlivněny.
- Pokud fronta není frontou klastru, je perzistence zprávy převzata z atributu **DefPersistence** definovaného v lokálním správci front, a to i v případě, že je cílový správce front vzdálený.

  Pokud je v cestě rozlišení názvu fronty více než jedna definice, je výchozí perzistence převzata z hodnoty tohoto atributu v první definici v cestě. To může být:

  – Alias fronty
  – Lokální fronta
  – Lokální definice vzdálené fronty
  – Alias správce front
  – Přenosová fronta (například fronta `DefXmitQName`).

Hodnota **DefPersistence** se při vložení zprávy zkopíruje do pole MDPER . Pokud se **DefPersistence** následně změní, zprávy, které již byly vloženy, nebudou ovlivněny.

Ve stejné frontě mohou existovat trvalé i přechodné zprávy.

Při odpovídání na zprávu by aplikace měly obvykle pro zprávu odpovědi používat perzistenci zprávy požadavku.

Pro volání MQGET je vrácená hodnota buď PEPER, nebo PENPER.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je PEQDEF.

MDPRI (10místné celé číslo se znaménkem)

Priorita zprávy.

Pro volání MQPUT a MQPUT1 musí být hodnota větší nebo rovna nule; nula je nejnižší priorita. Lze také použít následující speciální hodnotu:

PRQDEF
Výchozí priorita pro frontu.

- Pokud se jedná o frontu klastru, priorita zprávy je převzata z atributu **DefPriority** , jak je definováno ve správci cílové fronty, který vlastní konkrétní instanci fronty, do které je zpráva umístěna. Obvykle mají všechny instance fronty klastru stejnou hodnotu pro atribut **DefPriority** , i když to není nařízeno.

  Hodnota **DefPriority** se zkopíruje do pole MDPRI , když se zpráva umístí do cílové fronty. Pokud se **DefPriority** následně změní, zprávy, které již byly umístěny do fronty, nebudou ovlivněny.
- Pokud fronta není frontou klastru, priorita zprávy je převzata z atributu **DefPriority** , jak je definován v lokálním správci front, i když je cílový správce front vzdálený.

  Pokud je v cestě rozlišení názvu fronty více než jedna definice, výchozí priorita je převzata z hodnoty tohoto atributu v první definici v cestě. To může být:

  – Alias fronty
  – Lokální fronta
  – Lokální definice vzdálené fronty
  – Alias správce front
  – Přenosová fronta (například fronta DefXmitQName).

  Hodnota **DefPriority** se při vložení zprávy zkopíruje do pole MDPRI . Pokud se **DefPriority** následně změní, zprávy, které již byly vloženy, nebudou ovlivněny.

Hodnota vrácená voláním MQGET je vždy větší nebo rovna nule; hodnota PRQDEF není nikdy vrácena.

Pokud je vložena zpráva s prioritou vyšší, než je maximum podporované lokálním správcem front (toto maximum je dáno atributem správce front **MaxPriority**), je zpráva přijata správcem front, ale umístěna do fronty s maximální prioritou správce front; volání MQPUT nebo MQPUT1 je dokončeno s hodnotou CCWARN a kódem příčiny RC2049. V poli MDPRI je však zachována hodnota určená aplikací, která zprávu vložila.

Při odpovídání na zprávu by aplikace měly obvykle pro zprávu odpovědi používat prioritu zprávy požadavku. V jiných situacích umožňuje zadání PRQDEF provést ladění priority bez změny aplikace.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je PRQDEF.

MDPT (8bajtový znakový řetězec)

Čas, kdy byla zpráva vložena.

Toto je část **původního kontextu** zprávy. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Formát použitý pro čas, kdy je toto pole generováno správcem front, je:

- HHMMSSTH

kde znaky představují (v pořadí):

HH
hodiny (00 až 23)

MM
minut (00 až 59)

SS
sekund (00 až 59; viz poznámka)

T
desetiny sekundy (0 až 9)

H
setiny sekundy (0 až 9)

Poznámka: Pokud jsou systémové hodiny synchronizovány s velmi přesným časovým standardem, je možné, že se ve vzácných případech vrátí 60 nebo 61 sekund v souboru MDPT. K tomu dochází, když jsou do globálního časového standardu vloženy přestupné sekundy.

Greenwichský střední čas (GMT) se používá pro pole MDPD a MDPT pod podmínkou, že systémové hodiny jsou přesně nastaveny na GMT.

Pokud byla zpráva vložena jako součást pracovní jednotky, čas je ten, kdy byla zpráva vložena, a ne čas, kdy byla transakce potvrzena.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PMO** uvedeno PMSETA. Obsah pole není kontrolován správcem front s tím rozdílem, že veškeré informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota PMSETA, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje hodnotu MDPT , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. Bude to hodnota MDPT , která je uchována se zprávou, pokud je uchována (viz popis PMRET, kde jsou další podrobnosti o zachovaných publikacích), ale nepoužívá se jako MDPT , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání MDPT ve všech publikacích, která jim byla odeslána. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána LNPTIM. Počáteční hodnota tohoto pole je 8 prázdných znaků.

MDREP (10místné celé číslo se znaménkem)

Volby pro zprávy sestavy.

Zpráva sestavy je zpráva o jiné zprávě, která se používá k informování aplikace o očekávaných nebo neočekávaných událostech souvisejících s původní zprávou. Pole MDREP umožňuje aplikaci odesílající původní zprávu určit, které zprávy sestavy jsou povinné, zda do nich mají být zahrnuta data zprávy aplikace a také (pro sestavy i odpovědi), jak mají být nastaveny identifikátory zprávy a korelace v sestavě nebo zprávě odpovědi. Lze požadovat jakýkoli nebo všechny (nebo žádný) z následujících typů zpráv sestavy:

- Výjimka
- Konec platnosti
- Potvrdit při příjezdu (COA)
- Potvrdit při doručení (COD)
- Pozitivní oznámení akce (PAN)

- Negativní oznámení akce (NAN)

Je-li požadován více než jeden typ zprávy sestavy nebo jsou-li zapotřebí jiné volby sestavy, lze hodnoty sečíst (nepřidávat stejnou konstantu více než jednou).

Aplikace, která obdrží zprávu sestavy, může určit příčinu, proč byla sestava vygenerována, prozkoumáním pole MDFB v deskriptoru MQMD; další podrobnosti viz pole MDFB .

Použití voleb sestavy při vkládání zprávy do tématu může způsobit generování a odeslání jedné nebo více zpráv sestavy do aplikace. Důvodem je skutečnost, že publikační zpráva může být odeslána do žádné, jedné nebo více odebírajícím aplikacím.

Volby výjimky: Můžete určit jednu z následujících voleb pro vyžádání zprávy sestavy výjimek.

AKTIVITA

Jsou vyžadovány sestavy aktivity

Tato volba sestavy umožňuje generování sestavy aktivity vždy, když je zpráva s touto sadou voleb sestavy zpracována podpůrnými aplikacemi.

Zprávy s touto nastavenou volbou sestavy musí být přijímány libovolným správcem front, a to i v případě, že tuto volbu 'nerozumí'. To umožňuje nastavit volbu sestavy pro libovolnou uživatelskou zprávu, a to i v případě, že jsou zpracovány předchozími správci front. Pro dosažení tohoto cíle je volba sestavy umístěna do podpole ROAUM.

Pokud proces (správce front nebo uživatelský proces) provádí aktivitu pro zprávu se sadou ROACT, může zvolit generování a vložení sestavy aktivity.

Volba sestavy aktivity umožňuje trasovat všechny zprávy v rámci sítě správců front. Volba sestavy může být uvedena v jakékoli aktuální uživatelské zprávě a okamžitě mohou začít počítat trasu zprávy přes síť. Pokud aplikace generující zprávu nemůže povolit generování sestavy aktivity, lze ji povolit pomocí křížové uživatelské procedury rozhraní API dodané administrátory správce front.

Pro sestavy aktivit lze použít několik podmínek:

1. Trasa bude méně podrobná, pokud bude v síti méně správců front, kteří budou schopni generovat sestavy aktivity.
2. Zprávy o činnosti nemusí být snadno "uspořádatelné", aby bylo možné určit trasu.
3. Sestavy aktivit nemusí být schopny najít trasu k požadovanému cíli.

ROEXC

Jsou vyžadovány sestavy výjimek.

Tento typ sestavy může být generován agentem kanálu zpráv, když je zpráva odeslána jinému správci front a zprávu nelze doručit do určené cílové fronty. Například cílová fronta nebo mezilehlá přenosová fronta mohou být plné nebo zpráva může být pro frontu příliš velká.

Generování zprávy sestavy výjimek závisí na perzistenci původní zprávy a na rychlosti kanálu zpráv (normální nebo rychlé), kterým původní zpráva prochází:

- Pro všechny trvalé zprávy a pro přechodné zprávy procházející běžnými kanály zpráv je zpráva o výjimce generována pouze v případě, že akci určenou odesílající aplikací pro chybový stav lze úspěšně dokončit. Odesílající aplikace může při výskytu chybového stavu určit jednu z následujících akcí pro řízení dispozice původní zprávy:
  - RODLQ (to způsobí, že původní zpráva bude umístěna do fronty nedoručených zpráv).
  - RODISC (způsobí vyřazení původní zprávy).

  Pokud akci určenou odesílající aplikací nelze úspěšně dokončit, původní zpráva je ponechána v přenosové frontě a není generována žádná zpráva sestavy výjimek.
- Pro přechodné zprávy procházející kanály rychlých zpráv je původní zpráva odebrána z přenosové fronty a zpráva o výjimce je vygenerována i v případě, že uvedenou akci pro chybový stav nelze úspěšně dokončit. Pokud je například zadána hodnota RODLQ, ale původní zprávu

nelze umístit do fronty nedoručených zpráv, protože (řekněme) je tato fronta plná, vygeneruje se zpráva sestavy výjimek a původní zpráva se vyřadí.

Další informace o normálních a rychlých kanálech zpráv naleznete v tématu Trvání zpráv .

Zpráva o výjimce se negeneruje, pokud aplikace, která vložila původní zprávu, může být synchronně upozorněna na problém pomocí kódu příčiny vráceného voláním MQPUT nebo MQPUT1 .

Aplikace mohou také odesílat sestavy výjimek, aby označily, že zprávu, kterou přijaly, nelze zpracovat (například proto, že se jedná o debetní transakci, která by způsobila, že by účet překročil svůj limit kreditu).

Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Nezadávejte více než jednu z možností ROEXC, ROEXCD a ROEXCF.

ROEXCD
Sestavy výjimek s požadovanými daty.

Toto je stejné jako ROEXC, kromě toho, že prvních 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví MQ , jsou zahrnuty do zprávy sestavy kromě 100 bajtů dat aplikace.

Nezadávejte více než jednu z možností ROEXC, ROEXCD a ROEXCF.

ROEXCF
Jsou vyžadovány sestavy výjimek s úplnými daty.

Toto je stejné jako ROEXC, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Nezadávejte více než jednu z možností ROEXC, ROEXCD a ROEXCF.

Volby vypršení platnosti: Můžete uvést jednu z následujících voleb pro vyžádání zprávy sestavy vypršení platnosti.

ROEXP
Jsou vyžadovány sestavy vypršení platnosti.

Tento typ sestavy je generován správcem front, pokud je zpráva před doručením do aplikace vyřazena, protože uplynula doba vypršení platnosti (viz pole MDEXP). Není-li tato volba nastavena, není generována žádná zpráva sestavy, pokud je zpráva z tohoto důvodu vyřazena (i když je zadána jedna z voleb ROEXC*).

Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Neuvádějte více než jednu z možností ROEXP, ROEXPD a ROEXPF.

ROEXPD
Jsou vyžadovány sestavy vypršení platnosti s daty.

Toto je stejné jako ROEXP, kromě toho, že prvních 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví MQ , jsou zahrnuty do zprávy sestavy kromě 100 bajtů dat aplikace.

Neuvádějte více než jednu z možností ROEXP, ROEXPD a ROEXPF.

ROEXPF
Jsou vyžadovány sestavy vypršení platnosti s úplnými daty.

Toto je stejné jako ROEXP, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Neuvádějte více než jednu z možností ROEXP, ROEXPD a ROEXPF.

Volby Potvrdit-při-příchodu: Můžete uvést jednu z následujících voleb, chcete-li vyžádat zprávu potvrzení-při-příchodu.

ROCOA-Přepočet
Jsou vyžadovány sestavy potvrzení při příjezdu.

Tento typ sestavy je generován správcem front, který vlastní cílovou frontu, když je zpráva umístěna do cílové fronty. Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Je-li zpráva vložena jako součást pracovní jednotky a cílovou frontou je lokální fronta, bude zpráva sestavy COA vygenerovaná správcem front k dispozici pro načtení pouze tehdy, je-li transakce potvrzena.

Sestava COA se nevygeneruje, pokud je pole MDFMT v deskriptoru zprávy FMXQH nebo FMDLH. To zabraňuje generování sestavy COA, pokud je zpráva vložena do přenosové fronty nebo je nedoručitelná a vložena do fronty nedoručených zpráv.

Nezadávejte více než jednu hodnotu ROCOA, ROCOAD a ROCOAF.

ROCOAD
Potvrďte-při-příchodu sestavy s požadovanými daty.

Toto je stejné jako ROCOA, kromě toho, že prvních 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví MQ , jsou zahrnuty do zprávy sestavy kromě 100 bajtů dat aplikace.

Nezadávejte více než jednu hodnotu ROCOA, ROCOAD a ROCOAF.

ROCOAF
Potvrďte-při-příjezdu sestavy s požadovanými úplnými daty.

Toto je stejné jako ROCOA, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta ve zprávě sestavy.

Nezadávejte více než jednu hodnotu ROCOA, ROCOAD a ROCOAF.

Volby vyřazení a vypršení platnosti: Můžete určit následující volbu pro nastavení času vypršení platnosti a příznak vyřazení pro zprávy sestavy.

ROPDAE
Nastavte čas vypršení platnosti zprávy sestavy a příznak vyřazení.

Tato volba zajišťuje, že zprávy sestav a zprávy odpovědí zdědí čas vypršení platnosti a příznak vyřazení (zda mají či nemají být vyřazovány) z původních zpráv. S touto volbou nastavenou na zprávy sestav a odpovědí:

1. Zdědit příznak RODISC (pokud byl nastaven).
2. Zdědit zbývající čas vypršení platnosti zprávy, pokud zpráva není sestavou vypršení platnosti. Jedná-li se o zprávu o vypršení platnosti, je doba vypršení platnosti nastavena na 60 sekund.

Při nastavení této volby platí následující:

Poznámka:

1. Zprávy sestav a odpovědí jsou generovány s příznakem vyřazení a hodnotou vypršení platnosti a nemohou zůstat v systému.
2. Zprávy trasovací trasy nemohou dosáhnout cílových front ve správcích front s povoleným netrasovacím trasováním.
3. Fronty nemohou být vyplněny sestavami, které nemohou být doručeny, pokud jsou komunikační spojení přerušena.
4. Odpovědi příkazového serveru zdědí zbývající vypršení platnosti požadavku.

Volby potvrzení při doručení: Můžete uvést jednu z následujících voleb pro vyžádání zprávy se zprávou potvrzení při doručení.

ROCOD
Jsou vyžadovány zprávy o potvrzení při doručení.

Tento typ sestavy je generován správcem front, když aplikace načte zprávu z cílové fronty způsobem, který způsobí odstranění zprávy z fronty. Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Pokud je zpráva načtena jako součást pracovní jednotky, zpráva sestavy se vygeneruje v rámci stejné pracovní jednotky, takže sestava nebude k dispozici, dokud nebude transakce potvrzena. Je-li jednotka práce vrácena zpět, sestava se neodešle.

Sestava COD se negeneruje, pokud je pole MDFMT v deskriptoru zprávy FMDLH. Tím zabráníte generování sestavy COD, pokud je zpráva nedoručitelná a vložená do fronty nedoručených zpráv.

Hodnota ROCOD není platná, pokud je cílovou frontou fronta XCF.

Neuvádějte více než jeden z ROCOD, ROCODD a ROCODF.

ROCODD
Sestavy potvrzení při doručení s požadovanými daty.

Toto je stejné jako ROCOD, kromě toho, že prvních 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví MQ , jsou zahrnuty do zprávy sestavy kromě 100 bajtů dat aplikace.

Je-li ve volání MQGET pro původní zprávu uveden parametr GMATM a načtená zpráva je oříznuta, množství dat zprávy aplikace umístěných ve zprávě sestavy je minimum:

- Délka původní zprávy
- 100 bajtů.

Parametr ROCODD není platný, pokud je cílovou frontou fronta XCF.

Neuvádějte více než jeden z ROCOD, ROCODD a ROCODF.

ROCODF
Sestavy potvrzení při doručení s požadovanými úplnými daty.

Toto je stejné jako ROCOD, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Parametr ROCODF není platný, pokud je cílovou frontou fronta XCF.

Neuvádějte více než jeden z ROCOD, ROCODD a ROCODF.

Volby oznámení akce: Můžete uvést jednu nebo obě z následujících voleb, chcete-li požadovat, aby přijímající aplikace odeslala zprávu sestavy kladné nebo záporné akce.

ROPAN
Jsou vyžadovány sestavy s pozitivními oznámením akce.

Tento typ sestavy je generován aplikací, která načte zprávu a jedná na ní. Označuje, že akce požadovaná ve zprávě byla úspěšně provedena. Aplikace generující sestavu určuje, zda mají být do sestavy zahrnuta nějaká data.

Kromě předání tohoto požadavku aplikaci, která zprávu načítá, neprovede správce front na základě této volby žádnou akci. Je odpovědností aplikace načítání generovat sestavu, je-li to vhodné.

RONAN
Jsou vyžadovány negativní sestavy oznámení akce.

Tento typ sestavy je generován aplikací, která načte zprávu a jedná na ní. Označuje, že akce požadovaná ve zprávě nebyla úspěšně provedena. Aplikace generující sestavu určuje, zda mají být do sestavy zahrnuta nějaká data. Může být například žádoucí zahrnout některá data označující, proč nebylo možné požadavek provést.

Kromě předání tohoto požadavku aplikaci, která zprávu načítá, neprovede správce front na základě této volby žádnou akci. Je odpovědností aplikace načítání generovat sestavu, je-li to vhodné.

Za určení, které podmínky odpovídají pozitivnímu jednání a které odpovídají negativnímu jednání, odpovídá žádost. Doporučuje se však, aby v případě, že žádost byla provedena pouze částečně, byla vygenerována zpráva NAN, a nikoli zpráva PAN. Doporučuje se také, aby každá možná podmínka odpovídala buď pozitivní akci, nebo negativní akci, ale ne obojí.

Volby identifikátoru zprávy: Můžete určit jednu z následujících voleb pro řízení způsobu nastavení `MDMID` zprávy sestavy (nebo zprávy odpovědi).

RONMI
Nový identifikátor zprávy.

Jedná se o výchozí akci a označuje, že pokud je sestava nebo odpověď vygenerována jako výsledek této zprávy, bude pro sestavu nebo zprávu odpovědi vygenerován nový soubor `MDMID` .

ROPMI
Předejte identifikátor zprávy.

Pokud je sestava nebo odpověď vygenerována jako výsledek této zprávy, `MDMID` této zprávy se zkopíruje do `MDMID` zprávy sestavy nebo odpovědi.

Hodnota `MsgId` zprávy publikování se bude lišit pro každého odběratele, který obdrží kopii publikace, a proto se hodnota `MsgId` zkopírovaná do zprávy sestavy nebo odpovědi bude pro každého z nich lišit.

Není-li tato volba uvedena, předpokládá se RONMI.

Volby identifikátoru korelace: Můžete určit jednu z následujících voleb pro řízení způsobu nastavení `MDCID` zprávy sestavy (nebo zprávy odpovědi).

ROCMTC
Kopírovat identifikátor zprávy do identifikátoru korelace.

Toto je výchozí akce a označuje, že pokud je sestava nebo odpověď vygenerována jako výsledek této zprávy, `MDMID` této zprávy se zkopíruje do `MDCID` zprávy sestavy nebo odpovědi.

Hodnota `MsgId` publikační zprávy se bude pro každého odběratele, který obdrží kopii publikace, lišit, a proto se hodnota `MsgId` zkopírovaná do `CorrelId` zprávy sestavy nebo odpovědi bude pro každého z nich lišit.

ROPCI
Předejte identifikátor korelace.

Pokud je sestava nebo odpověď vygenerována jako výsledek této zprávy, `MDCID` této zprávy se zkopíruje do `MDCID` zprávy sestavy nebo odpovědi.

Hodnota `MDCID` zprávy publikování bude specifická pro odběratele, pokud nepoužívá volbu SOSCID a nenastaví pole `SCDIC` v MQSD na hodnotu CINONE. Proto je možné, že se soubor `MDCID` zkopírovaný do souboru `MDCID` zprávy sestavy nebo odpovědi bude pro každou zprávu lišit.

Není-li tato volba uvedena, předpokládá se ROCMTC.

Servery, které odpovídají na požadavky nebo generují zprávy sestavy, se doporučují zkontrolovat, zda byly volby ROPMI nebo ROPCI nastaveny v původní zprávě. Pokud ano, servery by měly provést akci popsanou pro tyto volby. Není-li ani jeden z nich nastaven, servery by měly provést odpovídající výchozí akci.

: Můžete určit jednu z následujících voleb pro řízení dispozice původní zprávy, když ji nelze doručit do cílové fronty. Tyto volby se vztahují pouze na situace, které by vedly k vygenerování zprávy sestavy výjimek, pokud by byla vyžádána odesílající aplikací. Aplikace může nastavit volby odebrání nezávisle na vyžádání sestav výjimek.

RODLQ
Umístěte zprávu do fronty nedoručených zpráv.

Jedná se o výchozí akci a označuje, že zpráva by měla být umístěna do fronty nedoručených zpráv, pokud zprávu nelze doručit do cílové fronty. K tomu dochází v následujících situacích:

- Pokud aplikace, která vložila původní zprávu, nemůže být na problém upozorněna synchronně pomocí kódu příčiny vráceného voláním MQPUT nebo MQPUT1 . Pokud byla zpráva o výjimce vyžádána odesilatelem, je vygenerována zpráva o výjimce.
- Když aplikace, která vložila původní zprávu, vkládala do tématu

Pokud byla zpráva o výjimce vyžádána odesilatelem, bude vygenerována.

RODISC
Zrušit zprávu.

To znamená, že zpráva by měla být vyřazena, pokud ji nelze doručit do cílové fronty. K tomu dochází v následujících situacích:

- Pokud aplikace, která vložila původní zprávu, nemůže být na problém upozorněna synchronně pomocí kódu příčiny vráceného voláním MQPUT nebo MQPUT1 . Pokud byla zpráva o výjimce vyžádána odesilatelem, je vygenerována zpráva o výjimce.
- Když aplikace, která vložila původní zprávu, vkládala do tématu

Pokud byla zpráva o výjimce vyžádána odesilatelem, bude vygenerována.

Je-li vyžadováno vrácení původní zprávy odesilateli, aniž by původní zpráva byla umístěna do fronty nedoručených zpráv, měl by odesilatel zadat příkaz RODISC s parametrem ROEXCF.

Výchozí volba: Pokud nejsou vyžadovány žádné volby sestavy, můžete zadat následující:

RONONE
Nejsou vyžadovány žádné sestavy.

Tuto hodnotu lze použít k označení, že nebyly zadány žádné další volby. RONONE je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby tato volba byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Obecné informace:

1. Všechny požadované typy sestav musí být specificky vyžádány aplikací odesílající původní zprávu. Je-li například požadována sestava COA, ale není-li sestava výjimek, vygeneruje se sestava COA, když je zpráva umístěna do cílové fronty, ale není generována žádná sestava výjimek, pokud je cílová fronta plná, když zpráva dorazí do cílové fronty. Nejsou-li nastaveny žádné volby MDREP , správce front nebo agent kanálu zpráv (MCA) nevygenerují žádné zprávy sestavy.

   Některé volby sestavy lze zadat i v případě, že je lokální správce front nerozpozná. To je užitečné v případě, že má být volba zpracována cílovým správcem front. Další podrobnosti viz "Volby sestavy a příznaky zpráv na IBM i" na stránce 1425.

   Je-li požadována zpráva sestavy, název fronty, do které se má sestava odeslat, musí být uveden v poli MDRQ . Když je přijata zpráva sestavy, lze povahu sestavy určit prozkoumáním pole MDFB v deskriptoru zprávy.
2. Pokud správce front nebo agent MCA, který generuje zprávu sestavy, nemůže vložit zprávu sestavy do fronty odpovědí (například proto, že fronta odpovědí nebo přenosová fronta jsou plné), bude zpráva sestavy umístěna do fronty nedoručených zpráv. Pokud se to také nezdaří nebo neexistuje fronta nedoručených zpráv, bude akce provedena v závislosti na typu zprávy sestavy:
   - Pokud je zpráva sestavy sestavou výjimek, zpráva, která způsobila vygenerování sestavy výjimek, zůstane ve své přenosové frontě; tím se zajistí, že zpráva nebude ztracena.
   - Pro všechny ostatní typy sestav je zpráva sestavy vyřazena a zpracování pokračuje normálně. To se provádí buď proto, že původní zpráva již byla bezpečně doručena (pro zprávy sestavy COA nebo COD), nebo již není zajímavá (pro zprávu sestavy vypršení platnosti).

   Jakmile byla zpráva sestavy úspěšně umístěna do fronty (buď do cílové fronty, nebo do intermediační přenosové fronty), zpráva již není předmětem speciálního zpracování; je s ní zacházeno stejně jako s jakoukoli jinou zprávou.
3. Při generování sestavy se otevře fronta MDRQ a zpráva sestavy se vloží s použitím oprávnění MDUID v deskriptoru MQMD zprávy, která danou sestavu způsobila, s výjimkou následujících případů:

- Zprávy o výjimkách generované přijímajícím agentem MCA jsou vloženy s jakýmkoli oprávněním, které agent MCA použil při pokusu o vložení zprávy, která tuto zprávu způsobila. Atribut kanálu CDPA určuje použitý identifikátor uživatele.
- Sestavy COA generované správcem front jsou vkládány s libovolným oprávněním, které bylo použito v době, kdy byla zpráva způsobující sestavu vložena do správce front generujícího sestavu. Pokud byla například zpráva vložena přijímajícím agentem MCA s použitím identifikátoru uživatele agenta MCA, správce front vloží sestavu COA s použitím identifikátoru uživatele agenta MCA.

Aplikace generující sestavy by měly obvykle používat stejné oprávnění, jaké by měly použít ke generování odpovědi; toto by mělo být obvykle oprávnění identifikátoru uživatele v původní zprávě.

Pokud má sestava cestovat do vzdáleného cíle, mohou se odesilatelé a příjemci rozhodnout, zda ji přijmou, stejně jako to dělají pro jiné zprávy.

4. Pokud je požadována zpráva sestavy s daty:
   - Zpráva sestavy je vždy generována s množstvím dat požadovaných odesilatelem původní zprávy. Pokud je zpráva sestavy příliš velká pro frontu odpovědí, dojde k dříve popsanému zpracování; zpráva sestavy se nikdy nezkrátí, aby se vešla do fronty odpovědí.
   - Pokud je MDFMT původní zprávy FMXQH, data zahrnutá v sestavě nezahrnují MQXQH. Data sestavy začínají prvním bajtem dat za hodnotou MQXQH v původní zprávě. K tomu dochází, zda je fronta přenosovou frontou.
5. Je-li ve frontě odpovědí přijata zpráva COA, COD nebo zpráva o vypršení platnosti, je zaručeno, že původní zpráva byla doručena, byla doručena nebo vypršela její platnost. Avšak pokud je požadována jedna nebo více těchto zpráv sestavy a není přijata, nelze předpokládat opak, protože mohlo dojít k jedné z následujících událostí:
   a. Zpráva sestavy je zadržena, protože odkaz je mimo provoz.
   b. Zpráva sestavy je zadržena, protože v mezilehlé přenosové frontě nebo ve frontě odpovědí existuje blokovací podmínka (například fronta je plná nebo blokovaná pro vložení).
   c. Zpráva sestavy je ve frontě nedoručených zpráv.
   d. Když se správce front pokoušel vygenerovat zprávu sestavy, nemohl ji vložit do příslušné fronty a také ji nemohl vložit do fronty nedoručených zpráv, takže zprávu sestavy nebylo možné vygenerovat.
   e. Došlo k selhání správce front mezi hlášenou akcí (příjem, doručení nebo vypršení platnosti) a generováním odpovídající zprávy sestavy. (To se nestane pro zprávy sestavy COD, pokud aplikace načte původní zprávu v rámci pracovní jednotky, protože zpráva sestavy COD je generována v rámci stejné pracovní jednotky.)

   Zprávy sestavy výjimek mohou být zadrženy stejným způsobem z důvodů 1, 2 a 3 dříve. Pokud však agent MCA nemůže vygenerovat zprávu o výjimce (zprávu sestavy nelze vložit do fronty odpovědí ani do fronty nedoručených zpráv), původní zpráva zůstává v přenosové frontě u odesilatele a kanál je uzavřen. K tomu dochází bez ohledu na to, zda má být zpráva sestavy generována na odesílajícím nebo přijímajícím konci kanálu.
6. Pokud je původní zpráva dočasně blokována (což má za následek vygenerování zprávy o výjimce a vložení původní zprávy do fronty nedoručených zpráv), ale zablokování se vymaže a aplikace pak přečte původní zprávu z fronty nedoručených zpráv a znovu ji umístí do svého cíle, může dojít k následujícímu:
   - I když byla vygenerována zpráva sestavy výjimek, původní zpráva nakonec úspěšně dorazí na místo určení.
   - Pro jednu původní zprávu je generována více než jedna zpráva sestavy výjimek, protože původní zpráva může později narazit na další zablokování.

Hlásit zprávy při vkládání do tématu:

1. Sestavy lze generovat při vkládání zprávy do tématu. Tato zpráva bude odeslána všem odběratelům tématu, což může být nula, jedna nebo mnoho. To by mělo být zohledněno při výběru použití voleb sestavy, protože v důsledku toho by mohlo být generováno více zpráv sestavy.
2. Při vkládání zprávy do tématu může existovat mnoho cílových front, kterým má být poskytnuta kopie zprávy. Pokud se u některých z těchto cílových front vyskytl problém, například zaplnění fronty, závisí úspěšné dokončení operace MQPUT na nastavení NPMSGDLV nebo PMSGDLV (v závislosti na perzistenci zprávy). Pokud je nastavení takové, že doručení zprávy do cílové fronty musí být úspěšné (například se jedná o trvalou zprávu pro trvalého odběratele a parametr PMSGDLV je nastaven na hodnotu ALL nebo ALLDUR), úspěch je definován jako jedno z následujících kritérií:
   - Úspěšné vložení do fronty odběratele
   - Použití RODLQ a úspěšné vložení do fronty nedoručených zpráv, pokud fronta odběratele nemůže přijmout zprávu
   - Použijte příkaz RODISC, pokud fronta odběratele nemůže přijmout zprávu.

Zprávy sestavy pro segmenty zpráv:

1. Zprávy sestavy lze požadovat pro zprávy, které mají povolenou segmentaci (viz popis příznaku MFSEGA). Pokud správce front zjistí, že je nutné zprávu segmentovat, může být pro každý segment, který následně zjistí příslušnou podmínku, vygenerována zpráva sestavy. Aplikace by proto měly být připraveny přijímat více zpráv sestav pro každý typ požadovaných zpráv sestav. Pole `MDGID` ve zprávě sestavy lze použít ke korelaci více sestav s identifikátorem skupiny původní zprávy a pole `MDFB` , které se používá k identifikaci typu každé zprávy sestavy.
2. Pokud se k načtení zpráv sestav pro segmenty používá GMLOGO, mějte na paměti, že následující volání MQGET mohou vracet sestavy různých typů. Pokud jsou například pro zprávu segmentovanou správcem front požadovány sestavy COA i sestavy COD, mohou volání MQGET pro zprávy sestavy vracet zprávy sestav COA a COD prokládané nepředvídatelným způsobem. Tomu se lze vyhnout použitím volby GMCMPM (volitelně s GMATM). GMCMPM způsobí, že správce front znovu sestaví zprávy sestavy, které mají stejný typ sestavy. Například první volání MQGET může znovu sestavit všechny zprávy COA související s původní zprávou a druhé volání MQGET může znovu sestavit všechny zprávy COD. To, které se sestaví jako první, závisí na tom, který typ zprávy sestavy se objeví jako první ve frontě.
3. Aplikace, které samy vkládají segmenty, mohou pro každý segment určit různé volby sestavy. Je však třeba poznamenat následující body:
   - Pokud jsou segmenty načteny pomocí volby GMCMPM, správce front bude ctít pouze volby sestavy v prvním segmentu.
   - Pokud jsou segmenty načteny jeden po druhém a většina z nich má jednu z voleb ROCOD*, ale alespoň jeden segment nikoli, nebude možné použít volbu GMCMPM k načtení zpráv sestavy s jedním voláním MQGET nebo použít volbu GMASGA ke zjištění, kdy byly všechny zprávy sestavy přijaty.
4. V síti MQ mohou mít správci front různé schopnosti. Pokud je zpráva sestavy pro segment generována správcem front nebo agentem MCA, který nepodporuje segmentaci, správce front nebo agent MCA standardně nezahrnou nezbytné informace o segmentu do zprávy sestavy, což může ztížit identifikaci původní zprávy, která způsobila vygenerování sestavy. Tomuto problému se lze vyhnout vyžádáním dat se zprávou sestavy, tj. uvedením příslušných voleb RO* D nebo RO* F. Mějte však na paměti, že pokud je zadána hodnota RO* D, může být aplikaci, která načte zprávu sestavy, vrácena méně než 100 bajtů dat zprávy aplikace, pokud je zpráva sestavy generována správcem front nebo agentem MCA, který nepodporuje segmentaci.

Obsah deskriptoru zprávy pro zprávu sestavy: Když správce front nebo agent kanálu zpráv (MCA) vygeneruje zprávu sestavy, nastaví pole v deskriptoru zprávy na následující hodnoty a pak zprávu vloží běžným způsobem.

Tabulka 708. Hodnoty použité pro pole MQMD, když je zpráva sestavy generována systémem

Pole v deskriptoru MQMD	Použitá hodnota
MDSID	MDSIDV
MDVER	MDVER2
MDREP	RONONE
MDMT	MTRPRT
MDEXP	EIULIM
MDFB	Podle povahy zprávy (hodnota FBCOA, FBCOD, FBEXP nebo RC*)
MDENC	Zkopírováno z původního deskriptoru zprávy
MDCSI	Zkopírováno z původního deskriptoru zprávy
MDFMT	Zkopírováno z původního deskriptoru zprávy
MDPRI	Zkopírováno z původního deskriptoru zprávy
MDPER	Zkopírováno z původního deskriptoru zprávy
MDMID	Jak je určeno volbami sestavy v původním deskriptoru zprávy
MDCID	Jak je určeno volbami sestavy v původním deskriptoru zprávy
MDBOC	0
MDRQ	Mezery
MDRM	Název správce front
MDUID	Jak je nastaveno volbou PMPASI
MDACC	Jak je nastaveno volbou PMPASI
MDAID	Jak je nastaveno volbou PMPASI
MDPAT	ATQM nebo podle potřeby pro agenta kanálu zpráv
MDPAN	Prvních 28 bajtů názvu správce front nebo názvu agenta kanálu zpráv. Pro zprávy sestavy generované mostem IMS toto pole obsahuje název skupiny XCF a název člena XCF systému IMS , ke kterému se zpráva vztahuje.
MDPD	Datum odeslání zprávy sestavy
MDPT	Čas odeslání zprávy sestavy
MDAOD	Mezery
MDGID	Zkopírováno z původního deskriptoru zprávy
MDSEQ	Zkopírováno z původního deskriptoru zprávy
MDOFF	Zkopírováno z původního deskriptoru zprávy
MDMFL	Zkopírováno z původního deskriptoru zprávy
MDOLN	Zkopírováno z původního deskriptoru zprávy, pokud není OLUNDF, a nastaveno na délku původních dat zprávy, jinak

Aplikace generující sestavu se doporučuje nastavit podobné hodnoty, s výjimkou následujících:

- Pole MDRM lze nastavit na mezery (správce front jej při vložení zprávy změní na název lokálního správce front).
- Pole kontextu by měla být nastavena pomocí volby, která by byla použita pro odpověď, obvykle PMPASI.

Analýza pole sestavy: Pole MDREP obsahuje dílčí pole; z tohoto důvodu by aplikace, které potřebují zkontrolovat, zda odesilatel zprávy požadoval určitou sestavu, měly použít jednu z technik popsaných v části "Analýza pole sestavy na IBM i" na stránce 1426.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je RONONE.

MDRM (48bajtový znakový řetězec)

Název správce front odpovědí.

Jedná se o název správce front, kterému má být odeslána zpráva odpovědi nebo zpráva sestavy. MDRQ je lokální název fronty, která je definována v tomto správci front.

Je-li pole MDRM prázdné, lokální správce front vyhledá název **MDRQ** ve svých definicích front. Pokud existuje lokální definice vzdálené fronty s tímto názvem, je hodnota **MDRM** v přenášené zprávě nahrazena hodnotou atributu **RemoteQMgrName** z definice vzdálené fronty a tato hodnota bude vrácena v deskriptoru zprávy, když přijímající aplikace vydá pro zprávu volání MQGET. Pokud lokální definice vzdálené fronty neexistuje, je název MDRM , který je přenášen se zprávou, názvem lokálního správce front.

Je-li uveden název, může obsahovat koncové mezery; první znak null a následující znaky jsou považovány za mezery. V opačném případě se však neprovádí žádná kontrola, zda název splňuje pravidla pojmenování pro správce front nebo zda je tento název pro odesílajícího správce front znám. To platí i pro přenášený název, pokud je v přenášené zprávě nahrazen parametr **MDRM** .

Pokud není fronta pro odpověď požadována, doporučuje se (i když to není zaškrtnuto), aby pole MDRM bylo nastaveno na mezery; pole by nemělo být ponecháno neinicializované.

Pro volání MQGET vrací správce front vždy název vyplněný mezerami na délku pole.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Délka tohoto pole je dána hodnotou LNQMN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

MDRQ (48bajtový znakový řetězec)

Název fronty odpovědí.

Jedná se o název fronty zpráv, do které by aplikace, která vydala požadavek na získání zprávy, měla odesílat zprávy MTRPLY a MTRPRT. Název je lokální název fronty, která je definována ve správci front určeném pomocí MDRM. Tato fronta by neměla být modelovou frontou, ačkoli odesílající správce front tuto skutečnost při vložení zprávy neověřuje.

Pro volání MQPUT a MQPUT1 nesmí být toto pole prázdné, pokud má pole MDMT hodnotu MTRQST nebo pokud pole MDREP požaduje nějaké zprávy sestavy. Avšak uvedená hodnota (nebo nahrazená) se předá aplikaci, která vydá požadavek na získání zprávy, bez ohledu na typ zprávy.

Je-li pole MDRM prázdné, lokální správce front vyhledá název MDRQ ve svých vlastních definicích front. Pokud existuje lokální definice vzdálené fronty s tímto názvem, je hodnota MDRQ v přenášené zprávě nahrazena hodnotou atributu **RemoteQName** z definice vzdálené fronty a tato hodnota bude vrácena v deskriptoru zprávy, když přijímající aplikace vydá pro zprávu volání MQGET. Pokud lokální definice vzdálené fronty neexistuje, MDRQ se nezmění.

Je-li uveden název, může obsahovat koncové mezery; první znak null a následující znaky jsou považovány za mezery. Jinak se však neprovádí žádná kontrola, zda název splňuje pravidla pojmenování pro fronty; to platí i pro přenesený název, pokud je MDRQ nahrazen v přenášené zprávě. Jediná kontrola je, že název byl uveden, pokud to okolnosti vyžadují.

Pokud není fronta pro odpověď požadována, doporučuje se (i když to není zaškrtnuto), aby pole MDRQ bylo nastaveno na mezery; pole by nemělo být ponecháno neinicializované.

Pro volání MQGET vrací správce front vždy název vyplněný mezerami na délku pole.

Pokud zprávu, která vyžaduje zprávu sestavy, nelze doručit a zprávu sestavy také nelze doručit do uvedené fronty, původní zpráva i zpráva sestavy přejdou do fronty nedoručených zpráv (nedoručených zpráv). Viz atribut **DeadLetterQName** popsaný v části "Atributy pro správce front v systému IBM i" na stránce 1391.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

MDSEQ (10místné celé číslo se znaménkem)

Pořadové číslo logické zprávy ve skupině.

Pořadová čísla začínají na 1 a zvyšují se o 1 pro každou novou logickou zprávu ve skupině, maximálně 999 999 999 999. Fyzická zpráva, která není ve skupině, má pořadové číslo 1.

Toto pole nemusí být nastaveno aplikací ve volání MQPUT nebo MQGET, pokud:

- Ve volání MQPUT je zadána hodnota PMLOGO.
- Ve volání MQGET není uvedeno MOSEQN.

Toto jsou doporučené způsoby použití těchto volání pro zprávy, které nejsou zprávami sestavy. Pokud však aplikace vyžaduje větší kontrolu nebo volání je MQPUT1, musí aplikace zajistit, aby byl parametr MDSEQ nastaven na odpovídající hodnotu.

Při vstupu pro volání MQPUT a MQPUT1 používá správce front hodnotu popsanou v tabulce 1. Ve výstupu volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána se zprávou.

Při vstupu do volání MQGET používá správce front hodnotu popsanou v části Tabulka 1. Ve výstupu volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Počáteční hodnota tohoto pole je jedna. Toto pole se ignoruje, pokud je MDVER menší než MDVER2.

MDSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

MDSIDV
Identifikátor pro strukturu deskriptoru zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MDSIDV.

MDUID (12bajtový znakový řetězec)

Identifikátor uživatele.

Toto je část *kontextu identity* zprávy. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

MDUID uvádí identifikátor uživatele aplikace, která zprávu vyvolala. Správce front považuje tyto informace za znaková data, ale nedefinuje jejich formát.

Po přijetí zprávy lze MDUID použít v poli ODAU parametru **OBJDSC** následného volání MQOPEN nebo MQPUT1 , aby se místo aplikace provádějící otevření provedla kontrola autorizace pro uživatele MDUID .

Když správce front vygeneruje tyto informace pro volání MQPUT nebo MQPUT1 , použije správce front identifikátor uživatele určený z prostředí.

Když je identifikátor uživatele určen z prostředí:

- z/OS V systému z/OSpoužívá správce front:
  - V případě dávky se jedná o identifikátor uživatele z karty JES JOB nebo spuštěné úlohy.
  - V případě TSO se jedná o identifikátor přihlášeného uživatele.
  - V případě systému CICSse jedná o identifikátor uživatele přidružený k úloze.
  - V případě systému IMSZávisí identifikátor uživatele na typu aplikace:
    - Počet:
      - Oblasti bez zprávy BMP

- Regiony veřejné instituce IFP bez zpráv
- Zpráva BMP a zpráva oblasti IFP, které nevyvolaly úspěšné volání GU

Správce front používá identifikátor uživatele z karty JES JOB oblasti nebo identifikátor uživatele TSO. Pokud jsou prázdné nebo mají hodnotu null, použije název bloku specifikace programu (PSB).

- Počet:
  - Zpráva BMP a zpráva oblasti IFP, které vydaly úspěšné volání GU
  - Regiony veřejných zakázek

  správce front používá jednu z následujících možností:
  - Identifikátor přihlášeného uživatele přidružený ke zprávě
  - Název logického terminálu (LTERM)
  - Identifikátor uživatele z oblasti karty JES JOB
  - Identifikátor uživatele TSO
  - Název PSB

- IBM i V systému IBM ipoužívá správce front název profilu uživatele přidruženého k úloze aplikace.
- Linux AIX V systému AIX and Linuxpoužívá správce front:
  - Přihlašovací jméno aplikace
  - Efektivní identifikátor uživatele procesu, pokud není k dispozici žádné přihlášení
  - Identifikátor uživatele přidružený k transakci, pokud je aplikace transakcí CICS
- V systému VSE/ESAse jedná o vyhrazené pole.
- Windows V systému Windowspoužívá správce front prvních 12 znaků jména přihlášeného uživatele.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je v parametru **PMO** uvedeno PMSETI nebo PMSETA. Jakékoli informace následující za znakem null v poli jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li zadána hodnota PMSETI nebo PMSETA, je toto pole na vstupu ignorováno a jedná se o pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 toto pole obsahuje soubor `MDUID` , který byl spolu se zprávou přenesen, pokud byl vložen do fronty. Bude to hodnota `MDUID` , která je uchována se zprávou, pokud je uchována (viz popis PMRET, kde jsou další podrobnosti o zachovaných publikacích), ale nepoužívá se jako `MDUID` , když je zpráva odeslána jako publikování odběratelům, protože poskytují hodnotu k přepsání `MDUID` ve všech publikacích, která jim byla odeslána. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána hodnotou LNUID. Počáteční hodnota tohoto pole je 12 prázdných znaků.

MDVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

MDVER1
Version-1 struktura deskriptoru zpráv.

MDVER2
Version-2 struktura deskriptoru zpráv.

Poznámka: Je-li použita databáze MQMD version-2 , správce front provede další kontroly všech struktur záhlaví MQ , které mohou být přítomny na začátku dat zprávy aplikace; další podrobnosti naleznete v poznámkách k použití pro volání MQPUT.

Pole, která existují pouze v novější verzi struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MDVERC
Aktuální verze struktury deskriptoru zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MDVER1.

Počáteční hodnoty

Tabulka 709. Pole v deskriptoru MQMD

Název pole	Název konstanty	Hodnota konstanty
MDSID	MDSIDV	'MD¬¬'
MDVER	MDVER1	1
MDREP	RONONE	0
MDMT	MTDGRM	8
MDEXP	EIULIM	-1
MDFB	FBNONE	0
MDENC	ENNAT	Závisí na prostředí
MDCSI	CSQM	0
MDFMT	FMNONE	Mezery
MDPRI	PRQDEF	-1
MDPER	PEQDEF	2
MDMID	DOLNÉ	Hodnoty null
MDCID	CINONE	Hodnoty null
MDBOC	Není	0
MDRQ	Není	Mezery
MDRM	Není	Mezery
MDUID	Není	Mezery
MDACC	ACNONE	Hodnoty null
MDAID	Není	Mezery
MDPAT	ATNCON	0
MDPAN	Není	Mezery
MDPD	Není	Mezery
MDPT	Není	Mezery
MDAOD	Není	Mezery
MDGID	GINONE	Hodnoty null
MDSEQ	Není	1
MDOFF	Není	0
MDMFL	MFNONE	0
MDOLN	OLUNDF	-1

Tabulka 709. Pole v deskriptoru MQMD (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
Notes: 1. Symbol ¬ představuje jeden prázdný znak.		

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQMD Structure
D*
D* Structure identifier
D  MDSID                  1      4    INZ('MD  ')
D* Structure version number
D  MDVER                  5      8I 0 INZ(1)
D* Options for report messages
D  MDREP                  9     12I 0 INZ(0)
D* Message type
D  MDMT                  13     16I 0 INZ(8)
D* Message lifetime
D  MDEXP                 17     20I 0 INZ(-1)
D* Feedback or reason code
D  MDFB                  21     24I 0 INZ(0)
D* Numeric encoding of message data
D  MDENC                 25     28I 0 INZ(273)
D* Character set identifier of messagedata
D  MDCSI                 29     32I 0 INZ(0)
D* Format name of message data
D  MDFMT                 33     40    INZ('        ')
D* Message priority
D  MDPRI                 41     44I 0 INZ(-1)
D* Message persistence
D  MDPER                 45     48I 0 INZ(2)
D* Message identifier
D  MDMID                 49     72    INZ(X'00000000000000-
D                                     00000000000000000000-
D                                     000000000000')
D* Correlation identifier
D  MDCID                 73     96    INZ(X'00000000000000-
D                                     00000000000000000000-
D                                     000000000000')
D* Backout counter
D  MDBOC                 97    100I 0 INZ(0)
D* Name of reply queue
D  MDRQ                 101    148    INZ
D* Name of reply queue manager
D  MDRM                 149    196    INZ
D* User identifier
D  MDUID                197    208    INZ
D* Accounting token
D  MDACC                209    240    INZ(X'00000000000000-
D                                     00000000000000000000-
D                                     00000000000000000000-
D                                     000000')
D* Application data relating toidentity
D  MDAID                241    272    INZ
D* Type of application that put themessage
D  MDPAT                273    276I 0 INZ(0)
D* Name of application that put themessage
D  MDPAN                277    304    INZ
D* Date when message was put
D  MDPD                 305    312    INZ
D* Time when message was put
D  MDPT                 313    320    INZ
D* Application data relating toorigin
D  MDAOD                321    324    INZ
D* Group identifier
D  MDGID                325    348    INZ(X'00000000000000-
D                                     00000000000000000000-
D                                     000000000000')
D* Sequence number of logical messagewithin group
D  MDSEQ                349    352I 0 INZ(1)
D* Offset of data in physical messagefrom start of logical message
```

```
     D  MDOFF                353    356I 0 INZ(0)
     D* Message flags
     D  MDMFL                357    360I 0 INZ(0)
     D* Length of original message
     D  MDOLN                361    364I 0 INZ(-1)
```

IBM i MQMDE (rozšíření deskriptoru zpráv) na IBM i

Přehled

Účel: Struktura MQMDE popisuje data, která se někdy vyskytují před daty zprávy aplikace. Struktura obsahuje pole MQMD, která existují v deskriptoru MQMD version-2 , ale nikoli v deskriptoru MQMD version-1 .

Název formátu: FMMDE.

Znaková sada a kódování: Data v prostředí MQMDE musí být ve znakové sadě určené atributem správce front **CodedCharSetId** a v kódování lokálního správce front, které určuje ENNAT pro programovací jazyk C.

Znaková sada a kódování MQMDE musí být nastaveny do polí *MDCSI* a *MDENC* v:

- MQMD (pokud je struktura MQMDE na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQMDE (všechny ostatní případy).

Pokud prostředí MQMDE není ve znakové sadě a kódování správce front, bude prostředí MQMDE přijato, ale nebude uznáno, tj. prostředí MQMDE bude považováno za data zprávy.

Použití: Normální aplikace by měly používat modul MQMD version-2 . V takovém případě nebudou narazit na strukturu MQMDE. Avšak specializované aplikace a aplikace, které nadále používají MQMD version-1 , mohou v některých situacích narazit na MQMDE. Struktura MQMDE se může vyskytnout za následujících okolností:

- Určeno pro volání MQPUT a MQPUT1 .
- Vráceno voláním MQGET
- Ve zprávách v přenosových frontách

- “MQMDE určeno pro volání MQPUT a MQPUT1” na stránce 1148
- “MQMDE vráceno voláním MQGET” na stránce 1149
- “MQMDE ve zprávách v přenosových frontách” na stránce 1149
- “Pole” na stránce 1150
- “Počáteční hodnoty” na stránce 1152
- “Deklarace RPG” na stránce 1152

MQMDE určeno pro volání MQPUT a MQPUT1

Pokud u volání MQPUT a MQPUT1 aplikace poskytuje MQMD version-1 , může aplikace volitelně přidat k datům zprávy předponu MQMDE a nastavit pole *MDFMT* v MQMD na hodnotu FMMDE tak, aby označovala přítomnost MQMDE. Pokud aplikace neposkytne prostředí MQMDE, správce front předpokládá výchozí hodnoty pro pole v prostředí MQMDE. Výchozí hodnoty, které správce front používá, jsou stejné jako počáteční hodnoty struktury-viz Tabulka 711 na stránce 1152.

Pokud aplikace poskytuje předpony version-2 MQMD *a* před data zprávy aplikace MQMDE, struktury se zpracují, jak ukazuje Tabulka 710 na stránce 1149.

Tabulka 710. Akce správce front při zadání MQMDE na MQPUT nebo MQPUT1			
Verze MQMD	**Hodnoty polí version-2**	**Hodnoty odpovídajících polí v MQMDE**	**Akce prováděná správcem front**
1	-	Platný	MQMDE je uznána
2	Výchozí	Platný	MQMDE je uznána
2	Není výchozí	Platný	MQMDE se považuje za data zprávy
1 nebo 2	Libovolný	Neplatné	Volání selže s odpovídajícím kódem příčiny
1 nebo 2	Libovolný	MQMDE je v nesprávné znakové sadě nebo kódování nebo se jedná o nepodporovanou verzi	MQMDE se považuje za data zprávy

Existuje jeden zvláštní případ. Pokud aplikace používá příkaz MQMD version-2 k vložení zprávy, která je segmentem (tj. je nastaven příznak MFSEG nebo MFLSEG), a název formátu v deskriptoru MQMD je FMDLH, správce front vygeneruje strukturu MQMDE a vloží ji *mezi* strukturu MQDLH a data, která ji následují. V deskriptoru MQMD, který správce front uchovává se zprávou, jsou pole version-2 nastavena na výchozí hodnoty.

Některá pole, která existují v deskriptoru version-2 MQMD, ale nikoli version-1 MQMD, jsou vstupní/výstupní pole v deskriptoru MQPUT a MQPUT1. Správce front však nevrací žádné hodnoty v ekvivalentních polích ve výstupu MQMDE z volání MQPUT a MQPUT1 . Pokud aplikace tyto výstupní hodnoty vyžaduje, musí použít MQMD version-2 .

MQMDE vráceno voláním MQGET

Pokud aplikace ve volání MQGET poskytuje MQMD version-1 , správce front před zprávu vrácenou s MQMDE předřadí, ale pouze v případě, že jedno nebo více polí v MQMDE má jinou než výchozí hodnotu. Správce front nastaví pole *MDFMT* v MQMD na hodnotu FMMDE, aby označil přítomnost MQMDE.

Pokud aplikace poskytuje MQMDE na začátku parametru **BUFFER** , je MQMDE ignorováno. Při návratu z volání MQGET je nahrazen MQMDE pro zprávu (je-li potřeba) nebo přepsán daty zprávy aplikace (není-li MQMDE potřeba).

Pokud je MQMDE vráceno voláním MQGET, data v MQMDE jsou obvykle ve znakové sadě a kódování správce front. Prostředí MQMDE však může být v jiné znakové sadě a kódování, pokud:

- S MQMDE bylo zacházeno jako s daty volání MQPUT nebo MQPUT1 (okolnosti, které to mohou způsobit, viz Tabulka 710 na stránce 1149).
- Zpráva byla přijata ze vzdáleného správce front připojeného prostřednictvím připojení TCP a přijímající agent kanálu zpráv (MCA) nebyl správně nastaven (další informace viz Zabezpečení IBM MQ for IBM i objektů).

MQMDE ve zprávách v přenosových frontách

Zprávy v přenosových frontách mají předponu ve struktuře MQXQH, která v sobě obsahuje version-1 MQMD. Může být také přítomno prostředí MQMDE umístěné mezi strukturou MQXQH a daty zprávy aplikace, ale obvykle bude přítomno pouze v případě, že jedno nebo více polí v prostředí MQMDE má jinou než výchozí hodnotu.

Mezi strukturou MQXQH a daty zpráv aplikace se mohou vyskytovat i jiné struktury záhlaví IBM MQ . Pokud je například přítomno záhlaví nedoručených zpráv MQDLH a zpráva není segmentem, je pořadí následující:

- MQXQH (obsahující version-1 MQMD)
- MQMDE

- MQDLH
- Data zprávy aplikace

Pole

Struktura MQMDE obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

MECSI (10místné celé číslo se znaménkem)

Identifikátor znakové sady dat, která následují za MQMDE.

Tato volba určuje identifikátor znakové sady dat, která následují za strukturou MQMDE; nevztahuje se na znaková data v samotné struktuře MQMDE.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je toto pole platné. Lze použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota CSINHT vrácena voláním MQGET.

CSINHT nelze použít, pokud je hodnota pole *MDPAT* v MQMD ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

MEENC (10místné celé číslo se znaménkem)
MEENC (10místné celé číslo se znaménkem)

Tato volba určuje číselné kódování dat, která jsou v souladu se strukturou MQMDE; nevztahuje se na číselná data v samotné struktuře MQMDE.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je pole platné. Další informace o kódování dat viz pole *MDENC* popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105 .

Počáteční hodnota tohoto pole je ENNAT.

MEFLG (10místné celé číslo se znaménkem)

Obecné příznaky.

Lze zadat následující příznak:

MEFNON
Žádné příznaky.

Počáteční hodnota tohoto pole je MEFNON.

MEFMT (8bajtový znakový řetězec)

Název formátu dat, která následují za MQMDE.

Tato volba určuje název formátu dat, která se řídí strukturou MQMDE.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je toto pole platné. Další informace o názvech formátů viz pole *MDFMT* popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105 .

Počáteční hodnota tohoto pole je FMNONE.

MEGID (24bajtový bitový řetězec)

Identifikátor skupiny.

Viz pole *MDGID* popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105. Počáteční hodnota tohoto pole je GINONE.

MELEN (10místné celé číslo se znaménkem)

Délka struktury MQMDE.

Je definována následující hodnota:

MELEN2
Délka struktury rozšíření deskriptoru zpráv version-2 .

Počáteční hodnota tohoto pole je MELEN2.

MEMFL (10místné celé číslo se znaménkem)

Příznaky zprávy.

Viz pole *MDMFL* popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105. Počáteční hodnota tohoto pole je MFNONE.

MEOFF (10místné celé číslo se znaménkem)

Posunutí dat ve fyzické zprávě od začátku logické zprávy.

Viz pole *MDOFF* popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105. Počáteční hodnota tohoto pole je 0.

MEOLN (10místné celé číslo se znaménkem)

Délka původní zprávy.

Viz pole *MDOLN* popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105. Počáteční hodnota tohoto pole je OLUNDF.

MESEQ (10místné celé číslo se znaménkem)

Pořadové číslo logické zprávy ve skupině.

Viz pole *MDSEQ* popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105. Počáteční hodnota tohoto pole je 1.

MESID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

MESIDV
Identifikátor pro strukturu rozšíření deskriptoru zpráv.

Počáteční hodnota tohoto pole je MESIDV.

MEVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

MEVER2
Version-2 struktura rozšíření deskriptoru zpráv.

Následující konstanta určuje číslo verze aktuální verze:

MEVERC
Aktuální verze struktury rozšíření deskriptoru zpráv.

Počáteční hodnota tohoto pole je MEVER2.

Počáteční hodnoty

Tabulka 711. Pole v prostředí MQMDE		
Název pole	**Název konstanty**	**Hodnota konstanty**
MESID	MESIDV	'MDE¬'
MEVER	MEVER2	2
MELEN	MELEN2	72
MEENC	ENNAT	Závisí na prostředí
MECSI	CSUNDF	0
MEFMT	FMNONE	Mezery
MEFLG	MEFNON	0
MEGID	GINONE	Hodnoty null
MESEQ	Není	1
MEOFF	Není	0
MEMFL	MFNONE	0
MEOLN	OLUNDF	-1

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQMDE Structure
D*
D* Structure identifier
D  MESID                  1      4    INZ('MDE ')
D* Structure version number
D  MEVER                  5      8I 0 INZ(2)
D* Length of MQMDE structure
D  MELEN                  9     12I 0 INZ(72)
D* Numeric encoding of data that followsMQMDE
D  MEENC                 13     16I 0 INZ(273)
D* Character-set identifier of data thatfollows MQMDE
D  MECSI                 17     20I 0 INZ(0)
D* Format name of data that followsMQMDE
D  MEFMT                 21     28    INZ('        ')
D* General flags
D  MEFLG                 29     32I 0 INZ(0)
D* Group identifier
D  MEGID                 33     56    INZ(X'00000000000000-
D                                     000000000000000000000-
D                                     000000000000')
D* Sequence number of logical messagewithin group
D  MESEQ                 57     60I 0 INZ(1)
D* Offset of data in physical messagefrom start of logical message
D  MEOFF                 61     64I 0 INZ(0)
D* Message flags
D  MEMFL                 65     68I 0 INZ(0)
D* Length of original message
D  MEOLN                 69     72I 0 INZ(-1)
```

IBM i MQMHBO (Message handle to buffer options) na IBM i

Struktura definující volby popisovače zprávy do vyrovnávací paměti

Přehled

Účel: Struktura MQMHBO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou vyrovnávací paměti vytvářeny z manipulátorů zpráv. Struktura je vstupní parametr pro volání MQMHBUF.

Znaková sada a kódování: Data v MQMHBO musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1153
- "Počáteční hodnoty" na stránce 1154
- "Deklarace RPG" na stránce 1154

Pole

Struktura MQMHBO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

MBOPT (10místné celé číslo se znaménkem)

Popisovač zprávy pro strukturu voleb vyrovnávací paměti-pole MBOPT.

Tyto volby řídí akci MQMHBUF.

Musíte uvést následující volbu:

MBPRRF-přídavný modul
Při převodu vlastností z manipulátoru zprávy do vyrovnávací paměti je převeďte do formátu MQRFH2 .

Volitelně můžete také zadat následující volbu. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

MBDLPR
Vlastnosti přidané do vyrovnávací paměti jsou odstraněny z manipulátoru zprávy. Pokud volání selže, nebudou odstraněny žádné vlastnosti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MBPRRF.

MBSID (10místné celé číslo se znaménkem)

Popisovač zprávy pro strukturu voleb vyrovnávací paměti-pole MBSID.

Jedná se o identifikátor struktury. Hodnota musí být:

MBSIDV
Identifikátor pro strukturu voleb popisovače zprávy do vyrovnávací paměti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole isMBSIDV.

MBVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury. Hodnota musí být:

MBVER1
Číslo verze pro strukturu voleb popisovače zprávy do vyrovnávací paměti.

Následující konstanta určuje číslo verze aktuální verze:

MBVERC-všechny
Aktuální verze popisovače zprávy do struktury voleb vyrovnávací paměti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MBVER1.

Počáteční hodnoty

Tabulka 712. Pole v MQMHBO		
Název pole	**Název konstanty**	**Hodnota konstanty**
MVSID	MBSIDV	'MHBO'
MBVER	MBVER1	1
MBOPT	MBPRRF-přídavný modul	

Notes:

1. Hodnota Null řetězec nebo mezery označují prázdný znak.

Deklarace RPG

```
D* MQMHBO Structure
D*
D*
D* Structure identifier
D  MBSID                  1      4    INZ('MHBO')
D*
D* Structure version number
D  MBVER                  5      8I 0 INZ(1)
D*
D* Options that control the action of MQMHBUF
D  MBOPT                  9     12I 0 INZ(1)
```

IBM i MQOD (Object descriptor) na systému IBM i

Struktura MQOD se používá k určení objektu podle názvu.

Přehled

Účel: Následující typy objektů jsou platné:

- Fronta nebo distribuční seznam
- Seznam názvů
- Definice procesu
- Správce front
- Téma

Struktura je vstupní/výstupní parametr volání MQOPEN a MQPUT1 .

Verze: Aktuální verze produktu MQOD je ODVER4. Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi produktu MQOD podporovanou prostředím, ale s počáteční hodnotou pole *ODVER* nastavenou na hodnotu ODVER1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole *ODVER* na číslo verze požadované verze.

Chcete-li otevřít distribuční seznam, *ODVER* musí být ODVER2 nebo vyšší.

Znaková sada a kódování: Data v produktu MQOD musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , musí být struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1155
- "Počáteční hodnoty" na stránce 1161
- "Deklarace RPG" na stránce 1162

Pole

Struktura MQOD obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

ODASI (40bajtový bitový řetězec)

Alternativní identifikátor zabezpečení.

Jedná se o identifikátor zabezpečení, který je předán s produktem *ODAU* autorizační službě, aby bylo možné provést odpovídající kontroly autorizace. Parametr *ODASI* se používá pouze v případě, že:

- OOALTU je uveden ve volání MQOPEN, nebo
- PMALTU je uveden ve volání MQPUT1 ,

a pole *ODAU* není zcela prázdné až do prvního znaku null nebo konce pole.

Pole *ODASI* má následující strukturu:

- První bajt je binární celé číslo obsahující délku důležitých dat, která následují; hodnota vylučuje samotný bajt délky. Pokud není přítomen žádný identifikátor zabezpečení, je délka nula.
- Druhý bajt označuje typ identifikátoru zabezpečení, který je přítomen; jsou možné následující hodnoty:

  SITWNT
  Windows identifikátor zabezpečení.

  ORG. JEDNOTKY
  Žádný identifikátor zabezpečení.
- Třetí a následující bajty až do délky definované prvním bajtem obsahují samotný identifikátor zabezpečení.
- Zbývající bajty v poli jsou nastaveny na binární nulu.

Lze použít následující speciální hodnotu:

SINONE
Nebyl uveden žádný identifikátor zabezpečení.

Hodnota je binární nula pro délku pole.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou LNSCID. Počáteční hodnota tohoto pole je SINONE. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER3.

ODAU (12bajtový znakový řetězec)

Alternativní identifikátor uživatele.

Je-li pro volání MQOPEN zadána hodnota OOALTU nebo pro volání MQPUT1 hodnota PMALTU, obsahuje toto pole alternativní identifikátor uživatele, který má být použit ke kontrole autorizace pro otevření, namísto identifikátoru uživatele, pod kterým je aplikace aktuálně spuštěna. Některé kontroly jsou však stále prováděny s aktuálním identifikátorem uživatele (například kontextové kontroly).

Pokud OOALTU a PMALTU nejsou uvedeny a toto pole je zcela prázdné až po první znak null nebo konec pole, otevření může být úspěšné pouze tehdy, když není potřeba žádná autorizace uživatele k otevření tohoto objektu se zadanými volbami.

Není-li zadána hodnota OOALTU ani PMALTU, bude toto pole ignorováno.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou LNUID. Počáteční hodnota tohoto pole je 12 prázdných znaků.

ODDN (48bajtový znakový řetězec)

Název dynamické fronty.

Jedná se o název dynamické fronty, která má být vytvořena voláním MQOPEN. Toto má význam pouze v případě, že parametr *ODON* určuje název modelové fronty; ve všech ostatních případech je parametr *ODDN* ignorován.

Znaky, které jsou platné v názvu, jsou stejné jako znaky pro *ODON*, kromě toho, že hvězdička je také platná. Název, který je prázdný (nebo ve kterém jsou před prvním znakem null zobrazeny pouze mezery), není platný, pokud *ODON* je název modelové fronty.

Je-li posledním neprázdným znakem v názvu hvězdička (*), správce front nahradí hvězdičku řetězcem znaků, který zaručuje, že název vygenerovaný pro frontu je v lokálním správci front jedinečný. Pro povolení dostatečného počtu znaků je hvězdička platná pouze na pozicích 1 až 33. Za hvězdičkou nesmí být žádné jiné znaky než mezery nebo znak null.

Je platné, aby se hvězdička vyskytovala na první znakové pozici. V takovém případě se název skládá pouze ze znaků generovaných správcem front.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je `'AMQ.*'`, vyplněná mezerami.

ODIDC (10místné celé číslo se znaménkem)

Počet front, které se nepodařilo otevřít.

Jedná se o počet front v rozdělovníku, které se nepodařilo úspěšně otevřít. Je-li toto pole přítomno, je také nastaveno při otevírání jedné fronty, která není v rozdělovníku.

Poznámka: Je-li toto pole přítomno, je nastaveno pouze v případě, že parametr **CMPCOD** ve volání MQOPEN nebo MQPUT1 je CCOK nebo CCWARN; není nastaveno, pokud je parametr **CMPCOD** CCFAIL.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER2.

ODKDC (10místné celé číslo se znaménkem)

Počet úspěšně otevřených lokálních front.

Jedná se o počet front v rozdělovníku, které se interpretují na lokální fronty a které byly úspěšně otevřeny. Tento počet nezahrnuje fronty, které se interpretují na vzdálené fronty (i když se lokální přenosová fronta používá na počátku k uložení zprávy). Je-li toto pole přítomno, je také nastaveno při otevírání jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER2.

ODMN (48bajtový znakový řetězec)

Název správce front objektů.

Jedná se o název správce front, v němž je definován objekt *ODON* . Znaky platné v názvu jsou stejné jako znaky pro *ODON* (viz dříve). Název, který je zcela prázdný až do prvního znaku null nebo do konce pole, označuje správce front, ke kterému je aplikace připojena (lokální správce front).

Pro uvedené typy objektů platí následující body:

- Má-li parametr *ODOT* hodnotu OTTOP, OTNLST, OTPRO nebo OTQM, musí být *ODMN* prázdný nebo název lokálního správce front.
- Pokud je *ODON* název modelové fronty, vytvoří správce front dynamickou frontu s atributy modelové fronty a v poli *ODMN* vrátí název správce front, ve kterém je fronta vytvořena; jedná se o název lokálního správce front. Modelová fronta může být určena pouze pro volání MQOPEN; modelová fronta není platná pro volání MQPUT1 .
- Pokud je *ODON* název fronty klastru a *ODMN* je prázdný, skutečný cíl zpráv odeslaných pomocí popisovače fronty vráceného voláním MQOPEN je vybrán správcem front (nebo uživatelskou procedurou pracovní zátěže klastru, pokud je nainstalována) takto:
  - Je-li zadána volba OOBNDO, správce front vybere instanci fronty klastru během zpracování volání MQOPEN a všechny zprávy vkládané s použitím tohoto manipulátoru fronty budou odeslány do této instance.
  - Je-li zadána hodnota OOBNDN, může správce front zvolit jinou instanci cílové fronty (umístěnou v jiném správci front v klastru) pro každé následné volání MQPUT, které používá tento manipulátor fronty.

Pokud aplikace potřebuje odeslat zprávu do *specifické* instance fronty klastru (tj. do instance fronty, která se nachází v konkrétním správci front v klastru), měla by aplikace zadat název tohoto správce front do pole *ODMN* . To vynutí, aby lokální správce front odeslal zprávu určenému cílovému správci front.

- Pokud je otevíraný objekt rozdělovník (to znamená, že *ODREC* je větší než nula), *ODMN* musí být prázdný nebo prázdný řetězec. Není-li tato podmínka splněna, volání selže s kódem příčiny RC2153.

Jedná se o vstupní/výstupní pole pro volání MQOPEN, když *ODON* je název modelové fronty a pole pouze pro vstup ve všech ostatních případech. Délka tohoto pole je dána hodnotou LNQMN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

ODON (48bajtový znakový řetězec)

Název objektu.

Jedná se o lokální název objektu, jak je definován ve správci front identifikovaném pomocí *ODMN*. Název může obsahovat následující znaky:

- Velká písmena abecedy (A-Z)
- Malá písmena abecedy (a-z)
- Číselné číslice (0-9)
- Tečka (.), dopředné lomítko (/), podtržítko (_), procento (%)

Název nesmí obsahovat úvodní ani vložené mezery, ale může obsahovat koncové mezery. Znak null lze použít k označení konce důležitých dat v názvu; hodnota null a všechny následující znaky jsou považovány za mezery. V označených prostředích platí následující omezení:

- V systémech, které používají EBCDIC Katakana, nelze použít malá písmena.
- V systému IBM imusí být názvy obsahující malá písmena, dopředné lomítko nebo procenta uzavřeny v uvozovkách, jsou-li zadány v příkazech. Tyto uvozovky nesmí být uvedeny pro názvy, které se vyskytují jako pole ve strukturách nebo jako parametry ve voláních.

Pro uvedené typy objektů platí následující body:

- Pokud je *ODON* název modelové fronty, správce front vytvoří dynamickou frontu s atributy modelové fronty a vrátí do pole *ODON* název vytvořené fronty. Modelová fronta může být určena pouze pro volání MQOPEN; modelová fronta není platná pro volání MQPUT1 .
- Pokud je otevíraný objekt rozdělovník (to znamená, že *ODREC* je přítomen a větší než nula), *ODON* musí být prázdný nebo prázdný řetězec. Není-li tato podmínka splněna, volání selže s kódem příčiny RC2152.
- Je-li *ODOT* OTQM, použijí se speciální pravidla; v tomto případě musí být název zcela prázdný až po první znak null nebo konec pole.
- Pokud je ODON název alias fronty s TARGTYPE (TOPIC), provede se nejprve kontrola zabezpečení v pojmenované alias frontě, jak je obvyklé pro použití alias front. Pokud je tato kontrola zabezpečení úspěšná, bude toto volání MQOPEN pokračovat a bude se chovat jako MQOPEN OTTOP, včetně provedení kontroly zabezpečení objektu administrativního tématu.

Jedná se o vstupní/výstupní pole pro volání MQOPEN, když *ODON* je název modelové fronty a pole pouze pro vstup ve všech ostatních případech. Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

Úplný název tématu lze sestavit ze dvou různých polí: *ODON* a *ODOS*. Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat.

ODORO (10místné celé číslo se znaménkem)

Posunutí prvního záznamu objektu od začátku MQOD.

Toto je posun v bajtech prvního záznamu objektu MQOR od začátku struktury MQOD. Posun může být kladný nebo záporný. *ODORO* se používá pouze při otevírání rozdělovníku. Je-li hodnota *ODREC* nula, pole se ignoruje.

Při otevírání distribučního seznamu musí být zadáno pole jednoho nebo více záznamů objektů MQOR, aby bylo možné určit názvy cílových front v distribučním seznamu. To lze provést jedním ze dvou způsobů:

- Pomocí pole offsetu *ODORO*

  V tomto případě by aplikace měla deklarovat svou vlastní strukturu obsahující MQOD následovanou polem záznamů MQOR (s tolika prvky pole, kolik je potřeba) a nastavit *ODORO* na offset prvního prvku v poli od začátku MQOD. Je třeba dbát na to, aby toto posunutí bylo správné.

- Pomocí pole ukazatele *ODORP*

  V tomto případě může aplikace deklarovat pole struktur MQOR odděleně od struktury MQOD a nastavit *ODORP* na adresu pole.

Ať je zvolena libovolná technika, musí se použít jedna z metod *ODORO* a *ODORP* ; volání selže s kódem příčiny RC2155 , pokud jsou obě nulové, nebo jsou obě nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER2.

ODORP (ukazatel)

Adresa prvního záznamu objektu.

Jedná se o adresu prvního záznamu objektu MQOR. *ODORP* se používá pouze při otevírání rozdělovníku. Je-li hodnota *ODREC* nula, pole se ignoruje.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null. Buď *ODORP* , nebo *ODORO* lze použít k uvedení záznamů objektů, ale ne obojí; podrobnosti viz popis pole *ODORO* dříve. Není-li parametr *ODORP* použit, musí být nastaven na nulový ukazatel nebo nulový počet bajtů. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER2.

ODOS (MQCHARV)

ODOS uvádí dlouhý název objektu, který se má použít.

Toto pole je odkazováno pouze pro určité hodnoty *ODOT*. Podrobnosti o tom, které hodnoty označují použití tohoto pole, naleznete v popisu *ODOT* .

Pokud je parametr *ODOS* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překračuje maximální délku, volání selže s kódem příčiny RC2441.

Toto je vstupní pole. Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQCHARV.

Úplný název tématu lze sestavit ze dvou různých polí: *ODON* a *ODOS*. Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER4.

ODOT (10místné celé číslo se znaménkem)

Typ objektu.

Typ objektu, který je pojmenován v souboru *ODON*. Možné hodnoty jsou:

OTQ (dotazů)
Fronta. Název objektu se nachází v adresáři *ODON*.

OTNLST
Seznam názvů. Název objektu se nachází v adresáři *ODON*.

OTPRO
Definice procesu. Název objektu se nachází v adresáři *ODON*.

OTQM-řízení kvality
Správce front. Název objektu se nachází v adresáři *ODON*.

OTTOP
. Úplný název tématu lze sestavit ze dvou různých polí: *ODON* a *ODOS*.

Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat.

Pokud objekt identifikovaný polem *ODON* nelze nalézt, volání selže s kódem příčiny RC2425 , i když je v souboru *ODOS*uveden řetězec.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je OTQ.

ODREC (10místné celé číslo se znaménkem)

Počet přítomných záznamů objektů.

Jedná se o počet záznamů objektů MQOR, které byly poskytnuty aplikací. Je-li toto číslo větší než nula, znamená to, že se otevírá distribuční seznam, přičemž *ODREC* je počet cílových front v seznamu. Je platné, aby rozdělovník obsahoval pouze jeden cíl.

Hodnota *ODREC* nesmí být menší než nula, a pokud je větší než nula *ODOT* , musí být OTQ; volání selže s kódem příčiny RC2154 , pokud nejsou tyto podmínky splněny.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER2.

ODRMN (48bajtový znakový řetězec)

Vyřešený název správce front.

Jedná se o název cílového správce front po provedení překladu názvů lokálním správcem front. Vrácený název je název správce front, který vlastní frontu označenou jako *ODRQN*. *ODRMN* může být název lokálního správce front.

Pokud je *ODRQN* sdílená fronta, kterou vlastní skupina sdílení front, do které lokální správce front patří, *ODRMN* je název skupiny sdílení front. Pokud je fronta vlastněna jinou skupinou sdílení front, může být *ODRQN* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (povaha vrácené hodnoty je určena definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou otevřenou pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Pokud je otevřený objekt některý z následujících, *ODRMN* je nastaven na mezery:

- Nejedná se o frontu
- Fronta, ale neotevřená pro procházení, vstup nebo výstup
- Fronta klastru se zadaným OOBNDN (nebo s OOBNDQ v platnosti, když má atribut fronty **DefBind** hodnotu BNDNOT)
- Distribuční seznam

Toto je výstupní pole. Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER3.

ODRO (MQCHARV)

ODRO je dlouhý název objektu poté, co správce front přeloží název uvedený v souboru *ODON*.

Toto pole je vráceno pouze pro určité typy objektů, témat a aliasů front, které odkazují na objekt tématu.

Pokud je dlouhý název objektu uveden v souboru *ODOS* a v souboru *ODON*není nic uvedeno, hodnota vrácená v tomto poli je stejná jako v souboru *ODOS*.

Je-li toto pole vynecháno (tj. ODRO.VSBufSize je nula), hodnota *ODRO* není vrácena, ale délka je vrácena v ODRO.VSLength. Pokud je délka kratší než plná hodnota *ODRO* , je oříznuta a vrací tolik znaků nejvíce vpravo, kolik se vejde do uvedené délky.

Pokud je parametr *ODRO* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překročí maximální délku, volání selže s kódem příčiny RC2520. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER4.

ODRQN (48bajtový znakový řetězec)

Vyřešený název fronty.

Jedná se o název cílové fronty po provedení rozlišování názvů lokálním správcem front. Vrácený název je název fronty, která existuje ve správci front identifikovaném pomocí *ODRMN*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou otevřenou pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Pokud je otevřený objekt některý z následujících, *ODRQN* je nastaven na mezery:

- Nejedná se o frontu
- Fronta, ale neotevřená pro procházení, vstup nebo výstup
- Distribuční seznam
- Alias fronty, která odkazuje na objekt tématu (místo toho viz "ODRO (MQCHARV)" na stránce 1159).

Toto je výstupní pole. Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je nulový řetězec v jazyce C a 48 prázdných znaků v jiných programovacích jazycích. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER3.

ODRRO (10místné celé číslo se znaménkem)

Posunutí prvního záznamu odezvy od začátku MQOD.

Toto je posun v bajtech prvního záznamu odpovědi MQRR od začátku struktury MQOD. Posun může být kladný nebo záporný. *ODRRO* se používá pouze při otevírání rozdělovníku. Je-li hodnota *ODREC* nula, pole se ignoruje.

Při otevírání distribučního seznamu lze zadat pole jednoho nebo více záznamů odpovědí MQRR, aby bylo možné identifikovat fronty, které se nepodařilo otevřít (pole*RRCC* v MQRR), a příčinu každého selhání (pole*RRREA* v MQRR). Data jsou vrácena v poli záznamů odpovědí ve stejném pořadí, v jakém se názvy front vyskytují v poli záznamů objektů. Správce front nastaví záznamy odpovědí pouze v případě, že je výsledek volání smíšený (to znamená, že některé fronty byly úspěšně otevřeny, zatímco jiné selhaly, nebo všechny selhaly, ale z různých důvodů); kód příčiny RC2136 z volání označuje tento případ. Pokud stejný kód příčiny platí pro všechny fronty, je tato příčina vrácena v parametru **REASON** volání MQOPEN nebo MQPUT1 a záznamy odezvy nejsou nastaveny. Záznamy odpovědí jsou volitelné, ale pokud jsou zadány, musí z nich být *ODREC* .

Záznamy odezvy mohou být poskytnuty stejným způsobem jako záznamy objektů, buď uvedením offsetu v *ODRRO*, nebo uvedením adresy v *ODRRP* ; Podrobné informace o tom, jak to provést, naleznete v popisu souboru *ODORO* . Avšak nelze použít více než jeden z *ODRRO* a *ODRRP* ; volání selže s kódem příčiny RC2156 , pokud jsou obě nenulové.

Pro volání MQPUT1 se tyto záznamy odpovědí používají k vrácení informací o chybách, které se vyskytnou při odeslání zprávy do front v distribučním seznamu, a také o chybách, které se vyskytnou při otevření front. Kód dokončení a kód příčiny z operace vložení pro frontu nahradí kód příčiny z operace otevření pro tuto frontu pouze v případě, že kód dokončení z této fronty byl CCOK nebo CCWARN.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER2.

ODRRP (ukazatel)

Adresa prvního záznamu odpovědi.

Jedná se o adresu prvního záznamu odpovědi MQRR. *ODRRP* se používá pouze při otevírání rozdělovníku. Je-li hodnota *ODREC* nula, pole se ignoruje.

K zadání záznamů odpovědí lze použít buď *ODRRP* , nebo *ODRRO* , ale ne obojí; podrobnosti viz předchozí popis pole *ODRRO* . Není-li parametr *ODRRP* použit, musí být nastaven na nulový ukazatel nebo nulový počet bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER2.

ODSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

ODSIDV
Identifikátor pro strukturu deskriptoru objektu.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je ODSIDV.

ODSS (MQCHARV)

ODSS obsahuje řetězec použitý k poskytnutí kritérií výběru použitých při načítání zpráv z fronty.

Parametr *ODSS* nesmí být uveden v následujících případech:

- Pokud *ODOT* není OTQ
- Pokud se otevíraná fronta neotevírá pomocí jedné ze vstupních voleb, OOINP*

Pokud je v těchto případech uveden parametr *ODSS* , volání selže s kódem příčiny RC2516.

Pokud je parametr *ODSS* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překročí maximální délku, volání selže s kódem příčiny RC2519. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER4.

ODUDC (10místné celé číslo se znaménkem)

Počet úspěšně otevřených vzdálených front

Jedná se o počet front v rozdělovníku, které se interpretují na vzdálené fronty a které byly úspěšně otevřeny. Je-li toto pole přítomno, je také nastaveno při otevírání jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *ODVER* menší než ODVER2.

ODVER (celé číslo se znaménkem 10 číslic)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

ODVER1
Version-1 struktura deskriptoru objektu.

ODVER2
Version-2 struktura deskriptoru objektu.

ODVER3
Struktura deskriptoru objektu Version-3 .

ODVER4
Struktura deskriptoru objektu Version-4 .

Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

Směrovač ODVERC
Aktuální verze struktury deskriptoru objektu.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je ODVER1.

Počáteční hodnoty

Tabulka 713. Pole v produktu MQOD

Název pole	Název konstanty	Hodnota konstanty
ODSID	ODSIDV	'OD¬¬'

Tabulka 713. Pole v produktu MQOD (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
ODVER	ODVER1	1
ODOT	OTQ (dotazů)	1
ODON	Není	Mezery
ODMN	Není	Mezery
ODDN	Není	'AMQ.*'
ODAU	Není	Mezery
ODREC	Není	0
ODKDC	Není	0
ODUDC	Není	0
ODIDC	Není	0
ODORO	Není	0
ODRRO	Není	0
ODORP	Není	Ukazatel Null nebo bajty s hodnotou Null
ODRRP	Není	Ukazatel Null nebo bajty s hodnotou Null
ODASI	SINONE	Hodnoty null
ODRQN	Není	Mezery
ODRMN	Není	Mezery
ODOS	Podle definice pro MQCHARV	Podle definice pro MQCHARV
ODRO	Jak je uvedeno v části *ODOS*	Jak je uvedeno v části *ODOS*
ODSS	Není	Mezery
Notes: 1. Symbol ¬ představuje jeden prázdný znak.		

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQOD Structure
D*
D*
D* Structure identifier
D  ODSID                  1      4    INZ('OD  ')
D*
D* Structure version number
D  ODVER                  5      8I 0 INZ(1)
D*
D* Object type
D  ODOT                   9     12I 0 INZ(1)
D*
D* Object name
D  ODON                  13     60    INZ
D*
D* Object queue manager name
D  ODMN                  61    108    INZ
```

```
D*
D* Dynamic queue name
D  ODDN                 109    156    INZ('AMQ.*')
D*
D* Alternate user identifier
D  ODAU                 157    168    INZ
D*
** Number of object records
D* present
D  ODREC                169    172I 0 INZ(0)
D*
** Number of local queues opened
D* successfully
D  ODKDC                173    176I 0 INZ(0)
D*
** Number of remote queues opened
D* successfully
D  ODUDC                177    180I 0 INZ(0)
D*
** Number of queues that failed to
D* open
D  ODIDC                181    184I 0 INZ(0)
D*
** Offset of first object record
D* from start of MQOD
D  ODORO                185    188I 0 INZ(0)
D*
** Offset of first response record
D* from start of MQOD
D  ODRRO                189    192I 0 INZ(0)
D*
D* Address of first object record
D  ODORP                193    208*   INZ(*NULL)
D*
** Address of first response
D* record
D  ODRRP                209    224*   INZ(*NULL)
D*
D* Alternate security identifier
D  ODASI                225    264    INZ(X'00000000000000000-
D                                     0000000000000000000000000-
D                                     0000000000000000000000000-
D                                     0000000000000')
D*
D* Resolved queue name
D  ODRQN                265    312    INZ
D*
D* Resolved queue manager name
D  ODRMN                313    360    INZ
D*
D* reserved field
D  ODRE1                361    364I 0 INZ(0)
D*
D* reserved field
D  ODRS2                365    368I 0 INZ(0)
D*
D* Object long name
D* Address of variable length string
D  ODOSCHRP             369    384*   INZ(*NULL)
D* Offset of variable length string
D  ODOSCHRO             385    388I 0 INZ(0)
D* Size of buffer
D  ODOSVSBS             389    392I 0 INZ(-1)
D* Length of variable length string
D  ODOSCHRL             393    396I 0 INZ(0)
D* CCSID of variable length string
D  ODOSCHRC             397    400I 0 INZ(-3)
D*
D* Message Selector
D* Address of variable length string
D  ODSSCHRP             401    416*   INZ(*NULL)
D* Offset of variable length string
D  ODSSCHRO             417    420I 0 INZ(0)
D* Size of buffer
D  ODSSVSBS             421    424I 0 INZ(-1)
D* Length of variable length string
D  ODSSCHRL             425    428I 0 INZ(0)
D* CCSID of variable length string
D  ODSSCHRC             429    432I 0 INZ(-3)
D*
D* Resolved long object name
D* Address of variable length string
```

```
     D  ODRSOCHRP            433    448*   INZ(*NULL)
     D* Offset of variable length string
     D  ODRSOCHRO            449    452I 0 INZ(0)
     D* Size of buffer
     D  ODRSOVSBS            453    456I 0 INZ(-1)
     D* Length of variable length string
     D  ODRSOCHRL            457    460I 0 INZ(0)
     D* CCSID of variable length string
     D  ODRSOCHRC            461    464I 0 INZ(-3)
     D*
     D* Alias queue resolved object type
     D  ODRT                 465    468I 0 INZ(0)
```

IBM i MQOR (Záznam objektu) na IBM i

Struktura MQOR se používá k určení názvu fronty a názvu správce front pro jednu cílovou frontu.

Přehled

Účel: MQOR je vstupní struktura pro volání MQOPEN a MQPUT1 .

Znaková sada a kódování: Data v MQOR musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , musí být struktura ve znakové sadě a kódování klienta.

Použití: poskytováním pole těchto struktur ve volání MQOPEN je možné otevřít seznam front; tento seznam se nazývá *distribuční seznam*. Každá zpráva vložená s použitím manipulátoru fronty vráceného tímto voláním MQOPEN je umístěna do každé z front v seznamu, pokud byla fronta úspěšně otevřena.

- "Pole" na stránce 1164
- "Počáteční hodnoty" na stránce 1164
- "Deklarace RPG" na stránce 1165

Pole

Struktura MQOR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

ORMN (48bajtový znakový řetězec)

Název správce front objektů.

Toto je stejné jako pole *ODMN* ve struktuře MQOD (podrobnosti viz MQOD).

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je 48 prázdných znaků.

ORON (48bajtový znakový řetězec)

Název objektu.

Toto je stejné jako pole *ODON* ve struktuře MQOD (podrobnosti viz MQOD), kromě následujících:

- Musí to být název fronty.
- Nesmí se jednat o název modelové fronty.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je 48 prázdných znaků.

Počáteční hodnoty

Tabulka 714. Pole v MQOR

Název pole	Název konstanty	Hodnota konstanty
ORON	Není	Mezery
ORMN	Není	Mezery

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQOR Structure
D*
D* Object name
D  ORON                   1     48    INZ
D* Object queue manager name
D  ORMN                  49     96    INZ
```

MQPD-deskriptor vlastnosti

MQPD se používá k definování atributů vlastnosti.

Přehled

Účel: Struktura je vstupní/výstupní parametr volání MQSETMP a výstupní parametr volání MQINQMP.

Znaková sada a kódování: Data v MQPD musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1165
- "Počáteční hodnoty" na stránce 1167
- "Deklarace RPG" na stránce 1167

Pole

Struktura MQPD obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

PDCT (10místné celé číslo se znaménkem)

Popisuje, do jakému kontextu zprávy vlastnost patří.

Když správce front obdrží zprávu obsahující vlastnost definovanou proměnnou IBM MQ, kterou správce front rozpozná jako nesprávnou. správce front opraví hodnotu pole *PDCT* .

Lze zadat následující volbu:

PDUSC
Vlastnost je přidružena ke kontextu uživatele.

Pro nastavení vlastnosti přidružené ke kontextu uživatele pomocí volání MQSETMP není vyžadována žádná speciální autorizace.

Pokud není dříve popsaná volba požadována, lze použít následující volbu:

PDNOC
Vlastnost není přidružena ke kontextu zprávy.

Nerozpoznaná hodnota je odmítnuta s kódem *PDREA* RC2482.

Jedná se o vstupní/výstupní pole pro volání MQSETMP a výstupní pole pro volání MQINQMP. Počáteční hodnota tohoto pole je PDNOC.

PDCPYOPT (10místné celé číslo se znaménkem)

Popisuje, do kterého typu zpráv má být vlastnost zkopírována.

Toto je pouze výstupní pole pro rozpoznané IBM MQ-defined properties; IBM MQ nastavuje odpovídající hodnotu.

Když správce front obdrží zprávu obsahující vlastnost definovanou proměnnou IBM MQ, kterou správce front rozpozná jako nesprávnou. správce front opraví hodnotu pole *CopyOptions* .

Můžete uvést jednu nebo více těchto voleb. Chcete-li zadat více než jednu volbu, buď sečtěte hodnoty (nepřidávejte stejnou konstantu více než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

COPFOR:
Tato vlastnost je zkopírována do předávané zprávy.

COPPUB
Tato vlastnost je zkopírována do zprávy přijaté odběratelem při publikování zprávy.

COPREP-předtisk
Tato vlastnost je zkopírována do zprávy odpovědi.

COPRP
Tato vlastnost je zkopírována do zprávy sestavy.

COPALL
Tato vlastnost je zkopírována do všech typů následných zpráv.

COPNON
Tato vlastnost není zkopírována do zprávy.

Výchozí volba: Pro dodání výchozí sady voleb kopírování lze zadat následující volbu:

COPDEF
Tato vlastnost je zkopírována do předávané zprávy, do zprávy sestavy nebo do zprávy přijaté odběratelem při publikování zprávy.

Jedná se o ekvivalent k určení kombinace voleb COPFOR, plus COPRP a COPPUB.

Pokud není vyžadována žádná z dříve popsaných voleb, použijte následující volbu:

COPNON
Pomocí této hodnoty označíte, že nebyly zadány žádné další volby kopírování; programově neexistuje žádný vztah mezi touto vlastností a následnými zprávami. Tato hodnota je vždy vrácena pro vlastnosti deskriptoru zpráv.

Jedná se o vstupní/výstupní pole pro volání MQSETMP a výstupní pole pro volání MQINQMP. Počáteční hodnota tohoto pole je COPDEF.

PDOPT (10místné celé číslo se znaménkem)

Hodnota musí být:

PDNONE
Nejsou uvedeny žádné volby

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je PDNONE.

PDSID (10místné celé číslo se znaménkem)

Toto je identifikátor struktury; hodnota musí být:

PSIDV
Identifikátor pro strukturu deskriptoru vlastností.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **PSIDV**.

PDSUP (10místné celé číslo se znaménkem)

Toto pole popisuje, jaká úroveň podpory pro vlastnost zprávy je vyžadována pro správce front, aby mohla být zpráva obsahující tuto vlastnost vložena do fronty. Toto platí pouze pro vlastnosti definované pomocí IBM MQ; podpora všech ostatních vlastností je volitelná.

Pole je automaticky nastaveno na správnou hodnotu, pokud správce front zná vlastnost IBM MQ-defined. Není-li vlastnost rozpoznána, je přiřazen PDSUPO. Když správce front obdrží zprávu obsahující vlastnost definovanou proměnnou IBM MQ, kterou správce front rozpozná jako nesprávnou. správce front opraví hodnotu pole *PDSUP* .

Při nastavení vlastnosti IBM MQ-defined pomocí volání MQSETMP na manipulátoru zprávy, kde byla nastavena volba CMNOVA, se *PDSUP* stane vstupním polem. To umožňuje aplikaci vložit vlastnost IBM MQ-defined se správnou hodnotou, kde vlastnost není podporována připojeným správcem front, ale kde je zpráva určena ke zpracování v jiném správci front.

Hodnota PDSUPO je vždy přiřazena k vlastnostem, které nejsou IBM MQdefinovanými vlastnostmi.

Při použití volání MQSETMP na manipulátoru zprávy, kde je nastavena volba CMNOVA, je vrácena jedna z následujících hodnot voláním MQINQMP nebo lze zadat jednu z těchto hodnot:

PDSUPO
Vlastnost je přijata správcem front i v případě, že není podporována. Vlastnost lze zrušit tak, aby zpráva směřovala do správce front, který nepodporuje vlastnosti zprávy. Tato hodnota je také přiřazena k vlastnostem, které nejsou IBM MQ-defined.

PDSUPR
Podpora pro vlastnost je povinná. Zpráva je odmítnuta správcem front, který nepodporuje vlastnost IBM MQ-defined. Volání MQPUT nebo MQPUT1 se nezdaří s kódem dokončení CCFAIL a kódem příčiny RC2490.

PDSUPL
Zpráva je odmítnuta správcem front, který nepodporuje vlastnost IBM MQ-defined, pokud je zpráva určena pro lokální frontu. Volání MQPUT nebo MQPUT1 se nezdaří s kódem dokončení CCFAIL a kódem příčiny RC2490.

Volání MQPUT nebo MQPUT1 je úspěšné, pokud je zpráva určena pro vzdáleného správce front.

Jedná se o výstupní pole pro volání MQINQMP a vstupní pole pro volání MQSETMP, pokud byl manipulátor zprávy vytvořen s nastavenou volbou CMNOVA. Počáteční hodnota tohoto pole je PDSUPO.

PDVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury; hodnota musí být:

PDVER1
Version-1 struktura deskriptoru vlastností.

Následující konstanta určuje číslo verze aktuální verze:

PDVERC
Aktuální verze struktury deskriptoru vlastností.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **PDVER1**.

Počáteční hodnoty

Tabulka 715. Pole v MQPD

Název pole	Název konstanty	Hodnota konstanty
PDSID	PDSIDV	'PD'
PDVER	PDVER1	1
PDOPT	PDNONE	0
PDSUP	PDSUPO	0
PDCT	PDNOC	0
PDCPYOPT	COPDEF	0

Deklarace RPG

```
D* MQDMHO Structure
D*
D*
D* Structure identifier
D  DMSID                  1      4    INZ('DMHO')
```

```
     D*
     D* Structure version number
     D  DMVER                  5      8I 0 INZ(1)
     D*
     D* Options that control the action of MQDLTMH
     D  DMOPT                  9     12I 0 INZ(0)
```

IBM i MQPMO (Vložit-volby zpráv) na IBM i

Struktura MQPMO umožňuje aplikaci určit volby, které řídí způsob umisťování zpráv do front nebo jejich publikování do témat.

Přehled

Účel
Struktura je vstupní/výstupní parametr volání MQPUT a MQPUT1 .

Verze
Aktuální verze MQPMO je PMVER2. Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi MQPMO, která je podporována prostředím, ale s počáteční hodnotou pole *PMVER* nastavenou na hodnotu PMVER1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , musí aplikace nastavit pole *PMVER* na číslo verze požadované verze.

Znaková sada a kódování
Data v MQPMO musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , musí být struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1168
- "Počáteční hodnoty" na stránce 1181
- "Deklarace RPG" na stránce 1182

Pole

Struktura MQPMO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

PMCT (10místné celé číslo se znaménkem)

Popisovač objektu vstupní fronty.

Je-li zadána hodnota PMPASI nebo PMPASA, musí toto pole obsahovat manipulátor vstupní fronty, ze kterého jsou převzaty informace o kontextu, které mají být přidruženy k vkládané zprávě.

Nejsou-li zadány hodnoty PMPASI a PMPASA, bude toto pole ignorováno.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

PMIDC (10místné celé číslo se znaménkem)

Počet zpráv, které se nepodařilo odeslat.

Jedná se o počet zpráv, které nebylo možné odeslat do front v rozdělovníku. Počet zahrnuje fronty, jejichž otevření se nezdařilo, a fronty, které byly úspěšně otevřeny, ale pro které operace vložení selhala. Toto pole je také nastaveno při vkládání zprávy do jedné fronty, která není v rozdělovníku.

Poznámka: Toto pole je nastaveno pouze v případě, že parametr **CMPCOD** ve volání MQPUT nebo MQPUT1 je CCOK nebo CCWARN; není nastaveno, pokud je parametr **CMPCOD** CCFAIL.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole není nastaveno, pokud je *PMVER* menší než PMVER2.

PMKDC (10místné celé číslo se znaménkem)

Počet zpráv úspěšně odeslaných do lokálních front.

Jedná se o počet zpráv, které aktuální volání MQPUT nebo MQPUT1 úspěšně odeslalo do front v distribučním seznamu, které jsou lokálními frontami. Tento počet nezahrnuje zprávy odeslané do front, které se interpretují do vzdálených front (i když se lokální přenosová fronta používá na počátku k uložení zprávy). Toto pole je také nastaveno při vkládání zprávy do jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole není nastaveno, pokud je *PMVER* menší než PMVER2.

PMOPT (10místné celé číslo se znaménkem)

Volby, které řídí akci MQPUT a MQPUT1.

Lze zadat libovolnou z následujících možností nebo žádnou z nich. Je-li vyžadováno více hodnot, lze je přidat (nepřidávejte stejnou konstantu více než jednou). Kombinace, které nejsou platné, jsou uvedeny; všechny ostatní kombinace jsou platné.

Volby publikování: Následující volby řídí způsob publikování zpráv do tématu.

PMSRTO

Odběratelům nejsou předávány žádné informace vyplněné do polí MDRQ a MDRM deskriptoru MQMD této publikace. Je-li tato volba použita s volbou sestavy, která vyžaduje Q ReplyTo, volání selže s volbou RC2027 .

PMRET

Odesílaná publikace má být zachována správcem front. To umožňuje odběrateli vyžádat si kopii tohoto publikování po jeho publikování pomocí volání MQSUBRQ. Umožňuje také odeslání publikace do aplikací, které si předávají své předplatné po provedení této publikace, pokud se nerozhodnou, že nebudou zasílány prostřednictvím volby SONEWP. Pokud je aplikaci odesláno publikování, které bylo zachováno, je označeno vlastností zprávy mq.IsRetained tohoto publikování.

V každém uzlu stromu témat lze zachovat pouze jedno publikování. To znamená, že pokud již existuje zachovaná publikace pro toto téma, publikovaná jinou aplikací, je nahrazena touto publikací. Proto je lepší vyhnout se tomu, aby více než jeden vydavatel uchovávající zprávy na stejné téma.

Pokud odběratel požaduje zachovaná publikování, může použitý odběr v tématu obsahovat zástupný znak. V takovém případě se může počet zachovaných publikování shodovat (v různých uzlech ve stromu témat) a do žádající aplikace může být odesláno několik publikování. Další podrobnosti viz popis volání “MQSUBRQ-Požadavek na odběr” na stránce 796 .

Je-li použita tato volba a publikování nelze zachovat, zpráva se nepublikuje a volání selže s volbou RC2479 .

Volby syncpoint: Následující volby se týkají účasti volání MQPUT nebo MQPUT1 v rámci pracovní jednotky:

PMSYP

Vložte zprávu s ovládacím prvkem synchronizačního bodu.

Požadavek je pracovat v rámci běžných protokolů jednotky práce. Zpráva není viditelná mimo jednotku práce, dokud není jednotka práce potvrzena. Pokud je jednotka práce odvolána, zpráva se odstraní.

Pokud tato volba a PMNSYP nejsou uvedeny, požadavek na vložení není v rámci pracovní jednotky.

PMSYP nesmí být uveden s PMNSYP.

PMNSYP

Vložit zprávu bez ovládacího prvku synchronizačního bodu.

Požadavek je pracovat mimo normální protokoly jednotky práce. Zpráva je k dispozici okamžitě a nelze ji odstranit zálohováním pracovní jednotky.

Pokud tato volba a PMSYP nejsou uvedeny, požadavek na vložení není v rámci pracovní jednotky.

PMNSYP nesmí být uveden s PMSYP.

Volby identifikátoru zprávy a identifikátoru korelace: Následující volby vyžadují, aby správce front vygeneroval nový identifikátor zprávy nebo identifikátor korelace:

PMNMID
Vygenerujte nový identifikátor zprávy.

Tato volba způsobí, že správce front nahradí obsah pole *MDMID* v MQMD novým identifikátorem zprávy. Tento identifikátor zprávy je odeslán spolu se zprávou a vrácen aplikaci při výstupu volání MQPUT nebo MQPUT1 .

Tuto volbu lze také zadat, když je zpráva vkládána do distribučního seznamu; podrobnosti viz popis pole *PRMID* ve struktuře MQPMR.

Použití této volby zbavuje aplikaci potřeby resetovat pole *MDMID* na hodnotu MINONE před každým voláním MQPUT nebo MQPUT1 .

PMNCID
Vygenerujte nový identifikátor korelace.

Tato volba způsobí, že správce front nahradí obsah pole *MDCID* v MQMD novým identifikátorem korelace. Tento identifikátor korelace je odeslán spolu se zprávou a vrácen aplikaci při výstupu volání MQPUT nebo MQPUT1 .

Tuto volbu lze také zadat, když je zpráva vkládána do distribučního seznamu; podrobnosti viz popis pole *PRCID* ve struktuře MQPMR.

PMNCID je užitečný v situacích, kdy aplikace vyžaduje jedinečný identifikátor korelace.

Volby skupiny a segmentu: Následující volba se týká zpracování zpráv ve skupinách a segmentech logických zpráv. Tyto definice mohou být užitečné při porozumění volbě:

Fyzická zpráva
Jedná se o nejmenší jednotku informací, kterou lze umístit do fronty nebo z ní odebrat. Často odpovídá informacím zadaným nebo načteným v rámci jednoho volání MQPUT, MQPUT1nebo MQGET. Každá fyzická zpráva má svůj vlastní deskriptor zpráv (MQMD). Obecně se fyzické zprávy rozlišují podle různých hodnot pro identifikátor zprávy (pole*MDMID* v deskriptoru MQMD), ačkoli to správce front nevynucuje.

Logická zpráva
Toto je jedna jednotka informací o aplikaci. Při absenci systémových omezení by logická zpráva byla stejná jako fyzická zpráva. Jsou-li však logické zprávy velké, mohou být kvůli systémovým omezením vhodné nebo nezbytné rozdělit logickou zprávu do dvou nebo více fyzických zpráv, které se nazývají *segmenty*.

Logická zpráva, která byla segmentována, se skládá ze dvou nebo více fyzických zpráv, které mají stejný nenulový identifikátor skupiny (pole*MDGID* v deskriptoru MQMD) a stejné pořadové číslo zprávy (pole*MDSEQ* v deskriptoru MQMD). Segmenty jsou odlišeny odlišnými hodnotami pro posunutí segmentu (pole*MDOFF* v MQMD), které poskytuje posunutí dat ve fyzické zprávě od začátku dat v logické zprávě. Protože každý segment je fyzickou zprávou, segmenty v logické zprávě mají obvykle různé identifikátory zpráv.

Logická zpráva, která nebyla segmentována, ale pro kterou byla segmentace povolena odesílající aplikací, má také nenulový identifikátor skupiny, ačkoli v tomto případě existuje pouze jedna fyzická zpráva s tímto identifikátorem skupiny, pokud logická zpráva nepatří do skupiny zpráv. Logické zprávy, pro které byla segmentace blokována odesílající aplikací, mají identifikátor skupiny s hodnotou null (GINONE), pokud logická zpráva nepatří do skupiny zpráv.

Skupina zpráv
Jedná se o sadu jedné nebo více logických zpráv, které mají stejný nenulový identifikátor skupiny. Logické zprávy ve skupině jsou rozlišeny různými hodnotami pro pořadové číslo zprávy, což je celé číslo v rozsahu 1 až n, kde n je počet logických zpráv ve skupině. Je-li jedna nebo více logických zpráv segmentováno, je ve skupině více než n fyzických zpráv.

PMLOGO
Zprávy ve skupinách a segmentech logických zpráv jsou uvedeny v logickém pořadí.

Tato volba informuje správce front o tom, jak aplikace vkládá zprávy do skupin a segmentů logických zpráv. Lze jej zadat pouze pro volání MQPUT; není platný pro volání MQPUT1 .

Je-li zadána hodnota PMLOGO, znamená to, že aplikace používá následná volání MQPUT pro:

- Vložila segmenty do každé logické zprávy kvůli zvýšení offsetu segmentu, počínaje 0, bez mezer.
- Před vložením segmentů do další logické zprávy vložte všechny segmenty do jedné logické zprávy.
- Vložila logické zprávy do každé skupiny zprávy, aby zvýšila pořadové číslo zprávy, počínaje 1, bez mezer.
- Před vložením logických zpráv do další skupiny zpráv vložte všechny logické zprávy do jedné skupiny zpráv.

Toto pořadí se nazývá "logické pořadí".

Vzhledem k tomu, že aplikace sdělila správci front, jak vkládá zprávy do skupin a segmentů logických zpráv, nemusí aplikace udržovat a aktualizovat informace o skupinách a segmentech pro každé volání MQPUT tak, jak to provádí správce front. Konkrétně to znamená, že aplikace nemusí nastavit pole *MDGID*, *MDSEQ*a *MDOFF* v deskriptoru MQMD, protože je správce front nastaví na příslušné hodnoty. Aplikace potřebuje nastavit pouze pole *MDMFL* v MQMD, aby označila, kdy zprávy patří do skupin nebo jsou segmenty logických zpráv, a aby označila poslední zprávu ve skupině nebo posledním segmentu logické zprávy.

Po spuštění skupiny zpráv nebo logické zprávy musí následná volání MQPUT určit příslušné příznaky MF* v souboru *MDMFL* v deskriptoru MQMD. Pokud se aplikace pokusí vložit zprávu, která není ve skupině, když existuje neukončená skupina zpráv, nebo vložit zprávu, která není segmentem, když existuje neukončená logická zpráva, volání selže s kódem příčiny RC2241 nebo RC2242 , podle potřeby. Správce front však zachová informace o aktuální skupině zpráv nebo aktuální logické zprávě a aplikace je může ukončit odesláním zprávy (pravděpodobně bez dat zprávy aplikace) s odpovídajícím určením MFLMIG nebo MFLSEG před opětovným zadáním volání MQPUT pro vložení zprávy, která není ve skupině nebo není segmentem.

Tabulka 716 na stránce 1172 zobrazuje platné kombinace voleb a příznaků a hodnoty polí *MDGID*, *MDSEQ*a *MDOFF* , které správce front používá v jednotlivých případech. Kombinace voleb a příznaků, které nejsou zobrazeny v tabulce, nejsou platné. Sloupce v tabulce mají následující význam:

PROTOKOLNÍ ZÁZNAM
Označuje, zda je ve volání uvedena volba PMLOGO.

MIG
Označuje, zda je ve volání uvedena volba MFMIG nebo MFLMIG.

Skupina SEG
Označuje, zda je ve volání uvedena volba MFSEG nebo MFLSEG.

SEG v pořádku
Označuje, zda je ve volání uvedena volba MFSEGA.

Aktuální grp
Označuje, zda před voláním existuje aktuální skupina zpráv.

Aktuální zpráva protokolu
Označuje, zda před voláním existuje aktuální logická zpráva.

Ostatní sloupce
Zobrazit hodnoty, které správce front používá. "Předchozí" označuje hodnotu použitou pro pole v předchozí zprávě pro popisovač fronty.

PMRLOC
Uvádí, že PMRQN ve struktuře MQPMO musí být dokončen s názvem lokální fronty, do které je zpráva skutečně vložena. Název ResolvedQMgrje podobně doplněn názvem lokálního správce front, který je hostitelem lokální fronty. Co to znamená, viz OORLOQ. Je-li uživatel autorizován pro vložení do fronty, má požadované oprávnění k zadání tohoto příznaku ve volání MQPUT. Není potřeba žádné speciální oprávnění.

Tabulka 716. Volby MQPUT související se zprávami ve skupinách a segmentech logických zpráv

Volby, které zadáte				**Stav skupiny a zprávy protokolu před voláním**		**Hodnoty, které správce front používá**		
LOG ORD	**MIG**	**SEG**	**SEG OK**	**Cur grp**	**Aktuální zpráva protokolu**	*MDGID*	*MDSEQ*	*MDOFF*
Ano	Ne	Ne	Ne	Ne	Ne	GINONE	1	0
Ano	Ne	Ne	Ano	Ne	Ne	ID nové skupiny	1	0
Ano	Ne	Ano	YES nebo NO	Ne	Ne	ID nové skupiny	1	0
Ano	Ne	Ano	YES nebo NO	Ne	Ano	ID předchozí skupiny	1	Předchozí offset + předchozí délka segmentu
Ano	Ano	YES nebo NO	YES nebo NO	Ne	Ne	ID nové skupiny	1	0
Ano	Ano	YES nebo NO	YES nebo NO	Ano	Ne	ID předchozí skupiny	Předchozí pořadové číslo + 1	0
Ano	Ano	Ano	YES nebo NO	Ano	Ano	ID předchozí skupiny	Předchozí pořadové číslo	Předchozí offset + předchozí délka segmentu
Ne	Ne	Ne	Ne	YES nebo NO	YES nebo NO	GINONE	1	0
Ne	Ne	Ne	Ano	YES nebo NO	YES nebo NO	Nové ID skupiny, pokud je GINONE, jinak hodnota v poli	1	0
Ne	Ne	Ano	YES nebo NO	YES nebo NO	YES nebo NO	Nové ID skupiny, pokud je GINONE, jinak hodnota v poli	1	Hodnota v poli

Tabulka 716. Volby MQPUT související se zprávami ve skupinách a segmentech logických zpráv (pokračování)

Volby, které zadáte				Stav skupiny a zprávy protokolu před voláním		Hodnoty, které správce front používá		
Ne	Ano	Ne	YES nebo NO	YES nebo NO	YES nebo NO	Nové ID skupiny, pokud je GINONE, jinak hodnota v poli	Hodnota v poli	0
Ne	Ano	Ano	YES nebo NO	YES nebo NO	YES nebo NO	Nové ID skupiny, pokud je GINONE, jinak hodnota v poli	Hodnota v poli	Hodnota v poli

Poznámka:

- PMLOGO není platný pro volání MQPUT1 .
- Pro pole *MDMID* generuje správce front nový identifikátor zprávy, je-li zadán parametr PMNMID nebo MINONE, a jinak použije hodnotu v poli.
- Pro pole *MDCID* generuje správce front nový identifikátor korelace, je-li uveden identifikátor PMNCID, a jinak použije hodnotu v poli.

Je-li zadána hodnota PMLOGO, správce front vyžaduje, aby všechny zprávy ve skupině a segmenty v logické zprávě byly vloženy se stejnou hodnotou do pole *MDPER* v deskriptoru MQMD, tj. všechny zprávy musí být trvalé nebo všechny musí být přechodné. Není-li tato podmínka splněna, volání MQPUT se nezdaří s kódem příčiny RC2185 .

Volba PMLOGO ovlivňuje jednotky práce takto:

- Je-li první fyzická zpráva ve skupině nebo logické zprávě vložena do pracovní jednotky, musí být všechny ostatní fyzické zprávy ve skupině nebo logické zprávě vloženy do pracovní jednotky, pokud je použita stejná obsluha fronty. Nicméně, nemusí být uvedeny v rámci stejné jednotky práce. To umožňuje rozdělit skupinu zpráv nebo logickou zprávu sestávající z mnoha fyzických zpráv na dvě nebo více po sobě jdoucích jednotek práce pro popisovač fronty.
- Není-li první fyzická zpráva ve skupině nebo logické zprávě vložena do pracovní jednotky, nemůže být žádná z ostatních fyzických zpráv ve skupině nebo logické zprávě vložena do pracovní jednotky, pokud je použita stejná obsluha fronty.

Nejsou-li tyto podmínky splněny, volání MQPUT se nezdaří s kódem příčiny RC2245 .

Je-li zadána hodnota PMLOGO, nesmí být hodnota MQMD zadaná ve volání MQPUT menší než hodnota MDVER2. Není-li tato podmínka splněna, volání se nezdaří s kódem příčiny RC2257 .

Není-li zadána hodnota PMLOGO, lze zprávy ve skupinách a segmentech logických zpráv řadit v libovolném pořadí a není nutné vkládat úplné skupiny zpráv ani úplné logické zprávy. Je odpovědností aplikace zajistit, aby pole *MDGID*, *MDSEQ*, *MDOFF*a *MDMFL* měla odpovídající hodnoty.

Jedná se o techniku, kterou lze použít k restartování skupiny zpráv nebo logické zprávy uprostřed po selhání systému. Když se systém restartuje, aplikace může nastavit pole *MDGID*, *MDSEQ*, *MDOFF*, *MDMFL*a *MDPER* na odpovídající hodnoty a pak vydat volání MQPUT s PMSYP nebo PMNSYP nastaveným jako *nezbytné*, ale bez uvedení PMLOGO. Je-li toto volání úspěšné, správce front zachová informace o skupině a segmentu a následná volání MQPUT používající tento manipulátor fronty mohou určit PMLOGO jako normální.

Informace o skupině a segmentu, které správce front uchovává pro volání MQPUT, jsou odděleny od informací o skupině a segmentu, které uchovává pro volání MQGET.

Pro každý daný popisovač fronty může aplikace směšovat volání MQPUT, která určují PMLOGO s voláními MQPUT, která nikoli, ale je třeba poznamenat následující body:

- Není-li zadána hodnota PMLOGO, každé úspěšné volání MQPUT způsobí, že správce front nastaví informace o skupině a segmentu pro manipulátor fronty na hodnoty určené aplikací. Tato volba nahradí existující informace o skupině a segmentu uchovávané správcem front pro manipulátor fronty.
- Není-li zadána hodnota PMLOGO, volání neselže, pokud existuje aktuální skupina zpráv nebo logická zpráva; volání však může být úspěšné s kódem dokončení CCWARN. Tabulka 717 na stránce 1174 zobrazuje různé případy, které mohou nastat. V těchto případech, pokud kód dokončení není CCOK, je kód příčiny jeden z následujících (podle potřeby):
  - RC2241
  - RC2242
  - RC2185
  - RC2245

  Poznámka: Správce front nekontroluje informace o skupině a segmentu pro volání MQPUT1 .

Tabulka 717. Výsledek, když volání MQPUT nebo MQCLOSE není konzistentní s informacemi o skupině a segmentu

Aktuální volání je	Předchozí volání bylo MQPUT s PMLOGO	Předchozí volání bylo MQPUT bez PMLOGO
MQPUT s PMLOGO	CCFAIL	CCFAIL
MQPUT bez PMLOGO	CCWARN (varování)	CCOK
MQCLOSE s neukončenou skupinou nebo logickou zprávou	CCWARN (varování)	CCOK

Aplikacím, které prostě chtějí umístit zprávy a segmenty v logickém pořadí, se doporučuje zadat PMLOGO, protože se jedná o nejjednodušší volbu, kterou lze použít. Tato volba zbavuje aplikaci potřeby spravovat informace o skupině a segmentu, protože správce front tyto informace spravuje. Specializované aplikace však mohou vyžadovat větší kontrolu, než poskytuje volba PMLOGO, a toho lze dosáhnout neuvedením této volby. Pokud se tak stane, musí aplikace zajistit, aby pole *MDGID*, *MDSEQ*, *MDOFF*a *MDMFL* v deskriptoru MQMD byla správně nastavena před každým voláním MQPUT nebo MQPUT1 .

Například aplikace, která chce předávat přijaté fyzické zprávy bez ohledu na to, zda jsou tyto zprávy ve skupinách nebo segmentech logických zpráv, nesmí určovat PMLOGO. Existují pro to dva důvody:

- Pokud jsou zprávy načteny a vloženy do pořadí, uvedení PMLOGO způsobí, že ke zprávám bude přiřazen nový identifikátor skupiny, což může ztížit nebo znemožnit původci zpráv korelaci všech odpovědí nebo zpráv sestav, které jsou výsledkem skupiny zpráv.
- V komplexní síti s více cestami mezi odesílajícími a přijímajícími správci front mohou fyzické zprávy přicházet mimo pořadí. Neuvedete-li PMLOGO a odpovídající GMLOGO ve volání MQGET, může aplikace předání načíst a předat každou fyzickou zprávu ihned po jejím doručení, aniž by musela čekat na další v logickém pořadí.

Aplikace, které generují zprávy sestav pro zprávy ve skupinách nebo segmentech logických zpráv, také nesmí při vkládání zprávy sestavy určovat PMLOGO.

PMLOGO lze uvést s jakoukoli jinou volbou PM*.

Volby kontextu: Následující volby řídí zpracování kontextu zprávy:

PMNOC
Ke zprávě nemá být přidružen žádný kontext.

Identita i původní kontext jsou nastaveny tak, aby neoznačovaly žádný kontext. To znamená, že pole kontextu v deskriptoru MQMD jsou nastavena na:

- Mezery pro znaková pole
- Hodnoty Null pro bajtová pole
- Nuly pro číselná pole

PMDEFC
Použít výchozí kontext.

Zpráva má mít k sobě přidružené výchozí informace o kontextu, a to jak pro identitu, tak pro původ. Správce front nastaví pole kontextu v deskriptoru zprávy následujícím způsobem:

Tabulka 718. Výchozí hodnoty informací o kontextu pro pole MQMD

Pole v deskriptoru MQMD	Použitá hodnota
MDUID	Určuje se z prostředí, je-li to možné; jinak se nastaví na mezery.
MDACC	Určuje se z prostředí, je-li to možné; jinak nastavte na hodnotu ACNONE.
MDAID	Nastavte na mezery.
MDPAT	Určeno z prostředí.
MDPAN	Určuje se z prostředí, je-li to možné; jinak se nastaví na mezery.
MDPD	Nastavit na datum, kdy je zpráva vložena.
MDPT	Nastavit na čas, kdy je zpráva vložena.
MDAOD	Nastavte na mezery.

Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Jedná se o výchozí akci, pokud nejsou zadány žádné kontextové volby.

PMPASI
Předejte kontext identity z manipulátoru vstupní fronty.

Zpráva má mít k sobě přidružené informace o kontextu. Kontext identity je převzat z popisovače fronty uvedeného v poli *PMCT* . Informace o původním kontextu jsou generovány správcem front stejným způsobem jako pro PMDEFC (hodnoty viz předchozí tabulka). Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Pro volání MQPUT musí být fronta otevřena s volbou OOPASI (nebo s volbou, která z ní vyplývá). Pro volání MQPUT1 se provádí stejná kontrola autorizace jako pro volání MQOPEN s volbou OOPASI.

PMPASA
Předejte veškerý kontext z manipulátoru vstupní fronty.

Zpráva má mít k sobě přidružené informace o kontextu. Identita i původní kontext jsou převzaty z popisovače fronty uvedeného v poli *PMCT* . Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Pro volání MQPUT musí být fronta otevřena s volbou OOPASA (nebo s volbou, která z ní vyplývá). Pro volání MQPUT1 se provádí stejná kontrola autorizace jako pro volání MQOPEN s volbou OOPASA.

PMSETI
Nastavte kontext identity z aplikace.

Zpráva má mít k sobě přidružené informace o kontextu. Aplikace určuje kontext identity ve struktuře MQMD. Informace o původním kontextu jsou generovány správcem front stejným způsobem jako pro PMDEFC (hodnoty viz předchozí tabulka). Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Pro volání MQPUT musí být fronta otevřena s volbou OOSETI (nebo s volbou, která ji označuje). Pro volání MQPUT1 se provádí stejná kontrola autorizace jako pro volání MQOPEN s volbou OOSETI.

PMSETA
Nastavte celý kontext z aplikace.

Zpráva má mít k sobě přidružené informace o kontextu. Aplikace určuje identitu a původní kontext ve struktuře MQMD. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Pro volání MQPUT musí být fronta otevřena s volbou OOSETA. Pro volání MQPUT1 se provádí stejná kontrola autorizace jako pro volání MQOPEN s volbou OOSETA.

Lze zadat pouze jednu z voleb kontextu PM*. Není-li uvedena žádná z těchto voleb, předpokládá se PMDEFC.

Vložit typy odpovědí. Následující volby řídí odezvu vrácenou na volání MQPUT nebo MQPUT1 . Můžete zadat pouze jednu z těchto voleb. Pokud nejsou zadány PMARES a PMSRES, předpokládá se PMRASQ nebo PMRAST.

PMARES

Volba PMARES požaduje, aby byla operace MQPUT nebo MQPUT1 dokončena, aniž by aplikace čekala na dokončení volání správcem front. Použití této volby může zlepšit výkon systému zpráv, zejména pro aplikace používající vazby klienta. Aplikace může pomocí příkazu MQSTAT pravidelně kontrolovat, zda během předchozích asynchronních volání nedošlo k chybě.

Při použití této volby je zaručeno, že v deskriptoru MQMD budou vyplněna pouze následující pole;

- MDAID
- MDPAT
- MDPAN
- MDAOD (Předávkování)

Kromě toho, pokud jsou jako volby uvedeny jedna nebo obě volby PMNMID nebo PMNCID, jsou vráceny také MDMID a MDCID. (PMNMID lze implicitně zadat zadáním prázdného pole MDMID).

Vyplní se pouze dříve uvedená pole. Další informace, které by normálně byly vráceny ve struktuře MQMD nebo MQPMO, nejsou definovány.

Při vyžádání asynchronní odezvy vložení pro MQPUT nebo MQPUT1nemusí CMPCOD a REASON pro CCOK a RCNONE nutně znamenat, že zpráva byla úspěšně vložena do fronty. Při vývoji aplikace MQI, která používá asynchronní odezvu vložení a vyžaduje potvrzení, že byly zprávy vloženy do fronty, byste měli zkontrolovat kódy CMPCOD i REASON z operací vložení a také použít MQSTAT k dotazování na informace o asynchronní chybě.

Ačkoli úspěch nebo selhání jednotlivých volání MQPUT/MQPUT1 nemusí být vráceno okamžitě, první chybu, která se vyskytla v rámci asynchronního volání, lze určit v pozdější fázi prostřednictvím volání MQSTAT.

Pokud se nezdaří doručení trvalé zprávy v synchronizačním bodu s použitím asynchronní odezvy vložení a pokusíte se potvrdit transakci, potvrzení se nezdaří a transakce se odvolá s kódem dokončení CCFAIL a příčinou je RC2003 . Aplikace může provést volání MQSTAT s cílem určit příčinu předchozího selhání MQPUT nebo MQPUT1 .

PMSRES
Zadáním této hodnoty pro volbu vložení ve struktuře MQPMO zajistíte, že operace MQPUT nebo MQPUT1 bude vždy zadána synchronně. Je-li operace úspěšná, jsou dokončena všechna pole v MQMD a MQPMO. Poskytuje se k zajištění synchronní odezvy bez ohledu na výchozí hodnotu odezvy vložení definovanou ve frontě nebo objektu tématu.

PMRASQ
Je-li tato hodnota určena pro volání MQPUT, je použitý typ odezvy vložení převzat z hodnoty DEFPRESP určené ve frontě při jejím otevření aplikací. Pokud je aplikace klienta připojena ke správci front na úrovni starší než IBM WebSphere MQ 7.0, chová se, jako by byla zadána hodnota PMSRES.

Je-li tato volba zadána pro volání MQPUT1 , hodnota DEFPRESP z definice fronty se nepoužije. Pokud volání MQPUT1 používá PMSYP, chová se jako pro PMARES a pokud používá PMNSYP, chová se jako pro PMSRES.

PMRAST
Jedná se o synonymum pro PMRASQ pro použití s objekty tématu.

Další volby: Následující volby řídí kontrolu autorizace a to, co se stane, když je správce front uveden do klidového stavu:

PMALTU
Ověřit s uvedeným identifikátorem uživatele.

To znamená, že pole *ODAU* v parametru **OBJDSC** volání MQPUT1 obsahuje identifikátor uživatele, který má být použit k ověření oprávnění pro vložení zpráv do fronty. Volání může být úspěšné pouze v případě, že je tento produkt *ODAU* autorizován k otevření fronty s uvedenými volbami, bez ohledu na to, zda je k tomu autorizován identifikátor uživatele, pod kterým je aplikace spuštěna. (Toto neplatí pro zadané kontextové volby, které jsou však vždy kontrolovány proti identifikátoru uživatele, pod kterým je aplikace spuštěna.)

Tato volba je platná pouze s voláním MQPUT1 .

PMFIQ
Selhání při uvedení správce front do klidového stavu.

Tato volba vynutí selhání volání MQPUT nebo MQPUT1 , pokud je správce front ve stavu uvedení do klidového stavu.

Volání vrátí kód dokončení CCFAIL s kódem příčiny RC2161 .

Výchozí volba: Pokud není vyžadována žádná z dříve popsaných voleb, lze použít následující volbu:

PMNONE
Nejsou uvedeny žádné volby.

Tuto hodnotu lze použít k označení, že nebyly zadány žádné další volby; všechny volby předpokládají své výchozí hodnoty. PMNONE je definován jako pomocná programová dokumentace; není zamýšleno, aby se tato volba používala s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Toto je vstupní pole. Počáteční hodnota pole *PMOPT* je PMNONE.

PMPRF (10místné celé číslo se znaménkem)

Příznaky označující, která pole MQPMR jsou přítomna.

Toto pole obsahuje příznaky, které musí být nastaveny tak, aby označovaly, která pole MQPMR jsou přítomna v záznamech vložených zpráv poskytnutých aplikací. *PMPRF* se používá pouze v případě, že je zpráva vkládána do rozdělovníku. Pole je ignorováno, pokud je hodnota *PMREC* nula, nebo jsou hodnoty *PMPRO* i *PMPRP* nula.

Pro pole, která jsou přítomna, používá správce front pro každé místo určení hodnoty z polí v odpovídajícím záznamu vkládané zprávy. Pro pole, která chybí, použije správce front hodnoty ze struktury MQMD.

Lze uvést jeden nebo více následujících příznaků, které označují, která pole jsou přítomna v záznamech vložených zpráv:

PFMID
Pole identifikátoru zprávy je přítomno.

PFCID
Pole identifikátoru korelace je přítomno.

PFGID
Pole identifikátoru skupiny je přítomno.

PFFB
Pole zpětné vazby je přítomno.

PFACC
Pole tokenu evidence je přítomno.

Je-li uveden tento příznak, musí být v poli *PMOPT* uvedeno buď PMSETI, nebo PMSETA; není-li tato podmínka splněna, volání selže s kódem příčiny RC2158 .

Nejsou-li uvedena žádná pole MQPMR, lze zadat následující:

PFNONE
Nejsou přítomna žádná pole záznamu vložená zpráva.

Je-li určena tato hodnota, *PMREC* musí být nula nebo obě hodnoty *PMPRO* a *PMPRP* musí být nula.

PFNONE je definován jako pomocná programová dokumentace. Není zamýšleno, aby se tato konstanta používala s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Pokud *PMPRF* obsahuje neplatné příznaky nebo jsou poskytnuty záznamy vkládané zprávy, ale *PMPRF* má hodnotu PFNONE, volání selže s kódem příčiny RC2158 .

Toto je vstupní pole. Počáteční hodnota tohoto pole je PFNONE. Toto pole je ignorováno, pokud je hodnota *PMVER* menší než hodnota PMVER2.

PMPRO (celé číslo se znaménkem 10 číslic)

Posun prvního vloženého záznamu zprávy od začátku MQPMO.

Jedná se o posun v bajtech prvního záznamu zprávy vložení MQPMR od začátku struktury MQPMO. Posun může být kladný nebo záporný. *PMPRO* se používá pouze v případě, že je zpráva vkládána do rozdělovníku. Je-li hodnota *PMREC* nula, pole se ignoruje.

Když je zpráva vkládána do distribučního seznamu, lze zadat pole jednoho nebo více záznamů vložených zpráv MQPMR, aby bylo možné určit určité vlastnosti zprávy pro každé místo určení jednotlivě; tyto vlastnosti jsou:

- Identifikátor zprávy
- identifikátor korelace
- Identifikátor skupiny
- hodnota zpětné vazby
- Token evidence

Není nutné uvádět všechny tyto vlastnosti, ale bez ohledu na zvolenou dílčí sadu musí být pole uvedena ve správném pořadí. Další podrobnosti viz popis struktury MQPMR.

Obvykle by měl být zadán stejný počet záznamů vložených zpráv, jako je počet záznamů objektů určených produktem MQOD při otevření distribučního seznamu; každý záznam vložených zpráv poskytuje vlastnosti zpráv pro frontu určenou odpovídajícím záznamem objektu. Fronty v distribučním seznamu, které se nepodařilo otevřít, musí mít stále přidělené záznamy zpráv na příslušných pozicích v poli, ačkoli vlastnosti zprávy jsou v tomto případě ignorovány.

Je možné, aby se počet záznamů vložených zpráv lišil od počtu záznamů objektů. Pokud existuje méně záznamů vložených zpráv než záznamů objektů, vlastnosti zpráv pro místa určení, která nemají vloženy záznamy zpráv, jsou převzaty z odpovídajících polí v deskriptoru zpráv MQMD. Pokud existuje více záznamů vkládaných zpráv než záznamů objektů, přebytečné záznamy se nepoužívají (i když musí být stále možné k nim přistupovat). Záznamy vložených zpráv jsou volitelné, ale pokud jsou zadány, musí být *PMREC* z nich.

Záznamy vložených zpráv mohou být poskytnuty podobným způsobem jako záznamy objektů v produktu MQOD, a to buď uvedením odchylky v souboru *PMPRO*, nebo zadáním adresy v souboru *PMPRP* ; Podrobnosti o tom, jak to provést, viz pole *ODORO* popsané v části “MQOD (Object descriptor) na systému IBM i” na stránce 1154.

Nelze použít více než jeden z *PMPRO* a *PMPRP* ; volání selže s kódem příčiny RC2159 , pokud jsou obě nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *PMVER* menší než hodnota PMVER2.

PMPRP (ukazatel)

Adresa prvního vloženého záznamu zprávy.

Jedná se o adresu prvního záznamu zprávy vložení MQPMR. *PMPRP* se používá pouze v případě, že je zpráva vkládána do rozdělovníku. Je-li hodnota *PMREC* nula, pole se ignoruje.

Buď *PMPRP* , nebo *PMPRO* lze použít k uvedení záznamů vložených zpráv, ale ne obou; podrobnosti viz popis pole PMRRO . Není-li parametr *PMPRP* použit, musí být nastaven na nulový ukazatel nebo nulový počet bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null. Toto pole je ignorováno, pokud je hodnota *PMVER* menší než hodnota PMVER2.

PMREC (10místné celé číslo se znaménkem)

Počet přítomných záznamů vložených zpráv nebo záznamů odpovědí.

Jedná se o počet záznamů vložených zpráv MQPMR nebo záznamů odpovědí MQRR, které byly poskytnuty aplikací. Toto číslo může být větší než nula pouze v případě, že je zpráva vkládána do rozdělovníku. Záznamy vložených zpráv a záznamy odpovědí jsou volitelné-aplikace nemusí poskytovat žádné záznamy, nebo se může rozhodnout poskytnout záznamy pouze jednoho typu. Pokud však aplikace poskytuje záznamy obou typů, musí poskytnout záznamy *PMREC* každého typu.

Hodnota *PMREC* nemusí být stejná jako počet cílů v rozdělovníku. Pokud je zadáno příliš mnoho záznamů, přebytečné záznamy se nepoužívají; pokud je zadáno příliš málo záznamů, použijí se výchozí hodnoty pro vlastnosti zpráv pro ta místa určení, která nevložila záznamy zpráv (viz *PMPRO* později v tomto tématu).

Pokud je *PMREC* menší než nula nebo je větší než nula, ale zpráva není vložena do rozdělovníku, volání selže s kódem příčiny RC2154 .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *PMVER* menší než hodnota PMVER2.

PMRMN (řetězec 48bajtových znaků)

Vyřešený název cílového správce front.

Jedná se o název cílového správce front po provedení překladu názvů lokálním správcem front. Vrácený název je název správce front, který vlastní frontu identifikovanou pomocí *PMRQN*, a může se jednat o název lokálního správce front.

Pokud je *PMRQN* sdílená fronta, kterou vlastní skupina sdílení front, do které lokální správce front patří, *PMRMN* je název skupiny sdílení front. Pokud je fronta vlastněna jinou skupinou sdílení front, může být *PMRQN* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (povaha vrácené hodnoty je určena definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou; pokud je objekt distribučním seznamem nebo tématem, vrácená hodnota není definována.

Toto je výstupní pole. Délka tohoto pole je dána hodnotou LNQMN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

PMRQN (48bajtový znakový řetězec)

Vyřešený název cílové fronty.

Jedná se o název cílové fronty po provedení rozlišování názvů lokálním správcem front. Vrácený název je název fronty, která existuje ve správci front identifikovaném pomocí *PMRMN*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou; pokud je objekt distribučním seznamem nebo tématem, vrácená hodnota není definována.

Toto je výstupní pole. Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

PMRRO (10místné celé číslo se znaménkem)

Posunutí prvního záznamu odpovědi od začátku MQPMO.

Jedná se o posun v bajtech prvního záznamu odpovědi MQRR od začátku struktury MQPMO. Posun může být kladný nebo záporný. *PMRRO* se používá pouze v případě, že je zpráva vkládána do rozdělovníku. Je-li hodnota *PMREC* nula, pole se ignoruje.

Když je zpráva vkládána do distribučního seznamu, lze poskytnout pole jednoho nebo více záznamů odpovědí MQRR, aby bylo možné identifikovat fronty, do kterých nebyla zpráva úspěšně odeslána (pole*RRCC* v MQRR), a příčinu každého selhání (pole*RRREA* v MQRR). Je možné, že zpráva nebyla odeslána buď proto, že se nezdařilo otevřít frontu, nebo proto, že se nezdařila operace vložení. Správce front nastaví záznamy odpovědí pouze v případě, že je výsledek volání smíšený (to znamená, že některé zprávy byly úspěšně odeslány, zatímco jiné selhaly, nebo všechny selhaly, ale z různých důvodů); kód příčiny RC2136 z volání označuje tento případ. Pokud se stejný kód příčiny vztahuje na všechny fronty, vrátí se tato příčina v parametru **REASON** volání MQPUT nebo MQPUT1 a záznamy odezvy nejsou nastaveny.

Obvykle by měl být zadán stejný počet záznamů odpovědí, jako je počet záznamů objektů určených produktem MQOD při otevření distribučního seznamu. V případě potřeby je každý záznam odezvy nastaven na kód dokončení a kód příčiny pro vložení do fronty určené odpovídajícím záznamem objektu. Fronty v distribučním seznamu, které se nepodařilo otevřít, musí mít stále záznamy odpovědí přidělené na příslušných pozicích v poli, i když jsou nastaveny na kód dokončení a kód příčiny vyplývající z operace otevření, spíše než na operaci vložení.

Počet záznamů odpovědí se může lišit od počtu záznamů objektů. Pokud je méně záznamů odpovědí než záznamů objektů, nemusí být možné, aby aplikace identifikovala všechna místa určení, pro která operace vložení selhala, nebo příčiny selhání. Pokud existuje více záznamů odpovědí než záznamů objektů, přebytečné záznamy se nepoužívají (i když musí být stále možné k nim přistupovat). Záznamy odpovědí jsou volitelné, ale pokud jsou zadány, musí z nich být *PMREC* .

Záznamy odezvy mohou být poskytnuty podobným způsobem jako záznamy objektů v produktu MQOD, a to buď uvedením odchylky v souboru *PMRRO*, nebo uvedením adresy v souboru *PMRRP* ; Podrobnosti o tom, jak to provést, viz pole *ODORO* popsané v části “MQOD (Object descriptor) na systému IBM i” na stránce 1154. Nelze však použít více než jeden z *PMRRO* a *PMRRP* ; volání selže s kódem příčiny RC2156 , pokud jsou obě nenulové.

Pro volání MQPUT1 musí být toto pole nula. Důvodem je skutečnost, že informace o odezvě (je-li požadována) jsou vráceny v záznamech odezvy určených deskriptorem objektu MQOD.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud je hodnota *PMVER* menší než hodnota PMVER2.

PMRRP (ukazatel)

Adresa prvního záznamu odpovědi.

Jedná se o adresu prvního záznamu odpovědi MQRR. *PMRRP* se používá pouze v případě, že je zpráva vkládána do rozdělovníku. Je-li hodnota *PMREC* nula, pole se ignoruje.

K zadání záznamů odpovědí lze použít buď *PMRRP* , nebo *PMRRO* , ale ne obojí; podrobnosti viz popis pole PMRRO . Není-li parametr *PMRRP* použit, musí být nastaven na nulový ukazatel nebo nulový počet bajtů.

Pro volání MQPUT1 musí být toto pole ukazatelem Null nebo nulovým počtem bajtů. Důvodem je skutečnost, že informace o odezvě (je-li požadována) jsou vráceny v záznamech odezvy určených deskriptorem objektu MQOD.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null. Toto pole je ignorováno, pokud je hodnota *PMVER* menší než hodnota PMVER2.

PMSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

PMSIDV
Identifikátor pro strukturu voleb vložení zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je PMSIDV.

PMSL (MQLONG)

Úroveň odběru, na kterou se zaměřuje tato publikace.

Toto publikování obdrží pouze ty odběry s nejvyšší hodnotou *PMSL* , která je menší nebo rovna této hodnotě. Tato hodnota musí být v rozsahu nula až 9; nula je nejnižší úroveň.

Počáteční hodnota tohoto pole je 9.

PMTO (10místné celé číslo se znaménkem)

Vyhrazeno.

Toto je vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je -1.

PMUDC (10místné celé číslo se znaménkem)

Počet zpráv úspěšně odeslaných do vzdálených front.

Jedná se o počet zpráv, které aktuální volání MQPUT nebo MQPUT1 úspěšně odeslalo do front v distribučním seznamu, které se interpretují do vzdálených front. Zprávy, které správce front dočasně uchovává ve formě distribučního seznamu, se počítají jako počet jednotlivých cílů, které tyto distribuční seznamy obsahují. Toto pole je také nastaveno při vkládání zprávy do jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole není nastaveno, pokud je *PMVER* menší než PMVER2.

PMVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

PMVER1
Version-1 struktura voleb vložení zprávy.

PMVER2
Version-2 struktura voleb vložení zprávy.

Pole, která existují pouze v novější verzi struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

PMVERC
Aktuální verze struktury voleb vložení zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je PMVER1.

Počáteční hodnoty

Tabulka 719. Pole v MQPMO

Název pole	Název konstanty	Hodnota konstanty
PMSID	PMSIDV	'PMO¬'
PMVER	PMVER1	1
PMOPT	PMNONE	0
PMTO	Není	-1
PMCT	Není	0
PMKDC	Není	0

Tabulka 719. Pole v MQPMO (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
PMUDC	Není	0
PMIDC	Není	0
PMRQN	Není	Mezery
PMRMN	Není	Mezery
PMREC	Není	0
PMPRF	PFNONE	0
PMPRO	Není	0
PMRRO	Není	0
PMPRP	Není	Ukazatel Null nebo bajty s hodnotou Null
PMRRP	Není	Ukazatel Null nebo bajty s hodnotou Null

Poznámka:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQPMO Structure
D*
D* Structure identifier
D  PMSID                  1      4    INZ('PMO ')
D* Structure version number
D  PMVER                  5      8I 0 INZ(1)
D* Options that control the action of MQPUT and MQPUT1
D  PMOPT                  9     12I 0 INZ(0)
D* Reserved
D  PMTO                  13     16I 0 INZ(-1)
D* Object handle of input queue
D  PMCT                  17     20I 0 INZ(0)
D* Number of messages sent successfully to local queues
D  PMKDC                 21     24I 0 INZ(0)
D* Number of messages sent successfully to remote queues
D  PMUDC                 25     28I 0 INZ(0)
D* Number of messages that could notbe sent
D  PMIDC                 29     32I 0 INZ(0)
D* Resolved name of destination queue
D  PMRQN                 33     80    INZ
D* Resolved name of destination queue manager
D  PMRMN                 81    128    INZ
D* Number of put message records or response records present
D  PMREC                129    132I 0 INZ(0)
D* Flags indicating which MQPMR fields are present
D  PMPRF                133    136I 0 INZ(0)
D* Offset of first put message record from start of MQPMO
D  PMPRO                137    140I 0 INZ(0)
D* Offset of first response record from start of MQPMO
D  PMRRO                141    144I 0 INZ(0)
D* Address of first put message record
D  PMPRP                145    160*   INZ(*NULL)
D* Address of first response record
D  PMRRP                161    176*   INZ(*NULL)
D* Original message handle
D  PMOMH                177    184I 0
D* New message handle
D  PMNMH                185    190I 0
D* The action being performed
D  PMACT                191    194I 0
```

```
     D* Reserved
     D  PMRE1                195    198I 0
```

IBM i MQPMR (Vložit-záznam zprávy) na IBM i

Struktura MQPMR se používá k určení různých vlastností zprávy pro jedno místo určení, když je zpráva vkládána do distribučního seznamu.

Přehled

Účel: MQPMR je vstupní/výstupní struktura pro volání MQPUT a MQPUT1 .

Znaková sada a kódování: Data v MQPMR musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , musí být struktura ve znakové sadě a kódování klienta.

Použití: Při zadání pole těchto struktur ve volání MQPUT nebo MQPUT1 je možné zadat různé hodnoty pro každou cílovou frontu v distribučním seznamu. Některá pole jsou pouze vstupní, jiná jsou vstupní/výstupní.

Poznámka: Tato struktura je neobvyklá v tom, že nemá pevné rozložení. Pole v této struktuře jsou volitelná a přítomnost nebo nepřítomnost jednotlivých polí je označena příznaky v poli *PMPRF* v MQPMO. Pole, která jsou přítomna v produktu ***, se musí vyskytovat v následujícím pořadí*** :

- *PRMID*
- *PRCID*
- *PRGID*
- *PRFB*
- *PRACC*

Pole, která chybí, nezabírají v záznamu žádné místo.

Protože MQPMR nemá pevné rozvržení, není v souboru COPY uvedena žádná jeho definice. Programátor aplikace by měl vytvořit deklaraci obsahující pole vyžadovaná aplikací a nastavit příznaky v souboru *PMPRF* tak, aby označovaly pole, která jsou přítomna.

- "Pole" na stránce 1183
- "Počáteční hodnoty" na stránce 1185
- "Deklarace RPG" na stránce 1185

Pole

Struktura MQPMR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

PRACC (32bitový bitový řetězec)

Účtovací token.

Jedná se o token evidence, který má být použit pro zprávu odeslanou do fronty s názvem, který byl určen odpovídajícím prvkem v poli struktur MQOR zadaných ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *MDACC* v deskriptoru MQMD pro vložení do jedné fronty. Informace o obsahu tohoto pole naleznete v popisu položky *MDACC* v souboru "MQMD (deskriptor zprávy) na IBM i" na stránce 1105 .

Pokud toto pole není přítomno, použije se hodnota v MQMD.

Toto je vstupní pole.

PRCID (24bajtový bitový řetězec)

Identifikátor korelace.

Jedná se o identifikátor korelace, který má být použit pro zprávu odeslanou do fronty s názvem určeným odpovídajícím prvkem v poli struktur MQOR určeném ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *MDCID* v deskriptoru MQMD pro vložení do jedné fronty.

Pokud toto pole není v záznamu MQPMR přítomno nebo existuje méně záznamů MQPMR než místa určení, hodnota v MQMD se použije pro místa určení, která nemají záznam MQPMR obsahující pole *PRCID* .

Je-li uvedeno PMNCID, vygeneruje se *jeden* nový identifikátor korelace a použije se pro všechna místa určení v distribučním seznamu bez ohledu na to, zda mají záznamy MQPMR. Liší se od způsobu zpracování PMNMID (viz pole *PRMID*).

Toto je vstupní/výstupní pole.

PRFB (10místné celé číslo se znaménkem)

Zpětná vazba nebo kód příčiny.

Jedná se o kód zpětné vazby, který má být použit pro zprávu odeslanou do fronty s názvem určeným příslušným prvkem v poli struktur MQOR poskytnutém ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *MDFB* v deskriptoru MQMD pro vložení do jedné fronty.

Pokud toto pole není přítomno, použije se hodnota v MQMD.

Toto je vstupní pole.

PRGID (24bajtový bitový řetězec)

Identifikátor skupiny.

Jedná se o identifikátor skupiny, který má být použit pro zprávu odeslanou do fronty s názvem, který byl určen odpovídajícím prvkem v poli struktur MQOR poskytnutém ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *MDGID* v deskriptoru MQMD pro vložení do jedné fronty.

Pokud toto pole není v záznamu MQPMR přítomno nebo existuje méně záznamů MQPMR než místa určení, hodnota v MQMD se použije pro místa určení, která nemají záznam MQPMR obsahující pole *PRGID* . Hodnota je zpracována podle dokumentace v souboru Tabulka 716 na stránce 1172, ale s následujícími rozdíly:

- V případech, kdy by byl použit nový identifikátor skupiny, správce front vygeneruje pro každé místo určení jiný identifikátor skupiny (tj. žádné dva cíle nemají stejný identifikátor skupiny).
- V těch případech, kdy by se použila hodnota v poli, volání selže s kódem příčiny RC2258.

Toto je vstupní/výstupní pole.

PRMID (24bajtový bitový řetězec)

Identifikátor zprávy.

Jedná se o identifikátor zprávy, který má být použit pro zprávu odeslanou do fronty s názvem určeným odpovídajícím prvkem v poli struktur MQOR ve volání MQOPEN nebo MQPUT1 . Je zpracován stejným způsobem jako pole *MDMID* v deskriptoru MQMD pro vložení do jedné fronty.

Pokud toto pole není v záznamu MQPMR přítomno nebo existuje méně záznamů MQPMR než místa určení, hodnota v MQMD se použije pro místa určení, která nemají záznam MQPMR obsahující pole *PRMID* . Je-li tato hodnota MINONE, vygeneruje se pro *každý* z těchto cílů nový identifikátor zprávy (tj. žádné dva z těchto cílů nemají stejný identifikátor zprávy).

Je-li uveden PMNMID, vygenerují se nové identifikátory zpráv pro všechna místa určení v distribučním seznamu bez ohledu na to, zda mají záznamy MQPMR. Liší se od způsobu zpracování PMNCID (viz pole *PRCID*).

Toto je vstupní/výstupní pole.

Počáteční hodnoty

Pro tuto strukturu nejsou definovány žádné počáteční hodnoty, protože není poskytnuta žádná deklarace struktury. Následující vzorová deklarace ukazuje, jak by měla být struktura deklarována aplikačním programátorem, pokud jsou vyžadována všechna pole.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQPMR Structure
D*
D* Message identifier
D  PRMID                  1     24
D* Correlation identifier
D  PRCID                 25     48
D* Group identifier
D  PRGID                 49     72
D* Feedback or reason code
D  PRFB                  73     76I 0
D* Accounting token
D  PRACC                 77    108
```

IBM i MQRFH (Pravidla a formátování záhlaví) na IBM i

Struktura MQRFH definuje rozvržení pravidel a záhlaví formátování.

Přehled

Účel: Toto záhlaví lze použít k odeslání řetězcových dat ve formě dvojic název-hodnota.

Název formátu: FMRFH.

Znaková sada a kódování: Pole ve struktuře MQRFH (včetně *RFNVS*) jsou ve znakové sadě a kódování poskytnuté poli *MDCSI* a *MDENC* ve struktuře záhlaví, která předchází MQRFH, nebo těmito poli ve struktuře MQMD, pokud je MQRFH na začátku dat zprávy aplikace.

Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech front.

- "Pole" na stránce 1185
- "Počáteční hodnoty" na stránce 1187
- "Deklarace RPG" na stránce 1188

Pole

Struktura MQRFH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

RFCSI (10místné celé číslo se znaménkem)

Identifikátor znakové sady dat, která následují za *RFNVS*.

Tato volba určuje identifikátor znakové sady dat, která následují za položkou *RFNVS* ; se nevztahuje na znaková data v samotné struktuře MQRFH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota CSINHT vrácena voláním MQGET.

CSINHT nelze použít, pokud je hodnota pole *MDPAT* v MQMD ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

Číselné kódování dat, která následují za *RFNVS*.

Tato volba určuje číselné kódování dat, která následují za *RFNVS* ; se nevztahuje na numerická data v samotné struktuře MQRFH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je ENNAT.

RFFLG (10místné celé číslo se znaménkem)

Příznaky.

Lze uvést následující:

RFNONE
Žádné příznaky.

Počáteční hodnota tohoto pole je RFNONE.

RFFMT (8bajtový znakový řetězec)

Název formátu dat, která následují za *RFNVS*.

Tato volba určuje název formátu dat, která následují za položkou *RFNVS*.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *MDFMT* v MQMD.

Počáteční hodnota tohoto pole je FMNONE.

RFLEN (10místné celé číslo se znaménkem)

Celková délka MQRFH včetně *RFNVS*.

Jedná se o délku struktury MQRFH v bajtech, včetně pole *RFNVS* na konci struktury. Délka nezahrnuje žádná uživatelská data, která následují za polem *RFNVS* .

Chcete-li se vyhnout problémům s převodem uživatelských dat v některých prostředích, zvažte použití *RFLEN* jako násobku čtyř.

Následující konstanta udává délku *pevné* části struktury, tj. délku bez pole *RFNVS* :

RFLENV
Délka pevné části struktury MQRFH.

Počáteční hodnota tohoto pole je RFLENV.

RFNVS (řetězec znaků o velikosti n bajtů)

Řetězec obsahující dvojice název-hodnota.

Jedná se o znakový řetězec proměnné délky obsahující dvojice název-hodnota ve tvaru:

```
name1 value1 name2 value2 name3 value3 ...
```

Každý název nebo hodnota musí být odděleny od sousedního názvu nebo hodnoty jedním nebo více prázdnými znaky; tyto mezery nejsou významné. Název nebo hodnota může obsahovat významné mezery pomocí předpony a přípon názvu nebo hodnoty znakem uvozovky; všechny znaky mezi počáteční uvozovkou a odpovídající koncovou uvozovkou jsou považovány za významné. V následujícím příkladu je název `FAMOUS_WORDS`a hodnota je `Hello World`:

```
FAMOUS_WORDS "Hello World"
```

Název nebo hodnota může obsahovat jakékoli jiné znaky než znak null (který se chová jako oddělovač pro *RFNVS*). V zájmu usnadnění interoperability však může aplikace raději omezit názvy na následující znaky:

- První znak: velká nebo malá písmena abecedy (A až Z nebo a až z) nebo podtržítko.
- Následné znaky: velká nebo malá písmena abecedy, desetinná číslice (0 až 9), podtržítko, pomlčka nebo tečka.

Pokud název nebo hodnota obsahuje jednu nebo více uvozovek, název nebo hodnota musí být uzavřeny v uvozovkách a každá uvozovka v řetězci musí být zdvojnásobena:

```
Famous_Words "The program displayed ""Hello World"""
```

Názvy a hodnoty rozlišují velká a malá písmena, to znamená, že malá písmena nejsou považována za stejná jako velká písmena. Například FAMOUS_WORDS a Famous_Words jsou dva různé názvy.

Délka v bajtech *RFNVS* je rovna *RFLEN* minus RFLENV. Chcete-li se vyhnout problémům s převodem uživatelských dat v některých prostředích, doporučuje se, aby tato délka byla násobkem čtyř. *RFNVS* musí být vyplněn mezerami na tuto délku nebo ukončen dříve umístěním znaku null za poslední významný znak v řetězci. Znak null a bajty, které za ním následují, až do uvedené délky *RFNVS*, jsou ignorovány.

Poznámka: Protože délka tohoto pole není pevná, pole je vynecháno z deklarací struktury, které jsou poskytnuty pro podporované programovací jazyky.

RFSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

RFSIDV
Identifikátor pro pravidla a strukturu záhlaví formátování.

Počáteční hodnota tohoto pole je RFSIDV.

RFVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

RFVER1
Version-1 pravidla a struktura záhlaví formátování.

Počáteční hodnota tohoto pole je RFVER1.

Počáteční hodnoty

Tabulka 720. Pole v MQRFH

Název pole	Název konstanty	Hodnota konstanty
RFSID	RFSIDV	'RFH¬'
RFVER	RFVER1	1
RFLEN	RFLENV	32
RFENC	ENNAT	Závisí na prostředí
RFCSI	CSUNDF	0
RFFMT	FMNONE	Mezery
RFFLG	RFNONE	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQRFH Structure
D*
D* Structure identifier
D  RFSID                  1      4    INZ('RFH ')
D* Structure version number
D  RFVER                  5      8I 0 INZ(1)
D* Total length of MQRFH includingNameValueString
D  RFLEN                  9     12I 0 INZ(32)
D* Numeric encoding of data that followsNameValueString
D  RFENC                 13     16I 0 INZ(273)
D* Character set identifier of data thatfollows NameValueString
D  RFCSI                 17     20I 0 INZ(0)
D* Format name of data that followsNameValueString
D  RFFMT                 21     28    INZ('        ')
D* Flags
D  RFFLG                 29     32I 0 INZ(0)
```

IBM i MQRFH2 (Pravidla a formátování záhlaví 2) na IBM i

Struktura MQRFH2 definuje formát pravidel version-2 a záhlaví formátování.

Přehled

Účel: Toto záhlaví lze použít k odeslání dat, která byla zakódována pomocí syntaxe podobné XML. Zpráva může obsahovat dvě nebo více struktur MQRFH2 v řadě, přičemž uživatelská data mohou volitelně následovat po poslední struktuře MQRFH2 v řadě.

Název formátu: FMRFH2.

Znaková sada a kódování: Pro znakovou sadu a kódování použité pro strukturu MQRFH2 platí zvláštní pravidla:

- Jiná pole než *RF2NVD* jsou ve znakové sadě a kódování, které jsou dány poli *MDCSI* a *MDENC* ve struktuře záhlaví, která předchází MQRFH2, nebo těmito poli ve struktuře MQMD, pokud je MQRFH2 na začátku dat zprávy aplikace.

  Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech front.

  Je-li ve volání MQGET zadána hodnota GMCONV, správce front tato pole převede na požadovanou znakovou sadu a kódování.
- *RF2NVD* je ve znakové sadě dané polem *RF2NVC* . Pouze určité znakové sady Unicode jsou platné pro *RF2NVC* (podrobnosti viz popis *RF2NVC*).

  Některé znakové sady mají reprezentaci, která je závislá na kódování. Pokud je *RF2NVC* jednou z těchto znakových sad, *RF2NVD* musí být ve stejném kódování jako ostatní pole v MQRFH2.

  Je-li ve volání MQGET zadána volba GMCONV, správce front převede soubor *RF2NVD* na požadované kódování, ale nezmění svou znakovou sadu.

- "Pole" na stránce 1188
- "Počáteční hodnoty" na stránce 1193
- "Deklarace RPG" na stránce 1193

Pole

Struktura MQRFH2 obsahuje následující pole; pole jsou popsána v abecedním pořadí:

RF2CSI (10místné celé číslo se znaménkem)

Identifikátor znakové sady dat, která následují za posledním polem *RF2NVD* .

Určuje identifikátor znakové sady dat, která následují za posledním polem *RF2NVD* . Nevztahuje se na znaková data v samotné struktuře MQRFH2 .

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota CSINHT vrácena voláním MQGET.

CSINHT nelze použít, pokud je hodnota pole *MDPAT* v MQMD ATBRKR.

Počáteční hodnota tohoto pole je CSINHT.

RF2ENC (10místné celé číslo se znaménkem)

Číselné kódování dat, která následují za posledním polem *RF2NVD* .

Tato volba určuje číselné kódování dat, která následují za posledním polem *RF2NVD* ; nevztahuje se na číselná data v samotné struktuře MQRFH2 .

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je ENNAT.

RF2FLG (10místné celé číslo se znaménkem)

Příznaky.

Musí být zadána následující hodnota:

RFNONE
Žádné příznaky.

Počáteční hodnota tohoto pole je RFNONE.

RF2FMT (8bajtový znakový řetězec)

Název formátu dat, která následují za posledním polem *RF2NVD* .

Tato volba určuje název formátu dat, která následují za posledním polem *RF2NVD* .

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *MDFMT* v MQMD.

Počáteční hodnota tohoto pole je FMNONE.

RF2LEN (10místné celé číslo se znaménkem)

Celková délka MQRFH2 včetně všech polí *RF2NVL* a *RF2NVD* .

Jedná se o délku struktury MQRFH2 v bajtech, včetně polí *RF2NVL* a *RF2NVD* na konci struktury. Je platné, aby na konci struktury bylo více dvojic polí *RF2NVL* a *RF2NVD* v pořadí:

```
length1, data1, length2, data2, ...
```

RF2LEN nezahrnuje žádná uživatelská data, která by mohla následovat za posledním *RF2NVD* polem na konci struktury.

Chcete-li se vyhnout problémům s převodem uživatelských dat v některých prostředích, zvažte použití *RF2LEN* jako násobku čtyř.

Následující konstanta udává délku *pevné* části struktury, tj. délku bez polí *RF2NVL* a *RF2NVD* :

RFLEN2
Délka pevné části struktury MQRFH2 .

Počáteční hodnota tohoto pole je RFLEN2.

RF2NVC (10místné celé číslo se znaménkem)

Identifikátor znakové sady *RF2NVD*.

Uvádí identifikátor kódované znakové sady dat v poli *RF2NVD* . Liší se od znakové sady ostatních řetězců ve struktuře MQRFH2 a může se lišit od znakové sady dat (pokud existuje), která následuje za posledním polem *RF2NVD* na konci struktury.

RF2NVC musí mít jednu z následujících hodnot CCSID:

1200
UTF-16, nejnovější podporovaná verze Unicode

13488
UTF-16, verze Unicode 2.0 dílčí sada

17584
Podmnožina UTF-16, Unicode verze 3.0 (obsahuje symbol Euro)

1208
UTF-8, nejnovější podporovaná verze Unicode

Pro znakové sady UTF-16 musí být kódování (pořadí bajtů) *RF2NVD* stejné jako kódování ostatních polí ve struktuře MQRFH2 . Náhradní znaky (X'D800'až X'DFFF') nejsou podporovány.

Poznámka: Pokud *RF2NVC* nemá jednu z výše uvedených hodnot a struktura MQRFH2 vyžaduje převod ve volání MQGET, volání se dokončí s kódem příčiny RC2111 a zpráva se vrátí nepřevedená.

Počáteční hodnota tohoto pole je 1208.

RF2NVD (řetězec n-bajtových znaků)

Data název/hodnota.

Jedná se o znakový řetězec s proměnnou délkou obsahující data kódovaná pomocí syntaxe podobné XML. Délka tohoto řetězce v bajtech je poskytnuta polem *RF2NVL* , které předchází poli *RF2NVD* ; tato délka by měla být násobkem čtyř.

Pole *RF2NVL* a *RF2NVD* jsou volitelná, ale pokud jsou přítomna, musí se vyskytovat jako dvojice a sousedící. Dvojici polí lze opakovat tolikrát, kolikrát je třeba, například:

```
length1 data1 length2 data2 length3 data3
```

Protože jsou tato pole volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.

Parametr *RF2NVD* je neobvyklý, protože není převeden na znakovou sadu určenou ve volání MQGET, když je zpráva načtena s použitím volby GMCONV. Produkt *RF2NVD* zůstává v původní znakové sadě. Příkaz *RF2NVD* je však převeden na kódování zadané ve volání MQGET.

Syntaxe dat název/hodnota: Řetězec se skládá z jedné "složky", která obsahuje nula nebo více vlastností. Složka je oddělena počáteční a koncovou značkou XML se stejným názvem jako složka:

```
<folder> property1 property2 ... </folder>
```

Znaky následující za koncovou značkou složky, až do délky definované pomocí *RF2NVL*, musí být prázdné. V rámci složky se každá vlastnost skládá z názvu a hodnoty a volitelně z datového typu:

```
<name dt="datatype">value</name>
```

V těchto příkladech:

- Znaky oddělovače (<, =, ",/a >) musí být zadány přesně tak, jak jsou zobrazeny.
- name je uživatelem zadaný název vlastnosti. Další informace o názvech naleznete v následujícím příkladu.

- `datatype` je volitelný datový typ vlastnosti určený uživatelem; platné datové typy jsou uvedeny v následujícím příkladu.
- `value` je uživatelem zadaná hodnota vlastnosti. Další informace o hodnotách naleznete v následujících odstavcích.
- Mezery jsou významné mezi znakem > , který předchází hodnotě, a znakem < , který následuje za hodnotou, a alespoň jedna mezera musí předcházet `dt=`. Jinde lze mezery volně kódovat mezi značkami, nebo před nebo za značkami (například za účelem zlepšení čitelnosti); tyto mezery nejsou významné.

Pokud spolu vlastnosti vzájemně souvisí, mohou být seskupeny tak, že je zapouzdřují do počáteční a koncové značky XML se stejným názvem jako skupina:

```
<folder> <group> property1 property2 ... </group> </folder>
```

Skupiny mohou být bez omezení vnořeny do jiných skupin a skupina se může ve složce vyskytovat více než jednou. Je také platné, aby složka obsahovala některé vlastnosti ve skupinách a jiné vlastnosti, které nejsou ve skupinách.

Názvy vlastností, skupin a složek: Názvy vlastností, skupin a složek musí být platnými názvy značek XML s výjimkou dvojtečky, která není povolena v názvu vlastnosti, skupiny nebo složky. Konkrétně se jedná o následující podmínky:

- Názvy musí začínat písmenem nebo podtržítkem. Platná písmena jsou definována ve specifikaci XML W3C a skládají se v podstatě z kategorií Unicode Ll, Lu, Lo, Lt a Nl.
- Zbývající znaky v názvu mohou být písmena, desetinná čísla, podtržítka, pomlčky nebo tečky. Ty odpovídají kategoriím Unicode Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm a Nd.
- Znaky kompatibility Unicode (X'F900' a vyšší) nejsou povoleny v žádné části názvu.
- Názvy nesmí začínat řetězcem XML v žádné kombinaci velkých a malých písmen.

Kromě toho:

- V názvech se rozlišují malá a velká písmena. Například ABC, abca Abc jsou tři různé názvy.
- Každá složka má samostatný obor názvů. Výsledkem je, že skupina nebo vlastnost v jedné složce není v konfliktu se skupinou nebo vlastností se stejným názvem v jiné složce.
- Skupiny a vlastnosti zabírají v rámci složky stejný obor názvů. V důsledku toho nemůže mít vlastnost stejný název jako skupina ve složce obsahující tuto vlastnost.

Obecně platí, že programy, které analyzují pole *RF2NVD* , by měly ignorovat vlastnosti nebo skupiny, které mají názvy, které program nerozpozná, za předpokladu, že tyto vlastnosti nebo skupiny jsou správně vytvořeny.

Datové typy vlastností: Každá vlastnost může mít volitelný datový typ. Je-li uveden, datový typ musí být jedna z následujících hodnot, a to velká, malá nebo malá písmena:

Tabulka 721. Datové typy a jejich využití

Datový typ	**Použití**
`string`	Libovolná posloupnost znaků. Určité znaky musí být uvedeny pomocí řídicích posloupností.
`boolean`	Znak 0 nebo 1 (1 označuje TRUE).
`bin.hex`	Hexadecimální číslice představující oktety.
`i1`	Celé číslo v rozsahu -128 až +127, vyjádřené pouze desetinnými číslicemi a volitelným znaménkem.

Tabulka 721. Datové typy a jejich využití (pokračování)	
Datový typ	**Použití**
`i2`	Celé číslo v rozsahu -32 768 až +32 767, vyjádřené pouze desetinnými číslicemi a volitelným znaménkem.
`i4`	Celé číslo v rozsahu -2 147 483 648 až + 2 147 483 647, vyjádřené pouze desetinnými číslicemi a volitelným znaménkem.
`i8`	Celé číslo v rozsahu -9 223 372 036 854 775 808 až + 9 223 372 036 854 775 807, vyjádřené pouze desetinnými číslicemi a volitelným znaménkem.
`int`	Celé číslo v rozsahu -9 223 372 036 854 775 808 až + 9 223 372 036 854 775 807, vyjádřené pouze desetinnými číslicemi a volitelným znaménkem. Toto lze použít místo `i1`, `i2`, `i4`nebo `i8` , pokud odesilatel nechce naznačovat konkrétní přesnost.
`r4`	Číslo s pohyblivou řádovou čárkou s velikostí v rozsahu 1.175E-37 až 3.402 823 47E+38vyjádřené pomocí desetinných číslic, volitelného znaménka, volitelných zlomkových číslic a volitelného exponentu.
`r8`	Číslo s pohyblivou řádovou čárkou s velikostí v rozsahu 2.225E-307 až 1.797 693 134 862 3E+308 vyjádřené pomocí desetinných číslic, volitelného znaménka, volitelných zlomkových číslic a volitelného exponentu.

Hodnoty vlastností: Hodnota vlastnosti se může skládat z libovolných znaků kromě speciálních znaků, které mají povinnou přidruženou řídicí posloupnost. Každý výskyt v hodnotě znaku označeného v následující tabulce jako "povinný" musí být nahrazen odpovídající řídicí posloupností. Tabulka také zobrazuje znaky, které mají volitelnou přidruženou řídicí posloupnost. Každý výskyt v hodnotě znaku označeného jako "volitelný" může být nahrazen odpovídající řídicí posloupností, ale to není povinné.

Tabulka 722. Řídicí znaky a jejich použití		
Znak	**Řídicí posloupnost**	**Pouľití**
`&`	`&`	Povinné
`<`	`<`	Povinné
`>`	`>`	Volitelné
`"`	`"`	Volitelné
`'`	`'`	Volitelné

Poznámka: Znak & na začátku řídicí posloupnosti nesmí být nahrazen znakem `&`.

V následujícím příkladu jsou mezery v hodnotě významné; avšak nejsou potřeba žádné řídicí posloupnosti:

```
<Famous_Words>The program displayed "Hello World"</Famous_Words>
```

RF2NVL (10místné celé číslo se znaménkem)

Délka řetězce *RF2NVD*.

Určuje délku dat v bajtech v poli *RF2NVD* . Chcete-li se vyhnout problémům s převodem dat (pokud existuje), který následuje za polem *RF2NVD* , *RF2NVL* by měl být násobkem čtyř.

Poznámka: Pole *RF2NVL* a *RF2NVD* jsou volitelná, ale pokud jsou přítomna, musí se vyskytovat jako dvojice a sousedící. Dvojici polí lze opakovat tolikrát, kolikrát je třeba, například:

```
length1 data1 length2 data2 length3 data3
```

Protože jsou tato pole volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.

RF2SID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

RFSIDV
Identifikátor pro pravidla a strukturu záhlaví formátování.

Počáteční hodnota tohoto pole je RFSIDV.

RF2VER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

RFVER2
Version-2 -pravidla a struktura formátování záhlaví.

Počáteční hodnota tohoto pole je RFVER2.

Počáteční hodnoty

Tabulka 723. Pole v produktu MQRFH2

Název pole	Název konstanty	Hodnota konstanty
RF2SID	RFSIDV	'RFH¬'
RF2VER	RFVER2	2
RF2LEN	RFLEN2	36
RF2ENC	ENNAT	Závisí na prostředí
RF2CSI	CSINHT	-2
RF2FMT	FMNONE	Mezery
RF2FLG	RFNONE	0
RF2NVC	Není	1208

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
     D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     D*
     D* MQRFH2 Structure
```

```
     D*
     D* Structure identifier
     D  RF2SID                 1      4    INZ('RFH ')
     D* Structure version number
     D  RF2VER                 5      8I 0 INZ(2)
     D* Total length of MQRFH2 including allNameValueLength and
     D* NameValueDatafields
     D  RF2LEN                 9     12I 0 INZ(36)
     D* Numeric encoding of data that followslast NameValueData field
     D  RF2ENC                13     16I 0 INZ(273)
     D* Character set identifier of data thatfollows last NameValueData field
     D  RF2CSI                17     20I 0 INZ(-2)
     D* Format name of data that follows lastNameValueData field
     D  RF2FMT                21     28    INZ('        ')
     D* Flags
     D  RF2FLG                29     32I 0 INZ(0)
     D* Character set identifier ofNameValueData
     D  RF2NVC                33     36I 0 INZ(1208)
```

IBM i MQRMH (záhlaví referenční zprávy) na IBM i

Struktura MQRMH definuje formát záhlaví referenční zprávy.

Přehled

Účel: Toto záhlaví se používá u uživatelských procedur kanálu zpráv napsaných uživatelem k odesílání velkého množství dat (nazývaných "hromadná data"). z jednoho správce front do jiného. Rozdíl ve srovnání s normálním systémem zpráv spočívá v tom, že hromadná data nejsou uložena ve frontě; místo toho je ve frontě uložen pouze *odkaz* na hromadná data. To snižuje možnost vyčerpání prostředků IBM MQ několika velkými zprávami.

Název formátu: FMRMH.

Znaková sada a kódování: Znaková data v MQRMH a řetězce adresované poli offsetu musí být ve znakové sadě lokálního správce front; to je dáno atributem správce front **CodedCharSetId** . Číselná data v MQRMH musí být v nativním kódování počítače; toto je dáno hodnotou ENNAT pro programovací jazyk C.

Znaková sada a kódování MQRMH musí být nastaveny do polí *MDCSI* a *MDENC* v:

- MQMD (pokud je struktura MQRMH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQRMH (všechny ostatní případy).

Použití: Aplikace vkládá zprávu sestávající z MQRMH, ale vynechává hromadná data. Když je zpráva přečtena z přenosové fronty agentem kanálu zpráv (MCA), je vyvolána uživatelská procedura pro zpracování záhlaví referenční zprávy. Uživatelská procedura může k referenční zprávě připojit hromadná data identifikovaná strukturou MQRMH před tím, než agent MCA odešle zprávu prostřednictvím kanálu dalšímu správci front.

Na přijímacím konci by měla existovat uživatelská procedura pro zprávy, která čeká na referenční zprávy. Když je přijata referenční zpráva, uživatelská procedura by měla vytvořit objekt z hromadných dat, která následují za MQRMH ve zprávě, a poté předat referenční zprávu bez hromadných dat. Referenční zprávu může později načíst aplikace, která načte referenční zprávu (bez hromadných dat) z fronty.

Za normálních okolností je struktura MQRMH vše, co je ve zprávě. Pokud se však zpráva nachází v přenosové frontě, bude před strukturou MQRMH předcházet jedno nebo více dalších záhlaví.

Referenční zprávu lze také odeslat do rozdělovníku. V tomto případě struktura MQDH a související záznamy předcházejí struktuře MQRMH, když je zpráva v přenosové frontě.

Poznámka: Referenční zpráva by neměla být odeslána jako segmentovaná zpráva, protože uživatelská procedura zprávy ji nemůže správně zpracovat.

- “Převod dat” na stránce 1195
- “Pole” na stránce 1195
- “Počáteční hodnoty” na stránce 1199

- “Deklarace RPG” na stránce 1200

Převod dat

Pro účely převodu dat zahrnuje převod struktury MQRMH převod dat zdrojového prostředí, název zdrojového objektu, data cílového prostředí a název cílového objektu. Jakékoli jiné bajty v rozsahu *RMLEN* bajtů od začátku struktury jsou buď vyřazeny, nebo mají nedefinované hodnoty po převodu dat. Hromadná data budou převedena za předpokladu, že všechny následující příkazy jsou pravdivé:

- Hromadná data jsou přítomna ve zprávě, když se provádí převod dat.
- Pole *RMFMT* v MQRMH má jinou hodnotu než FMNONE.
- Uživatelská procedura pro převod dat zapsaná uživatelem existuje se zadaným názvem formátu.

Mějte však na paměti, že obvykle nejsou hromadná data ve zprávě přítomna, když je zpráva ve frontě, a že v důsledku toho nebudou hromadná data převedena volbou GMCONV.

Pole

Struktura MQRMH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

RMCSI (10místné celé číslo se znaménkem)

Identifikátor znakové sady hromadných dat.

Tato volba určuje identifikátor znakové sady pro hromadná data; nevztahuje se na znaková data v samotné struktuře MQRMH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota CSINHT vrácena voláním MQGET.

CSINHT nelze použít, pokud je hodnota pole *MDPAT* v MQMD ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

RMDEL (10místné celé číslo se znaménkem)

Délka dat cílového prostředí.

Je-li toto pole nulové, nejsou k dispozici žádná data cílového prostředí a parametr *RMDEO* je ignorován.

RMDEO (10místné celé číslo se znaménkem)

Posunutí dat cílového prostředí.

Toto pole určuje posun dat cílového prostředí od začátku struktury MQRMH. Data cílového prostředí mohou být určena tvůrcem referenční zprávy, pokud jsou tato data tvůrci známa. Data cílového prostředí mohou být například cestou k adresáři objektu, kde se mají hromadná data uložit. Pokud však tvůrce nezná data cílového prostředí, je odpovědností uživatelské procedury pro zprávy dodané uživatelem určit potřebné informace o prostředí.

Délku dat cílového prostředí uvádí *RMDEL* ; Je-li tato délka nula, nejsou k dispozici žádná data cílového prostředí a parametr *RMDEO* je ignorován. Jsou-li k dispozici, musí být data cílového prostředí od začátku struktury zcela v rozsahu *RMLEN* bajtů.

Aplikace by neměly předpokládat, že data cílového prostředí jsou souvislá s libovolnými daty adresovanými poli *RMSEO*, *RMSNO*a *RMDNO* .

Počáteční hodnota tohoto pole je 0.

RMDL (10místné celé číslo se znaménkem)

Délka hromadných dat.

Pole *RMDL* určuje délku hromadných dat, na která odkazuje struktura MQRMH.

Pokud jsou ve zprávě přítomna hromadná data, data začínají s posunem *RMLEN* bajtů od začátku struktury MQRMH. Délka celé zprávy minus *RMLEN* udává délku přítomných hromadných dat.

Pokud jsou ve zprávě přítomna data, *RMDL* uvádí množství příslušných dat. Normální případ znamená, že *RMDL* má mít stejnou hodnotu jako délka dat přítomných ve zprávě.

Pokud struktura MQRMH představuje zbývající data v objektu (počínaje určeným logickým offsetem), hodnotu nula lze použít pro parametr *RMDL*, pokud nejsou ve zprávě přítomna hromadná data.

Pokud nejsou k dispozici žádná data, konec MQRMH se shoduje s koncem zprávy.

Počáteční hodnota tohoto pole je 0.

RMDNL (10místné celé číslo se znaménkem)

Délka názvu cílového objektu.

Je-li toto pole nula, neexistuje žádný název cílového objektu a parametr *RMDNO* je ignorován.

RMDNO (10místné celé číslo se znaménkem)

Posunutí názvu cílového objektu.

Toto pole určuje posun názvu cílového objektu od začátku struktury MQRMH. Název cílového objektu může určit tvůrce referenční zprávy, jsou-li tato data známa tvůrci. Pokud však tvůrce nezná název cílového objektu, je odpovědností uživatelské procedury pro zprávy určené k identifikaci objektu, který má být vytvořen nebo upraven.

Délka názvu cílového objektu je dána *RMDNL* ; Je-li tato délka nulová, neexistuje žádný název cílového objektu a parametr *RMDNO* je ignorován. Je-li uveden, název cílového objektu musí být od začátku struktury zcela v rozmezí *RMLEN* bajtů.

Aplikace by neměly předpokládat, že název cílového objektu je souvislý se všemi daty adresovanými poli *RMSEO*, *RMSNO*a *RMDEO* .

Počáteční hodnota tohoto pole je 0.

RMDO (10místné celé číslo se znaménkem)

Nízká odchylka hromadných dat.

Toto pole určuje nízkou odchylku hromadných dat od začátku objektu, jehož část hromadných dat tvoří. Posun hromadných dat od začátku objektu se nazývá *logický posun*. Nejedná se o fyzické posunutí hromadných dat od začátku struktury MQRMH-toto posunutí je dáno proměnnou *RMLEN*.

Chcete-li povolit odesílání velkých objektů pomocí referenčních zpráv, logický posun je rozdělen do dvou polí a skutečný logický posun je dán součtem těchto dvou polí:

- *RMDO* představuje zbytek získaný při dělení logického posunu o 1 000 000 000. Jedná se tedy o hodnotu v rozsahu 0 až 999 999 999 999.
- *RMDO2* představuje výsledek získaný vydělením logického posunu 1 000 000 000. Je to tedy počet úplných násobků 1 000 000 000, které existují v logickém posunu. Počet násobků je v rozsahu 0 až 999 999 999 999.

Počáteční hodnota tohoto pole je 0.

RMDO2 (10místné celé číslo se znaménkem)

Vysoká odchylka hromadných dat.

Toto pole určuje vysoké posunutí hromadných dat od začátku objektu, jehož část hromadných dat tvoří. Jedná se o hodnotu v rozsahu 0 až 999 999 999 999. Podrobnosti viz *RMDO* .

Počáteční hodnota tohoto pole je 0.

RMENC (10místné celé číslo se znaménkem)

Číselné kódování hromadných dat.

Tato volba určuje číselné kódování hromadných dat; nevztahuje se na číselná data v samotné struktuře MQRMH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je ENNAT.

RMFLG (10místné celé číslo se znaménkem)

Příznaky referenční zprávy.

Jsou definovány následující příznaky:

RMLAST
Referenční zpráva obsahuje nebo představuje poslední část objektu.

Tento příznak označuje, že referenční zpráva představuje nebo obsahuje poslední část odkazovaného objektu.

RMNLST
Referenční zpráva neobsahuje ani nepředstavuje poslední část objektu.

RMNLST je definován jako pomůcka pro programovou dokumentaci. Není zamýšleno, aby tato volba byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

Počáteční hodnota tohoto pole je RMNLST.

RMFMT (8bajtový znakový řetězec)

Název formátu hromadných dat.

Tato volba určuje název formátu hromadných dat.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *MDFMT* v MQMD.

Počáteční hodnota tohoto pole je FMNONE.

RMLEN (10místné celé číslo se znaménkem)

Celková délka MQRMH, včetně řetězců na konci pevných polí, ale ne hromadných dat.

Počáteční hodnota tohoto pole je nula.

RMOII (24bajtový bitový řetězec)

Identifikátor instance objektu.

Toto pole lze použít k identifikaci určité instance objektu. Pokud není potřeba, měla by být nastavena na následující hodnotu:

OIINON
Nebyl uveden žádný identifikátor instance objektu.

Hodnota je binární nula pro délku pole.

Délka tohoto pole je dána hodnotou LNOIID. Počáteční hodnota tohoto pole je OIINON.

RMOT (8bajtový znakový řetězec)

Typ objektu.

Jedná se o název, který může být použit uživatelskou procedurou pro zprávy k rozpoznání typů referenční zprávy, kterou podporuje. Zvažte možnost podřídit název stejným pravidlům jako pole *RMFMT* .

Počáteční hodnota tohoto pole je 8 mezer.

RMSEL (10místné celé číslo se znaménkem)

Délka dat zdrojového prostředí.

Je-li toto pole nula, nejsou k dispozici žádná data zdrojového prostředí a parametr *RMSEO* je ignorován.

Počáteční hodnota tohoto pole je 0.

RMSEO (10místné celé číslo se znaménkem)

Posunutí dat zdrojového prostředí.

Toto pole určuje posun zdrojových dat prostředí od začátku struktury MQRMH. Data zdrojového prostředí mohou být určena tvůrcem referenční zprávy, pokud jsou tato data tvůrci známa. Data zdrojového prostředí mohou být například cestou k adresáři objektu, který obsahuje hromadná data. Pokud však tvůrce nezná data zdrojového prostředí, je odpovědností uživatelské procedury pro zprávy dodané uživatelem určit veškeré potřebné informace o prostředí.

Délka zdrojových dat prostředí je dána *RMSEL* ; Je-li tato délka nulová, nejsou k dispozici žádná data zdrojového prostředí a parametr *RMSEO* je ignorován. Pokud jsou k dispozici, data zdrojového prostředí musí být od začátku struktury zcela v rozsahu *RMLEN* bajtů.

Aplikace by neměly předpokládat, že data prostředí se spustí okamžitě po posledním pevném poli ve struktuře, nebo že jsou souvislá s libovolnými daty adresovanými poli *RMSNO*, *RMDEO*a *RMDNO* .

Počáteční hodnota tohoto pole je 0.

RMSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

RMSIDV
Identifikátor pro strukturu záhlaví referenční zprávy.

Počáteční hodnota tohoto pole je RMSIDV.

RMSNL (10místné celé číslo se znaménkem)

Délka názvu zdrojového objektu.

Je-li toto pole nulové, neexistuje žádný název zdrojového objektu a parametr *RMSNO* je ignorován.

Počáteční hodnota tohoto pole je 0.

RMSNO (10místné celé číslo se znaménkem)

Posunutí názvu zdrojového objektu.

Toto pole určuje posun názvu zdrojového objektu od začátku struktury MQRMH. Název zdrojového objektu může určit tvůrce referenční zprávy, pokud jsou tato data tvůrci známa. Pokud však tvůrce nezná název zdrojového objektu, je za identifikaci objektu, ke kterému má být přistupováno, zodpovědností uživatelské procedury pro zprávy.

Délka názvu zdrojového objektu je dána *RMSNL* ; Je-li tato délka nula, neexistuje žádný název zdrojového objektu a parametr *RMSNO* je ignorován. Je-li uveden, musí být název zdrojového objektu zcela v rozmezí *RMLEN* bajtů od začátku struktury.

Aplikace by neměly předpokládat, že název zdrojového objektu je souvislý se všemi daty adresovanými poli *RMSEO*, *RMDEO*a *RMDNO* .

Počáteční hodnota tohoto pole je 0.

RMVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

RMVER1
Struktura záhlaví zprávy odkazu Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

RMVERC
Aktuální verze struktury záhlaví referenční zprávy.

Počáteční hodnota tohoto pole je RMVER1.

Počáteční hodnoty

Tabulka 724. Pole v MQRMH

Název pole	Název konstanty	Hodnota konstanty
RMSID	RMSIDV	'RMH¬'
RMVER	RMVER1	1
RMLEN	Není	0
RMENC	ENNAT	Závisí na prostředí
RMCSI	CSUNDF	0
RMFMT	FMNONE	Mezery
RMFLG	RMNLST	0
RMOT	Není	Mezery
RMOII	OIINON	Hodnoty null
RMSEL	Není	0
RMSEO	Není	0
RMSNL	Není	0
RMSNO	Není	0
RMDEL	Není	0
RMDEO	Není	0
RMDNL	Není	0
RMDNO	Není	0
RMDL	Není	0
RMDO	Není	0
RMDO2	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQRMH Structure
D*
D* Structure identifier
D  RMSID                  1      4    INZ('RMH ')
D* Structure version number
D  RMVER                  5      8I 0 INZ(1)
D* Total length of MQRMH, includingstrings at end of fixed fields,but not
D* the bulk data
```

```
     D  RMLEN                  9     12I 0 INZ(0)
     D* Numeric encoding of bulk data
     D  RMENC                 13     16I 0 INZ(273)
     D* Character set identifier of bulkdata
     D  RMCSI                 17     20I 0 INZ(0)
     D* Format name of bulk data
     D  RMFMT                 21     28    INZ('        ')
     D* Reference message flags
     D  RMFLG                 29     32I 0 INZ(0)
     D* Object type
     D  RMOT                  33     40    INZ
     D* Object instance identifier
     D  RMOII                 41     64    INZ(X'00000000000000-
     D                                     0000000000000000000000-
     D                                     000000000000')
     D* Length of source environmentdata
     D  RMSEL                 65     68I 0 INZ(0)
     D* Offset of source environmentdata
     D  RMSEO                 69     72I 0 INZ(0)
     D* Length of source object name
     D  RMSNL                 73     76I 0 INZ(0)
     D* Offset of source object name
     D  RMSNO                 77     80I 0 INZ(0)
     D* Length of destination environmentdata
     D  RMDEL                 81     84I 0 INZ(0)
     D* Offset of destination environmentdata
     D  RMDEO                 85     88I 0 INZ(0)
     D* Length of destination objectname
     D  RMDNL                 89     92I 0 INZ(0)
     D* Offset of destination objectname
     D  RMDNO                 93     96I 0 INZ(0)
     D* Length of bulk data
     D  RMDL                  97    100I 0 INZ(0)
     D* Low offset of bulk data
     D  RMDO                 101    104I 0 INZ(0)
     D* High offset of bulk data
     D  RMDO2                105    108I 0 INZ(0)
```

Deklarace RPG

IBM i MQRR (záznam odezvy) na IBM i

Struktura MQRR se používá k přijetí kódu dokončení a kódu příčiny, který je výsledkem operace otevření nebo vložení pro jednu cílovou frontu, když je cílem distribuční seznam.

Přehled

Účel: MQRR je výstupní struktura pro volání MQOPEN, MQPUT a MQPUT1 .

Znaková sada a kódování: Data v MQRR musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , musí být struktura ve znakové sadě a kódování klienta.

Použití: Při zadání pole těchto struktur ve voláních MQOPEN a MQPUT nebo ve volání MQPUT1 je možné určit kódy dokončení a kódy příčiny pro všechny fronty v rozdělovníku, když je výsledek volání smíšený, tj. když je volání úspěšné pro některé fronty v seznamu, ale jiné selhává. Kód příčiny RC2136 z volání označuje, že záznamy odpovědí (jsou-li poskytovány aplikací) byly nastaveny správcem front.

- "Pole" na stránce 1200
- "Počáteční hodnoty" na stránce 1201
- "Deklarace RPG" na stránce 1201

Pole

Struktura MQRR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

RRCC (10místné celé číslo se znaménkem)

Kód dokončení pro frontu.

Jedná se o kód dokončení, který je výsledkem operace otevření nebo vložení pro frontu s názvem určeným příslušným prvkem v poli struktur MQOR poskytnutém ve volání MQOPEN nebo MQPUT1 .

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je CCOK.

RRREA (10místné celé číslo se znaménkem)

Kód příčiny pro frontu.

Jedná se o kód příčiny, který je výsledkem operace otevření nebo vložení pro frontu s názvem, který byl určen odpovídajícím prvkem v poli struktur MQOR poskytnutém ve volání MQOPEN nebo MQPUT1 .

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je RCNONE.

Počáteční hodnoty

Tabulka 725. Pole v MQRR

Název pole	Název konstanty	Hodnota konstanty
RRCC	CCOK	0
RRREA	RCNONE	0

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQRR Structure
D*
D* Completion code for queue
D  RRCC                   1      4I 0 INZ(0)
D* Reason code for queue
D  RRREA                  5      8I 0 INZ(0)
```

IBM i MQSCO (volby konfigurace TLS) na systému IBM i

Struktura MQSCO (s poli TLS ve struktuře MQCD) umožňuje aplikaci spuštěné jako IBM MQ MQI client určit volby konfigurace, které řídí použití TLS pro připojení klienta, když je kanálový protokol TCP/IP.

Přehled

Účel: Struktura je vstupní parametr pro volání MQCONNX.

Není-li pro kanál klienta použit protokol TCP/IP, bude struktura MQSCO ignorována.

Znaková sada a kódování: Data v MQSCO musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT.

- “Pole” na stránce 1201
- “Počáteční hodnoty” na stránce 1206
- “Deklarace RPG” na stránce 1206

Pole

Struktura MQSCO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

SCAIC (10místné celé číslo se znaménkem)

Jedná se o počet záznamů ověřovacích informací (MQAIR) adresovaných poli *SCAIP* nebo *SCAIO* . Další informace viz“MQAIR (záznam ověřovacích informací) na IBM i” na stránce 1010. Hodnota musí být nula nebo větší. Pokud hodnota není platná, volání selže s kódem příčiny RC2383.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

SCAIO (10místné celé číslo se znaménkem)

Jedná se o posun v bajtech prvního záznamu ověřovacích informací od začátku struktury MQSCO. Posun může být kladný nebo záporný. Je-li hodnota *SCAIC* nula, pole se ignoruje.

K určení záznamů MQAIR můžete použít buď *SCAIO* , nebo *SCAIP* , ale ne obojí. Podrobnosti naleznete v popisu pole *SCAIP* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

SCAIP (10místné celé číslo se znaménkem)

Jedná se o adresu prvního záznamu ověřovacích informací. Je-li hodnota *SCAIC* nula, pole se ignoruje.

Pole záznamů MQAIR můžete poskytnout jedním ze dvou způsobů:

- Pomocí pole ukazatele *SCAIP*

  V tomto případě může aplikace deklarovat pole záznamů MQAIR, které je oddělené od struktury MQSCO, a nastavit *SCAIP* na adresu pole.

  Zvažte použití produktu *SCAIP* pro programovací jazyky, které podporují datový typ ukazatele způsobem, který je přenositelný do různých prostředí (například programovací jazyk C).

- Pomocí pole offsetu *SCAIO*

  V tomto případě musí aplikace deklarovat složenou strukturu obsahující MQSCO následovanou polem záznamů MQAIR a nastavit *SCAIO* na offset prvního záznamu v poli od začátku struktury MQSCO. Ujistěte se, že je tato hodnota správná a má hodnotu, kterou lze umístit v rámci MQLONG (nejrestriktivnějším programovacím jazykem je COBOL, pro který je platný rozsah -999 999 999 999 až +999 999 999 999).

  Zvažte použití *SCAIO* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele způsobem, který není přenositelný do různých prostředí (například programovací jazyk COBOL).

Bez ohledu na zvolenou techniku lze použít pouze jeden z *SCAIP* a *SCAIO* ; volání selže s kódem příčiny RC2384 , pokud jsou obě nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těch programovacích jazycích, které podporují ukazatele, a jinak bajtový řetězec s hodnotou Null.

Poznámka: Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

SCCERLBL (10místné celé číslo se znaménkem)

V tomto poli jsou uvedeny podrobnosti o používaném popisku certifikátu.

IBM MQ inicializuje hodnotu pro pole SCCERLBL jako mezery. Buď zadejte požadovanou hodnotu, nebo přijměte výchozí hodnotu.

`ibmwebspheremq`*`user_id`* je platná hodnota pro toto pole pro všechny verze produktu a pro verze MQSCO menší než 5.0 je to jediná platná hodnota. Proto se hodnota tohoto pole interpretuje za běhu a v případě potřeby se změní. Pokud uvedete verzi MQSCO nižší než 5.0, nebo přijmete výchozí hodnotu mezer pro pole SCCERLBL, systém použije hodnotu `ibmwebspheremq`*`user_id`*.

Toto je vstupní pole.

SCCERTVPOL (10místné celé číslo se znaménkem)

Toto pole uvádí, jaký typ zásady ověření platnosti certifikátu se používá. Pole lze nastavit na jednu z následujících hodnot:

MQ_CERT_VAL_POLICY_ANY
Použijte všechny zásady ověřování certifikátů podporované knihovnou zabezpečených soketů. Přijměte řetěz certifikátů, pokud některá ze zásad považuje řetěz certifikátů za platný.

MQ_CERT_VAL_POLICY_RFC5280
Použijte pouze zásadu ověření certifikátu kompatibilní s produktem RFC5280 . Toto nastavení poskytuje přísnější ověření než nastavení ANY, ale odmítá některé starší digitální certifikáty.

Počáteční hodnota tohoto pole je MQ_CERT_VAL_POLICY_ANY.

SCCH (10místné celé číslo se znaménkem)

V tomto poli jsou uvedeny podrobnosti o konfiguraci šifrovacího hardwaru připojeného ke klientskému systému.

Nastavte pole na řetězec v následujícím formátu, nebo jej ponechte prázdný nebo null:

```
GSK_PKCS11=the PKCS #11 driver path and file name;the PKCS #11
token label;the PKCS #11 token password;symmetric cipher setting>;
```

Chcete-li použít kryptografický hardware, který odpovídá rozhraní PKCS11 , například IBM 4960 nebo IBM 4963, zadejte cestu k ovladači PKCS11 , popisek tokenu PKCS11 a řetězce hesel tokenu PKCS11 , které jsou ukončeny středníkem.

Cesta k ovladači PKCS #11 je absolutní cesta ke sdílené knihovně poskytující podporu pro kartu PKCS #11 . Název souboru ovladače PKCS #11 je název sdílené knihovny. Příklad hodnoty požadované pro cestu a název souboru PKCS #11 je:

```
/usr/lib/pkcs11/PKCS11_API.so
```

Popisek tokenu PKCS #11 musí být zcela malými písmeny. Pokud jste nakonfigurovali hardware s popiskem tokenu se smíšenými nebo velkými písmeny, překonfigurujte jej s tímto malým popiskem.

Pokud není požadována žádná konfigurace šifrovacího hardwaru, nastavte pole na prázdnou hodnotu nebo na hodnotu null.

Pokud je hodnota kratší než délka pole, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku pole. Pokud hodnota není platná nebo vede k selhání, když se používá ke konfiguraci šifrovacího hardwaru, volání selže s kódem příčiny RC2382.

Toto je vstupní pole. Délka tohoto pole je dána LNSSCH. Počáteční hodnota tohoto pole je prázdná.

SCEPSUITEB (10místné celé číslo se znaménkem)
Toto pole určuje, zda je použito šifrování vyhovující standardu Suite B a jaká úroveň síly je použita. Hodnota může být jedna nebo více z následujících hodnot:

- SCEPSUITEB0

  Šifrování vyhovující standardu Suite B se nepoužívá.
- SCEPSUITEB1

  Používá se 128bitové zabezpečení Suite B.
- SCEPSUITEB2

  Používá se 192bitové zabezpečení síly Suite B.

Poznámka: Použití SCEPSUITEB0 s jakoukoli jinou hodnotou v tomto poli je neplatné.

SCFR (10místné celé číslo se znaménkem)

Produkt IBM MQ lze konfigurovat s šifrovacím hardwarem tak, aby používané šifrovací moduly byly ty, které poskytuje hardwarový produkt. Tyto moduly mohou být certifikovány FIPS na konkrétní úroveň v závislosti na používaném šifrovacím hardwarovém produktu.

Toto pole použijte, chcete-li uvést, že se použijí pouze algoritmy certifikované podle standardu FIPS, pokud je šifrování poskytnuto v softwaru poskytnutém produktem IBM MQ.

Když je nainstalován produkt IBM MQ , je také nainstalována implementace šifrování TLS, která poskytuje některé moduly s certifikací FIPS.

Hodnoty mohou být:

MQSSL_FIPS_NO
Toto je výchozí hodnota. Při nastavení na tuto hodnotu:

- Lze použít libovolnou specifikaci CipherSpec podporovanou na konkrétní platformě.
- Pokud se spustí bez použití šifrovacího hardwaru, následující CipherSpecs se spustí s použitím certifikovaného šifrování FIPS 140-2 na platformách IBM 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
Při nastavení na tuto hodnotu, pokud k provedení šifrování nepoužíváte kryptografický hardware, můžete si být jisti, že

- V CipherSpec , které se používají pro toto připojení klienta, lze použít pouze šifrovací algoritmy s certifikací FIPS.
- Příchozí a odchozí připojení kanálu TLS jsou úspěšná pouze v případě, že je použita jedna z následujících specifikací šifrování:
  - TLS_RSA_WITH_3DES_EDE_CBC_SHA
  - TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS_RSA_WITH_AES_256_CBC_SHA

Notes:

1. Deprecated CipherSpec TLS_RSA_WITH_3DES_EDE_CBC_SHA je zamítnuta.
2. Je-li konfigurován pouze standard FIPS CipherSpecs , klient MQI odmítne připojení, která neurčují specifikaci CipherSpec withRC2393. Produkt IBM MQ nezaručuje, že odmítne všechna taková připojení, a je vaší odpovědností určit, zda je konfigurace produktu IBM MQ kompatibilní s FIPS.

V 9.3.0 V 9.3.0 **SCKEYPWL (10místné celé číslo se znaménkem)**

Jedná se o délku přístupové fráze úložiště klíčů TLS.

Maximální délka přístupové fráze úložiště klíčů je 128 znaků. Pokud je přístupová fráze úložiště klíčů větší než maximální povolená délka, připojení se nezdaří s RC2381.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

V 9.3.0 V 9.3.0 **SCKEYPWO (10místné celé číslo se znaménkem)**

Toto je posun v bajtech přístupové fráze úložiště klíčů TLS. Posun může být kladný nebo záporný.

K určení přístupové fráze úložiště klíčů můžete použít buď SCKEYPWO , nebo SCKEYPWP , ale ne obojí. Další informace viz popis pole SCKEYPWP .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

V 9.3.0 V 9.3.0 **SCKEYPWP (ukazatel)**

Jedná se o adresu přístupové fráze úložiště klíčů TLS.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null.

Přístupovou frázi úložiště klíčů lze zadat jako prostý textový řetězec nebo přístupovou frázi, která byla zašifrována pomocí obslužného programu **runmqicred** .

Přístupová fráze úložiště klíčů uvedená pomocí tohoto pole přepíše jakoukoli přístupovou frázi úložiště klíčů uvedenou pomocí proměnné prostředí *MQKEYRPWD* nebo pomocí vlastnosti *SSLKeyRepositoryPassword* v sekci SSL konfiguračního souboru klienta.

K určení přístupové fráze úložiště klíčů můžete použít buď SCKEYPWO , nebo SCKEYPWP , ale ne obojí.

SCKR (10místné celé číslo se znaménkem)

Toto pole určuje umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty.

V 9.3.0 V 9.3.0 Není-li přípona souboru uvedena, automaticky se přidá přípona `.kdb` .

Každý soubor databáze klíčů může mít přidružený *soubor pro uložení hesla*. Tato volba uchovává šifrovaná hesla, která umožňují programový přístup k databázi klíčů. Soubor pro dočasné ukládání hesla musí být umístěn ve stejném adresáři a musí mít stejný systém souborů jako databáze klíčů a musí končit příponou `.sth`.

Pokud je například soubor databáze klíčů `/xxx/yyy/key.kdb`, pak soubor pro dočasné ukládání hesla musí být `/xxx/yyy/key.sth`, kde `xxx` a `yyy` představují názvy adresářů.

V 9.3.0 V 9.3.0 Heslo databáze klíčů lze také zadat pomocí polí *SCKEYPWP* nebo *SCKEYPWO* .

Pokud je hodnota kratší než délka pole, ukončete hodnotu znakem null nebo ji vyplní mezerami na délku pole. Hodnota není kontrolována; pokud dojde k chybě při přístupu do úložiště klíčů, volání selže s kódem příčiny RC2381.

Chcete-li spustit připojení TLS z IBM MQ MQI client, nastavte *SCKR* na platný název souboru databáze klíčů.

Toto je vstupní pole. Délka tohoto pole je dána LNSSKR. Počáteční hodnota tohoto pole je prázdný znak.

SCSID (10místné celé číslo se znaménkem)

Toto je identifikátor struktury; hodnota musí být:

SCSIDV
Identifikátor pro strukturu voleb konfigurace TLS.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SCSIDV.

SCVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury; hodnota musí být:

SCVER1
Version-1 Struktura voleb konfigurace TLS.

SCVER2
Version-2 Struktura voleb konfigurace TLS.

SCVER3
Version-3 Struktura voleb konfigurace TLS.

SCVER4
Struktura voleb konfigurace TLS Version-4 .

SCVER5
Version-5 Struktura voleb konfigurace TLS.

V 9.3.0 V 9.3.0 **SCVER6**
Version-6 Struktura voleb konfigurace TLS.

Následující konstanta určuje číslo verze aktuální verze:

SCVERC
Aktuální verze struktury voleb konfigurace TLS.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SCVER1.

Počáteční hodnoty

Tabulka 726. Pole v MQSCO		
Název pole	**Název konstanty**	**Hodnota konstanty**
SCSID	SCSIDV	'SCO¬'
SCVER	SCVER1	1
SCKR	Není	Prázdný řetězec nebo mezery
SCCH	Není	Prázdný řetězec nebo mezery
SCAIC	Není	0
SCAIO	Není	0
SCAIP	Není	Ukazatel Null nebo bajty s hodnotou Null
SCKRC	Není	Ukazatel Null nebo bajty s hodnotou Null
SCFR	Není	Ukazatel Null nebo bajty s hodnotou Null
SCEPSUITEB	Není	Ukazatel Null nebo bajty s hodnotou Null
SCCERTVPOL	Není	Ukazatel Null nebo bajty s hodnotou Null
SCCERLBL	Není	Ukazatel Null nebo bajty s hodnotou Null
V 9.3.0 V 9.3.0 *SCKEYPWP*	Není	Ukazatel Null nebo bajty s hodnotou Null
V 9.3.0 V 9.3.0 *SCKEYPWO*	Není	0
V 9.3.0 V 9.3.0 *SCKEYPWL*	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Volby SCEPSUITEB naleznete v části “Deklarace RPG” na stránce 1206 .

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQSCO Structure
D*
D* Structure identifier
D  SCSID                  1      4    INZ('SCO ')
D* Structure version number
D  SCVER                  5      8I 0 INZ(1)
D* Location of TLS key repository
D  SCKR                   9    264    INZ
```

```
     D* Cryptographic hardware configuration string
     D  SCCH                 265    520    INZ
     D* Number of MQAIR records present
     D  SCAIC                521    524I 0 INZ(0)
     D* Offset of first MQAIR record from start of MQSCO structure
     D  SCAIO                525    528I 0 INZ(0)
     D* Address of first MQAIR record
     D SCAIP                 529    544*   INZ(*NULL)
     D* Ver:1 **
     D* Number of unencrypted bytes sent/received before secret key is
     D* reset
     D SCKRC                 545    548I 0 INZ(0)
     D* Using FIPS-certified algorithms
     D SCFR                  549    552I 0 INZ(0)
     D* Ver:2 **
     * Use only Suite B cryptographic algorithms
     D SCEPSUITEB0
     D SCEPSUITEB1           553    556I 0 INZ(1)
     D SCEPSUITEB2           557    560I 0 INZ(0)
     D SCEPSUITEB3           561    564I 0 INZ(0)
     D SCEPSUITEB4           565    568I 0 INZ(0)
     D SCEPSUITEB                    10I 0 DIM(4) OVERLAY(SCEPSUITEB0)
     D* Ver:3 **
     D* Certificate validation policy
     D SCCERTVPOL            569    572I 0 INZ(0)
     D* Ver:4 **
```

IBM i MQSD (deskriptor odběru) na systému IBM i

Struktura MQSD se používá k určení podrobností o provedeného odběru.

Přehled

Účel

Struktura je vstupní/výstupní parametr volání MQSUB.

Spravované odběry

Pokud aplikace nemá specifickou potřebu používat konkrétní frontu jako místo určení pro publikování, která odpovídají jejímu odběru, může použít funkci spravovaného odběru. Pokud aplikace zvolí použití spravovaného odběru, správce front informuje odběratele o místě určení, kam jsou odesílány publikované zprávy, a to poskytnutím popisovače objektu jako výstupu z volání MQSUB. Další informace viz HOBJ (10místné celé číslo se znaménkem)-vstup/výstup.

Při odebrání odběru se správce front také zavazuje k vyčištění zpráv, které nebyly načteny ze spravovaného místa určení, v následujících situacích:

- Když je odběr odebrán-pomocí MQCLOSE s CORMSB-a spravovaný Hobj je uzavřen.
- Implicitní znamená, že dojde-li ke ztrátě připojení k aplikaci pomocí netrvalého odběru (SONDUR)
- Do vypršení platnosti, když je odběr odebrán, protože vypršela jeho platnost a spravovaný Hobj je uzavřen.

Je třeba použít spravované odběry s netrvalými odběry, aby mohlo dojít k vyčištění a aby zprávy pro uzavřené netrvalé odběry nezabívaly místo ve správci front. Trvalé odběry mohou také používat spravované cíle.

Znaková sada a kódování

Data v MQSD musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódování lokálního správce front dané ENNAT. Pokud je však aplikace spuštěna jako klient IBM MQ , musí být struktura ve znakové sadě a kódování klienta.

- “Pole” na stránce 1207
- “Počáteční hodnoty” na stránce 1219
- “Deklarace RPG” na stránce 1220

Pole

Struktura MQSD obsahuje následující pole; pole jsou popsána v abecedním pořadí:

SDAID (32bajtový znakový řetězec)

Tato hodnota je v poli *MDAID* deskriptoru zpráv (MQMD) všech zpráv publikování odpovídajících tomuto odběru. *SDAID* je součástí kontextu identity zprávy. Další informace o kontextu zprávy viz Kontext zprávy.

Další informace o *MDAID* viz MDAID.

Není-li zadána volba S0SETI , je hodnota *MDAID* , která je nastavena v každé zprávě publikované pro tento odběr, prázdná, jako výchozí informace o kontextu.

Je-li zadána volba S0SETI , je uživatelem generován soubor *SDAID* a toto pole je vstupní pole obsahující položku *MDAID* , která má být nastavena v každém publikování pro tento odběr.

Délka tohoto pole je dána hodnotou LNAIDD. Počáteční hodnota tohoto pole je 32 prázdných znaků.

Pokud měníte existující odběr pomocí volby SOALT, lze změnit hodnotu *SDAID* všech budoucích zpráv publikování.

Při návratu z volání MQSUB pomocí služby SORES je toto pole nastaveno na aktuální hodnotu *MDAID* používanou pro odběr.

SDACC (32bajtový znakový řetězec)

Tato hodnota je v poli *MDACC* deskriptoru zpráv (MQMD) všech zpráv publikování odpovídajících tomuto odběru. *MDACC* je součástí kontextu identity zprávy. Další informace o kontextu zprávy viz Kontext zprávy.

Další informace o *MDACC* viz MDACC.

Pro pole *SDACC* můžete použít následující speciální hodnotu:

ACNONE
Není uveden žádný token evidence.

Hodnota je binární nula pro délku pole.

Není-li zadána volba S0SETI , je token evidence generován správcem front jako výchozí informace o kontextu a toto pole je výstupním polem obsahujícím položku *MDACC* , která je nastavena v každé zprávě publikované pro tento odběr.

Je-li zadána volba S0SETI , je účtovací token generován uživatelem a toto pole je vstupní pole obsahující položku *MDACC* , která má být nastavena v každém publikování pro tento odběr.

Délka tohoto pole je dána LNACCT. Počáteční hodnota tohoto pole je ACNONE.

Pokud měníte existující odběr pomocí volby S0ALT , můžete změnit hodnotu *MDACC* v budoucích zprávách publikování.

Při návratu z volání MQSUB s použitím volby S0RESje toto pole nastaveno na aktuální hodnotu *MDACC* používanou pro odběr.

SDASI (40bajtový bitový řetězec)

Jedná se o identifikátor zabezpečení, který je předán s produktem *SDAU* autorizační službě, aby bylo možné provést odpovídající kontroly autorizace.

SDASI se používá pouze v případě, že je uvedeno S0ALTU a pole *SDAU* není zcela prázdné až do prvního znaku null nebo konce pole.

Při návratu z volání MQSUB s použitím volby S0RESje toto pole nezměněno.

Další informace viz popis ODASI v datovém typu MQOD.

SDAU (12bajtový znakový řetězec)

Zadáte-li hodnotu S0ALTU, bude toto pole obsahovat alternativní identifikátor uživatele, který se použije ke kontrole autorizace pro odběr a pro výstup do cílové fronty (určené v parametru **Hobj** volání MQSUB) namísto identifikátoru uživatele, pod kterým je aplikace aktuálně spuštěna.

V případě úspěchu je identifikátor uživatele uvedený v tomto poli zaznamenán jako identifikátor uživatele vlastnící odběr namísto identifikátoru uživatele, pod kterým je aplikace aktuálně spuštěna.

Je-li zadána hodnota SOALTU a toto pole je zcela prázdné až do prvního znaku null nebo konce pole, může být odběr úspěšný pouze v případě, že pro přihlášení k odběru tohoto tématu s určenými volbami nebo cílovou frontou pro výstup není potřebná žádná autorizace uživatele.

Není-li zadána hodnota SOALTU , bude toto pole ignorováno.

Při návratu z volání MQSUB s použitím volby SORESje toto pole nezměněno.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou LNUID. Počáteční hodnota tohoto pole je 12 prázdných znaků.

SDCID (24bajtový bitový řetězec)

Všechna publikování odeslaná za účelem shody s tímto odběrem obsahují tento identifikátor korelace v deskriptoru zprávy. Pokud více odběrů používá stejnou frontu k získání svých publikací, použití MQGET podle ID korelace umožňuje získat pouze publikování pro konkrétní odběr. Tento identifikátor korelace může být generován správcem front nebo uživatelem.

Není-li zadána volba SOSCID , je identifikátor korelace generován správcem front a toto pole je výstupním polem obsahujícím identifikátor korelace, který je nastaven v každé zprávě publikované pro tento odběr.

Pokud je zadána volba SOSCID , je identifikátor korelace generován uživatelem a toto pole je vstupní pole obsahující identifikátor korelace, který má být nastaven v každém publikování pro tento odběr. V tomto případě, pokud pole obsahuje CINONE, identifikátor korelace, který je nastaven v každé zprávě publikované pro tento odběr, je identifikátorem korelace, který byl vytvořen původním vložením zprávy.

Pokud je zadána volba SOGRP a zadaný identifikátor korelace je stejný jako existující seskupený odběr používající stejnou frontu a překrývající se řetězec tématu, je s kopií publikování poskytnut pouze nejvýznamnější odběr ve skupině.

Délka tohoto pole je dána hodnotou LNCID. Počáteční hodnota tohoto pole je CINONE.

Pokud měníte existující odběr pomocí volby SOALT a toto pole je vstupní pole, lze ID korelace odběru změnit, pokud nebyl odběr vytvořen pomocí volby SOGRP .

Při návratu z volání MQSUB s použitím volby SORESje toto pole nastaveno na aktuální ID korelace pro odběr.

SDEXP (10místné celé číslo se znaménkem)

Jedná se o dobu vyjádřenou v desetinách sekundy, po jejímž uplynutí vyprší platnost odběru. Po uplynutí tohoto intervalu nebudou tomuto odběru odpovídat žádná další publikování. Používá se také jako hodnota v poli *MDEXP* v deskriptoru MQMD publikování odeslaných tomuto odběrateli.

Je rozpoznána následující speciální hodnota:

EIULIM
Předplatné má neomezenou dobu vypršení platnosti.

Pokud měníte existující odběr pomocí volby SOALT , můžete změnit vypršení platnosti odběru.

Při návratu z volání MQSUB pomocí volby SORES je toto pole nastaveno na původní vypršení platnosti odběru, nikoli na zbývající dobu vypršení platnosti.

SDON (48bajtový znakový řetězec)

Jedná se o název objektu tématu, jak je definován v lokálním správci front.

Název může obsahovat následující znaky:

- Velká písmena abecedy (A až Z)
- Malá písmena abecedy (a až z)

- Číselné číslice (0 až 9)
- Tečka (.), dopředné lomítko (/), podtržítko (_), procento (%)

Název nesmí obsahovat úvodní ani vložené mezery, ale může obsahovat koncové mezery. Použijte znak null, abyste označili konec důležitých dat v názvu; hodnota null a všechny následující znaky jsou považovány za mezery. Platí následující omezení:

- V systémech, které používají EBCDIC Katakana, nelze použít malá písmena.
- Názvy obsahující malá písmena, dopředné lomítko nebo procenta musí být uzavřeny v uvozovkách, pokud jsou zadány v příkazech. Tyto uvozovky nesmí být uvedeny pro názvy, které se vyskytují jako pole ve strukturách nebo jako parametry ve voláních.

SDON se používá k vytvoření úplného názvu tématu.

Úplný název tématu lze sestavit ze dvou různých polí: *SDON* a *SDOS*. Podrobnosti o použití těchto dvou polí naleznete v tématu Kombinace řetězců témat.

Při návratu z volání MQSUB pomocí volby SORES je toto pole nezměněno.

Délka tohoto pole je dána hodnotou LNTOPN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

Pokud měníte existující odběr pomocí volby SDALT, nelze název objektu tématu, který je přihlášen k odběru, změnit. Toto pole a *SDOS* lze vynechat. Pokud jsou poskytnuty, musí se přeložit na stejný úplný název tématu, jinak volání selže s RC2510 .

SDOPT (10místné celé číslo se znaménkem)

Musíte uvést alespoň jednu z následujících voleb:

- SOALT
- SORES
- SOCRT

Hodnoty lze přidat. Nepřidávejte stejnou konstantu více než jednou. Tabulka ukazuje, jak můžete kombinovat tyto volby: Jsou zaznamenány neplatné kombinace; všechny ostatní kombinace jsou platné.

Volby přístupu nebo vytvoření

Volby přístupu a vytvoření určují, zda má být vytvořen odběr nebo zda má být vrácen či změněn existující odběr. Musíte uvést alespoň jednu z těchto voleb. Tabulka zobrazuje platné kombinace voleb přístupu nebo vytvoření.

Tabulka 727. Platné kombinace voleb přístupu a vytvoření

Kombinace voleb	**Notes**
SOCRT	Vytvoří odběr, pokud neexistuje; pokud odběr existuje, dojde k selhání.
SORES	Obnoví existující odběr. Pokud žádný odběr neexistuje, dojde k selhání.
SOCRT + VŘEDY	Vytvoří odběr, pokud neexistuje, a obnoví odpovídající odběr, pokud existuje. Užitečná kombinace, pokud se používá v aplikaci, která může být spuštěna vícekrát.
SORES + SOALT (viz poznámka)	Pokud neexistuje žádný odběr, obnoví existující odběr a změní pole tak, aby odpovídala polím určeným v modulu MQSD, dojde k selhání.

Tabulka 727. Platné kombinace voleb přístupu a vytvoření (pokračování)	
Kombinace voleb	**Notes**
SOCRT + SOALT (viz poznámka)	Vytvoří odběr, pokud neexistuje, a obnoví odpovídající odběr, pokud existuje, a změní libovolná pole tak, aby odpovídala polím určeným v modulu MQSD. Užitečná kombinace, pokud se používá v aplikaci, která chce zajistit, aby se její odběr nacházil v určitém stavu, než bude pokračovat.

Poznámka:

Volby určující hodnotu S0ALT mohou také určovat hodnotu S0RES, ale tato kombinace nemá žádný další vliv na samotné určení hodnoty S0ALT . S0ALT znamená S0RES, protože volání MQSUB za účelem změny odběru znamená, že odběry jsou také obnoveny. Opak není pravdou, nicméně: obnovení předplatného neznamená, že má být změněno.

SOCRT

Vytvořte odběr pro zadané téma. Pokud existuje odběr používající stejné *SDSN* , volání se nezdaří s RC2432 . Tomuto selhání se lze vyhnout kombinací volby S0CRT s volbou S0RES. Parametr *SDSN* není vždy nezbytný. Další podrobnosti viz popis tohoto pole.

Kombinace S0CRT a S0RES nejprve zkontroluje, zda pro zadaný *SDSN*existuje existující odběr, a pokud pro tento již existující odběr existuje manipulátor; pokud však neexistuje žádný existující odběr, vytvoří se nový odběr pomocí všech polí poskytnutých v MQSD.

S0CRT lze také kombinovat s S0ALT s podobným účinkem (viz podrobnosti o S0ALT později v tomto tématu).

SORES

Vraťte manipulátor k již existujícímu odběru, který odpovídá předplatnému zadanému parametrem *SDSN*. V odpovídajících atributech odběru nejsou provedeny žádné změny a jsou vráceny na výstupu ve struktuře MQSD. Většina obsahu disku MQSD se nepoužívá: použitá pole jsou *SDSID*, *SDVER*, *SDOPT*, *SDAID* a *SDASI*a *SDSN*.

Volání selže s kódem příčiny RC2428 , pokud neexistuje odběr odpovídající úplnému názvu odběru. Tomuto selhání se lze vyhnout kombinací volby S0CRT s volbou S0RES. Podrobnosti o S0CRTviz SOCRT.

ID uživatele odběru je ID uživatele, který vytvořil odběr, nebo pokud byl později změněn jiným ID uživatele, jedná se o ID uživatele poslední úspěšné změny. Je-li použit parametr *SDAID* a pro tohoto uživatele je povoleno použití alternativních ID uživatelů, je produkt *SDAID* zaznamenán jako ID uživatele, který vytvořil odběr, namísto ID uživatele, pod kterým byl odběr proveden.

ID uživatele, který vytvořil odběr, se zaznamená jako *SDAU* , pokud je toto pole použito, a pro tohoto uživatele je povoleno použití alternativních ID uživatelů.

Pokud existuje odpovídající odběr, který byl vytvořen bez volby S0AUID , a ID uživatele odběru se liší od ID aplikace, která požaduje manipulátor pro odběr, volání se nezdaří s kódem příčiny RC2434 .

Pokud existuje odpovídající odběr a je aktuálně používán jinou aplikací, volání se nezdaří s kódem příčiny RC2429 . Pokud je aktuálně používán stejným připojením, volání neselže a vrátí se obsluha odběru.

Pokud odběr uvedený v poli SubName není platným odběrem pro obnovení nebo změnu z aplikace, volání selže s volbou RC2523 .

Volba S0RES je odvozena od hodnoty S0ALT , a proto není nutné ji kombinovat s touto volbou, avšak pokud jsou tyto dvě volby zkombinovány, nejedná se o chybu.

ALT

Vraťte manipulátor k již existujícímu odběru s úplným názvem odběru, který odpovídá názvu uvedenému v souboru *SDSN*. Jakékoli atributy odběru, které se liší od atributů určených v modulu MQSD, jsou v odběru změněny, pokud není změna pro tento atribut zakázána. Podrobnosti jsou uvedeny v popisu každého atributu a jsou shrnuty v následující tabulce. Pokud se pokusíte změnit atribut, který nelze změnit, volání selže s kódem příčiny uvedeným v následující tabulce.

Volání selže s kódem příčiny RC2428 , pokud neexistuje odběr odpovídající úplnému názvu odběru. Tomuto selhání se lze vyhnout kombinací volby S0CRT s volbou S0ALT.

Kombinace S0CRT a S0ALT nejprve zkontroluje, zda existuje odběr pro zadaný úplný název odběru, a pokud se k tomuto existujícímu odběru vrátí manipulátor se změnami provedenými podle předchozího popisu; pokud však neexistuje žádný odběr, vytvoří se nový odběr pomocí všech polí uvedených v tabulce MQSD.

ID uživatele odběru je ID uživatele, který vytvořil odběr, nebo pokud byl později změněn jiným ID uživatele, jedná se o ID uživatele poslední úspěšné změny. Je-li použit parametr *SDAU* (a je-li pro tohoto uživatele povoleno použití alternativních ID uživatelů), bude alternativní ID uživatele zaznamenáno jako ID uživatele, který vytvořil odběr, namísto ID uživatele, pod kterým byl odběr vytvořen.

Pokud existuje odpovídající odběr, který byl vytvořen bez volby S0AUID , a ID uživatele odběru se liší od ID aplikace požadující manipulátor pro odběr, volání se nezdaří s kódem příčiny RC2434 .

Pokud existuje odpovídající odběr a je aktuálně používán jinou aplikací, volání selže s volbou RC2429 . Pokud je aktuálně používán stejným připojením, volání neselže a vrátí se obsluha odběru.

Pokud odběr uvedený v poli SubName není platným odběrem pro obnovení nebo změnu z aplikace, volání selže s volbou RC2523 .

V následujících tabulkách jsou uvedeny atributy odběru, které lze změnit pomocí volby S0ALT.

Tabulka 728. Atributy v MQSD a MQSUB, které lze změnit

Deskriptor datového typu nebo volání funkce	Název pole	Lze tento atribut změnit pomocí hodnoty SOALT?	Kód příčiny
MQSD	Možnosti trvanlivosti	Ne	RC2509
MQSD	Volby cíle	Ano	Není
MQSD	Volby registrace	Ano (viz poznámka 1)	RC2515 , pokud se pokusíte změnit SOGRP
MQSD	Volby publikování	Ano (viz poznámka 2)	Není
MQSD	Volby zástupných znaků	Ne	RC2510
MQSD	Další volby	Ne (viz poznámka 3)	Není
MQSD	ObjectName	Ne	RC2510
MQSD	SDAU	Ne (viz poznámka 4)	Není
MQSD	SDASI	Ne (viz poznámka 4)	Není
MQSD	SDEXP	Ano	Není
MQSD	SDOS	Ne	RC2510
MQSD	SDSN	Ne (viz poznámka 5)	Není
MQSD	SDSUD	Ano	Není
MQSD	SDCID	Ano (viz poznámka 6)	RC2515 v seskupeném odběru

Tabulka 728. Atributy v MQSD a MQSUB, které lze změnit (pokračování)			
Deskriptor datového typu nebo volání funkce	**Název pole**	**Lze tento atribut změnit pomocí hodnoty SOALT?**	**Kód příčiny**
MQSD	SDPRI	Ano	Není
MQSD	SDACC	Ano	Není
MQSD	SDAID	Ano	Není
MQSD	SDSL	Ne	RC2512
MQSUB	HOBJ	Ano (viz poznámka 6)	RC2515 v seskupeném odběru

Notes:

1. SOGRP nelze změnit.
2. Položku SONEWP nelze změnit, protože není součástí odběru.
3. Tyto volby nejsou součástí odběru
4. Tento atribut není součástí odběru
5. Tento atribut je identitou měněného odběru
6. Lze změnit s výjimkou případů, kdy je část seskupeného podsouboru (SOGRP)

Volby trvanlivosti: Následující volby řídí trvanlivost odběru. Můžete uvést pouze jednu z těchto voleb. Pokud měníte existující odběr pomocí volby SOALT , nemůžete změnit trvanlivost odběru. Při návratu z volání MQSUB s použitím volby SORESje nastavena příslušná volba trvanlivosti.

SODUR
Požádejte o to, aby odběr tohoto tématu zůstal, dokud nebude explicitně odebrán pomocí příkazu MQCLOSE s volbou CORMSB . Není-li tento odběr explicitně odebrán, zůstane zachován i po připojení této aplikace ke správci front.

Pokud je trvalý odběr požadován pro téma, které je definováno jako nepovolující trvalé odběry, volání selže s volbou RC2436 .

SONDUR
Požadavek na odebrání odběru tohoto tématu při zavření připojení aplikace ke správci front, pokud dosud nebyl explicitně odebrán. SONDUR je opak volby SODUR a je definován jako pomůcka pro programovou dokumentaci. Jedná se o předvolbu, není-li zadán ani jeden z nich.

Volby místa určení: Následující volby řídí místo určení, kam jsou odesílána publikování pro téma, které bylo přihlášeno k odběru. Pokud měníte existující odběr pomocí volby SOALT, můžete změnit místo určení použité pro publikování pro odběr. Při návratu z volání MQSUB pomocí volby SORESje tato volba nastavena, je-li to vhodné.

SOMAN-nestátní

Požadujte, aby místo určení, do kterého jsou odesílána publikování, bylo spravováno správcem front.

Popisovač objektu vrácený v produktu *HOBJ* představuje spravovanou frontu správce front a je určen pro použití s následnými voláními MQGET, MQCB, MQINQ nebo MQCLOSE.

Popisovač objektu vrácený z předchozího volání MQSUB nelze v parametru **Hobj** zadat, není-li zadán parametr SOMAN .

Volby registrace: Podrobnosti registrace provedené ve správci front pro tento odběr řídí následující volby. Pokud měníte existující odběr pomocí volby SOALT , lze tyto volby registrace změnit. Při návratu z volání MQSUB s použitím volby SORES jsou nastaveny příslušné volby registrace.

SOGRP

Tento odběr je seskupen s jinými odběry stejného produktu *SDSL* používajícími stejnou frontu a se stejným ID korelace tak, aby všechna publikování v tématech, která by mohla způsobit, že bude skupině odběrů poskytnuta více než jedna zpráva publikování, v důsledku použití překrývající se sady řetězců témat, byla do fronty doručena pouze jedna zpráva. Není-li tato volba použita, bude každý jedinečný odběr (identifikovaný jako *SDSN*), který odpovídá, poskytnut s kopií publikování, což může znamenat, že do fronty sdílené několika odběry může být umístěna více než jedna kopie publikování.

Pouze nejvýznamnější předplatné ve skupině je poskytováno s kopií publikace. Nejvýznamnější odběr je založen na úplném názvu tématu až do okamžiku, kdy je nalezen zástupný znak. Pokud je ve skupině použita směs schémat zástupných znaků, je důležitá pouze pozice zástupného znaku. Doporučuje se nekombinovat různá schémata zástupných znaků ve skupině odběrů, které sdílejí stejnou frontu.

Při vytváření nového seskupeného odběru musí mít stále jedinečný *SDSN*, ale pokud se shoduje s úplným názvem tématu existujícího odběru ve skupině, volání selže s RC2514 .

Pokud nejvýznamnější odběr ve skupině také uvádí SONOLC a jedná se o publikování ze stejné aplikace, pak se do fronty nedoručí žádné publikování.

Při změně odběru provedeného pomocí této volby nelze změnit pole, která implikují seskupení, *Hobj* ve volání MQSUB (reprezentující název fronty a správce front) a *SDCID* . Pokus o jejich změnu způsobí selhání volání s RC2515 .

Tato volba musí být kombinována s volbou SOSCID s položkou *SDCID* , která není nastavena na hodnotu CINONE, a nelze ji kombinovat s volbou SOMAN.

SOAUID

Je-li zadána volba SOAUID , není identita odběratele omezena na jediné ID uživatele. To umožňuje každému uživateli změnit nebo obnovit odběr, pokud má odpovídající oprávnění. Odběr může mít v daném okamžiku pouze jeden uživatel. Pokus o obnovení použití odběru, který je aktuálně používán jinou aplikací, způsobí selhání volání s produktem RC2429 .

Chcete-li přidat tuto volbu k existujícímu odběru, musí volání MQSUB s použitím volby SOALTpocházet ze stejného ID uživatele jako samotný původní odběr.

Pokud volání MQSUB odkazuje na existující odběr s nastavenou volbou SOAUID a ID uživatele se liší od původního odběru, bude volání úspěšné pouze v případě, že nové ID uživatele má oprávnění přihlásit se k odběru tématu. Po úspěšném dokončení jsou budoucí publikování pro tohoto odběratele vložena do fronty odběratele s novým ID uživatele nastaveným ve zprávě publikování.

Neuvádějte současně SOAUID a SOFUID. Není-li uveden ani jeden, předvolba je SOFUID.

SOFUID

Je-li zadána volba SOFUID , může být odběr změněn nebo obnoven pouze posledním ID uživatele, aby se změnil odběr. Pokud odběr nebyl změněn, jedná se o ID uživatele, který jej vytvořil.

Pokud příkaz MQSUB odkazuje na existující odběr s nastavenou hodnotou SOAUID a změní odběr pomocí SOALT tak, aby používal SOFUID, bude ID uživatele odběru nyní opraveno na tomto novém ID uživatele. Volání je úspěšné pouze v případě, že nové ID uživatele má oprávnění přihlásit se k odběru tématu.

Pokud se jiné ID uživatele, než které bylo zaznamenáno jako vlastník odběru, pokusí obnovit nebo změnit odběr SOFUID , volání selže s RC2434 . ID vlastnícího uživatele odběru lze zobrazit pomocí příkazu **DISPLAY SBSTATUS** .

Neuvádějte současně SOAUID a SOFUID. Není-li uveden ani jeden, předvolba je SOFUID.

Volby publikování: Způsob odesílání publikací tomuto odběrateli řídí následující volby. Pokud měníte existující odběr pomocí volby SOALT , můžete tyto volby publikování změnit.

SONOLC
Zprostředkovateli sdělí, že aplikace nechce zobrazit žádné vlastní publikace. Publikace jsou považovány za pocházející ze stejné aplikace, pokud jsou manipulátory připojení stejné. Při návratu z volání MQSUB s použitím volby SORES je tato volba nastavena, je-li to vhodné.

SONEWP-pracovní
Při vytvoření tohoto odběru nejsou odesílána žádná zachovaná publikování, pouze nová publikování. Tato volba platí pouze v případě, že je zadána volba SOCRE . Žádné následné změny v odběru nemění tok publikování, a proto všechna publikování, která byla uchována v tématu, již byla odeslána odběrateli jako nová publikování.

Je-li tato volba zadána bez parametru SOCRE , způsobí selhání volání s parametrem RC2046 . Při návratu z volání MQSUB s použitím volby SORES není tato volba nastavena ani v případě, že byl odběr vytvořen s použitím této volby.

Není-li tato volba použita, jsou dříve uchované zprávy odesílány do zadané cílové fronty. Pokud se tato akce nezdaří v důsledku chyby RC2525 nebo RC2526 , vytvoření odběru se nezdaří.

Tato volba není platná v kombinaci s volbou SOPUBR.

SOPUBR
Nastavení této volby označuje, že odběratel požaduje informace specificky v případě potřeby. Správce front neodesílá nevyžádané zprávy odběrateli. Zachované publikování (nebo případně více publikování, je-li v tématu uveden zástupný znak) je odesláno odběrateli při každém volání MQSUBRQ s použitím manipulátoru Hsub z předchozího volání MQSUB. V důsledku volání MQSUB pomocí této volby nejsou odesílána žádná publikování. Při návratu z volání MQSUB s použitím volby SORES je tato volba nastavena, je-li to vhodné.

Tato volba není platná v kombinaci s volbou SONEWP.

Volby se zástupnými znaky: Následující volby řídí způsob interpretace zástupných znaků v řetězci zadaném v poli *SDOS* disku MQSD. Můžete uvést pouze jednu z těchto voleb. Pokud měníte existující odběr pomocí volby SOALT , nelze tyto volby se zástupnými znaky změnit. Při návratu z volání MQSUB s použitím volby SORES je nastavena příslušná volba zástupného znaku.

SOWCHR
Zástupné znaky pracují pouze se znaky v řetězci tématu. Pole SOWCHR zachází s dopředným lomítkem (/) jako s jiným znakem bez zvláštního významu.

Chování definované pomocí SOWCHR je uvedeno v následující tabulce:

Tabulka 729. Způsob interpretace zástupných znaků

Speciální znak	**Chování**
*	Zástupný znak, žádný nebo více znaků
?	Zástupný znak, jeden znak
%	Znak změny významu umožňuje použití znaků '*', '?' nebo '%' v řetězci a nelze jej interpretovat jako speciální znak, například '% *', '%?' nebo' %%% '.

Například publikování na následující téma:

```
/level0/level1/level2/level3/level4
```

odpovídá odběratelům pomocí následujících témat:

```
*
/*
/ level0/level1/level2/level3/*
/ level0/level1/*/level3/level4
/ level0/level1/le?el2/level3/level4
```

Poznámka: Toto použití zástupných znaků přesně určuje význam uvedený v části IBM MQ V6 a WebSphere MB V6 při použití zpráv ve formátu MQRFH1 pro publikování/odběr. Doporučuje se, aby se toto nepoužívalo pro nově napsané aplikace a používalo se pouze pro aplikace, které byly dříve spuštěny pro tuto verzi a nebyly změněny tak, aby používaly výchozí chování zástupných znaků, jak je popsáno v tématu SOWTOP.

SOWTOP

Zástupné znaky pracují pouze s prvky tématu v řetězci tématu. Toto je výchozí chování, pokud není vybráno.

Chování požadované příkazem SOWTOP je uvedeno v následující tabulce:

Tabulka 730. Způsob interpretace zástupných znaků

Speciální znak	Chování
/	Oddělovač úrovně tématu
#	Zástupný znak: více úrovní tématu
+	Zástupný znak: úroveň jednoho tématu

Poznámka:

Znaky '+' a '#' nejsou považovány za zástupné znaky, pokud jsou v rámci úrovně tématu smíchány s jinými znaky (včetně sebe). V následujícím řetězci jsou znaky '#' a '+' považovány za běžné znaky.

```
level0/level1/#+/level3/level#
```

Například publikování na následující téma:

```
/level0/level1/level2/level3/level4
```

odpovídá odběratelům pomocí následujících témat:

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1/+/level3/level4
```

Poznámka: Toto použití zástupných znaků poskytuje význam poskytnutý v produktu WebSphere Message Broker 6 při použití zpráv ve formátu MQRFH2 pro publikování/odběr.

Další volby: Následující volby řídí způsob, jakým je volání rozhraní API vydáváno, a nikoli odběr. Při návratu z volání MQSUB s použitím volby SORES jsou tyto volby nezměněny.

SOALTU
Pole SDAU obsahuje identifikátor uživatele, který má být použit k ověření tohoto volání MQSUB. Volání může být úspěšné, pouze pokud je tato SDAU autorizována k otevření objektu s uvedenými volbami přístupu, bez ohledu na to, zda je k tomu autorizován identifikátor uživatele, pod kterým je aplikace spuštěna.

Identifikátor SOSCID
Odběr má používat identifikátor korelace zadaný v poli *SDCID* . Není-li tato volba určena, správce front v době odběru automaticky vytvoří identifikátor korelace a vrátí jej aplikaci v poli *SDCID* . Další informace viz SDCID (24bajtový bitový řetězec) SDCID .

SOSETI (nastavení)

Odběr bude používat údaje o tokenu evidence a identitě aplikace zadané v polích *SDACC* a *SDAID* .

Je-li zadána tato volba, provede se stejná kontrola autorizace, jako kdyby k cílové frontě bylo přistupováno pomocí volání MQOPEN s volbou OOSETI, s výjimkou případu, kdy je také

použita volba SOMAN . V takovém případě není v cílové frontě provedena žádná kontrola autorizace.

Není-li tato volba určena, publikace odeslané tomuto odběrateli mají k sobě přidruženy výchozí informace o kontextu:

Tabulka 731. Výchozí informace o kontextu pro publikování odeslaná tomuto odběrateli

Pole v deskriptoru MQMD	Použitá hodnota
MDUID	ID uživatele přidružené k odběru v době, kdy byl odběr proveden.
MDACC	Určuje se z prostředí, je-li to možné; nastavte na ACNONE, pokud ne.
MDAID	Nastavit na mezery

Tato volba je platná pouze s volbou SOCRE a SOALT. Pokud se používá s SORES, pole *SDACC* a *SDAID* se ignorují, takže tato volba nemá žádný účinek.

Je-li odběr změněn bez použití této volby, kde dříve předplatil informace o kontextu identity, budou pro změněný odběr vygenerovány výchozí informace o kontextu.

Pokud je odběr, který umožňuje použití různých ID uživatelů s volbou SOAUID, obnoven jiným ID uživatele, vygeneruje se výchozí kontext identity pro nové ID uživatele, které nyní vlastní odběr, a všechna následná publikování budou doručena obsahující nový kontext identity.

SOFIQ
Volání MQSUB se nezdaří, pokud je správce front ve stavu uvedení do klidového stavu. V systému z/OSpro aplikaci CICS nebo IMS tato volba také vynutí selhání volání MQSUB, pokud je připojení ve stavu uvedení do klidového stavu.

SDAU (12bajtový znakový řetězec)

Zadáte-li hodnotu SOALTU, bude toto pole obsahovat alternativní identifikátor uživatele, který se použije ke kontrole autorizace pro odběr a pro výstup do cílové fronty (určené v parametru **Hobj** volání MQSUB) namísto identifikátoru uživatele, pod kterým je aplikace aktuálně spuštěna.

V případě úspěchu je identifikátor uživatele uvedený v tomto poli zaznamenán jako identifikátor uživatele vlastnící odběr namísto identifikátoru uživatele, pod kterým je aplikace aktuálně spuštěna.

Je-li zadána hodnota SOALTU a toto pole je zcela prázdné až do prvního znaku null nebo konce pole, může být odběr úspěšný pouze v případě, že se žádná autorizace uživatele nesmí přihlásit k odběru tohoto tématu se zadanými volbami nebo cílovou frontou pro výstup.

Není-li zadána hodnota SOALTU , bude toto pole ignorováno.

Při návratu z volání MQSUB s použitím volby SORESje toto pole nezměněno.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou LNUID. Počáteční hodnota tohoto pole je 12 prázdných znaků.

SDPRI (celé číslo se znaménkem 10 číslic)

Jedná se o hodnotu, která je v poli *MQPRI* deskriptoru zpráv (MQMD) všech zpráv publikování odpovídajících tomuto odběru. Další informace o poli *MQPRI* v deskriptoru MQMD viz MDPRI.

Hodnota musí být větší nebo rovna nule; nula je nejnižší priorita. Lze také použít následující speciální hodnoty:

PRQDEF

Pokud je v poli Hobj ve volání MQSUB zadána fronta odběrů a nejedná se o spravovaný popisovač, bude priorita zprávy převzata z atributu **DefPriority** této fronty. Je-li takto identifikovaná fronta frontou klastru nebo je-li v cestě k rozlišení názvu fronty více než jedna definice, bude priorita určena při vložení zprávy publikování do fronty, jak je popsáno v tématu MDPRI.

Pokud volání MQSUB používá spravovaný popisovač, priorita zprávy je převzata z atributu **DefPriority** modelové fronty přidružené k odebíranému tématu.

PRPUB
Prioritou zprávy je priorita původní publikace. Toto je počáteční hodnota pole.

Pokud měníte existující odběr pomocí volby SOALT , můžete změnit *MQPRI* všech budoucích zpráv publikování.

Při návratu z volání MQSUB s použitím volby SORESje toto pole nastaveno na aktuální prioritu používanou pro odběr.

SDRO (MQCHARV)

SDRO je dlouhý název objektu poté, co správce front přeloží název uvedený v části *SDON*.

Pokud je dlouhý název objektu uveden v souboru *SDOS* a v souboru *SDON*není nic uvedeno, hodnota vrácená v tomto poli je stejná jako v souboru *SDOS*.

Pokud je toto pole vynecháno (tj. SDRO.VSBufSize je nula), hodnota *SDRO* není vrácena, ale délka je vrácena v SDRO.VSLength. Pokud je délka kratší než úplná hodnota *SDRO*, je oříznuta a vrací tolik znaků nejvíce vpravo, kolik se vejde do uvedené délky.

Pokud je parametr *SDRO* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV nebo pokud překročí maximální délku, volání se nezdaří s kódem příčiny RC2520 .

SDSID (4bajtový znakový řetězec)

Toto je identifikátor struktury; hodnota musí být:

SDSIDV
Identifikátor pro strukturu deskriptoru odběru.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SDSIDV .

SDSL (10místné celé číslo se znaménkem)

Toto je úroveň přidružená k odběru. Publikování jsou pro tento odběr doručena pouze v případě, že je v sadě odběrů s nejvyšší hodnotou *SDSL* menší nebo rovnou hodnotě PubLevel použité v době publikování.

Hodnota musí být v rozsahu od 0 do 9. Nula je nejnižší úroveň.

Počáteční hodnota tohoto pole je 1.

Pokud měníte existující odběr pomocí volby SOALT , pak *SDSL* nelze změnit.

SDSN (MQCHARV)

SDSN určuje název odběru.

Toto pole je povinné pouze v případě, že parametr *SDOPT* uvádí volbu SODUR , ale pokud je poskytnuta, je také použita správcem front pro SONDUR . Je-li zadána volba *SDSN* , musí být v rámci správce front jedinečná, protože se jedná o pole používané k identifikaci odběrů.

Maximální délka *SDSN* je 10240.

Toto pole slouží dvěma účelům. V případě odběru SODUR se jedná o způsob, jakým identifikujete odběr k jeho obnovení po jeho vytvoření, pokud jste buď zavřeli manipulátor pro odběr (pomocí volby COKPSB), nebo jste byli odpojeni od správce front. Identifikace odběru, který má být odebrán po jeho vytvoření, se provádí pomocí volání MQSUB s volbou SORES . Pole SDSN se také zobrazí v pohledu administrace odběrů v poli *SDSN* v části DISPLAY SBSTATUS.

Pokud je parametr *SDSN* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překračuje maximální délku, nebo pokud je vynechán v případě, že je vyžadován (tj. *SDSN*. *VCHRL* je nula), nebo pokud překročí maximální délku, volání selže s kódem příčiny RC2440 .

Toto je vstupní pole. Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQCHARV.

Pokud měníte existující odběr pomocí volby S0ALT , název odběru nelze změnit, protože se jedná o pole používané k identifikaci odběru. Při výstupu volání MQSUB s volbou S0RES se nezmění.

SDSS (MQCHARV)

SDSS je řetězec, který poskytuje kritéria výběru použitá při přihlašování k odběru zpráv z tématu.

Toto pole s proměnnou délkou je vráceno na výstupu volání MQSUB pomocí volby S0RES , pokud je poskytnuta vyrovnávací paměť a pokud je také kladná délka vyrovnávací paměti v VSBufSize. Není-li ve volání poskytnuta žádná vyrovnávací paměť, vrátí se v poli VSLength hodnoty MQCHARV pouze délka řetězce výběru. Je-li poskytnutá vyrovnávací paměť menší než prostor požadovaný pro vrácení pole, jsou v poskytnuté vyrovnávací paměti vráceny pouze bajty VSBufSize .

Pokud je parametr *SDSS* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV nebo pokud překročí maximální délku, volání se nezdaří s kódem příčiny RC2519 .

SDSUD (MQCHARV)

Data poskytnutá v rámci odběru v tomto poli jsou zahrnuta jako vlastnost zprávy mq.SubUserData pro každé publikování odeslané v rámci tohoto odběru.

Maximální délka *SDSUD* je 10240.

Pokud je parametr *SDSUD* zadán nesprávně, podle popisu způsobu použití struktury MQCHARV , nebo pokud překročí maximální délku, volání selže s kódem příčiny RC2431.

Toto je vstupní pole. Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQCHARV.

Při změně existujícího odběru pomocí volby S0ALT lze uživatelská data odběru změnit.

Toto pole s proměnnou délkou je vráceno na výstupu volání MQSUB pomocí volby S0RES , pokud je poskytnuta vyrovnávací paměť a v souboru *VSBufLen*je kladná délka vyrovnávací paměti. Není-li pro volání poskytnuta žádná vyrovnávací paměť, vrátí se v poli *VCHRL* atributu MQCHARV pouze délka uživatelských dat odběru. Pokud je poskytnutá vyrovnávací paměť menší než prostor požadovaný pro vrácení pole, vrátí se v poskytnuté vyrovnávací paměti pouze *VSBufLen* bajtů.

SDVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury; hodnota musí být:

SDVER1
Version-1 Struktura deskriptoru odběru.

Následující konstanta určuje číslo verze aktuální verze:

SDVERC
Aktuální verze struktury deskriptoru odběru.

Toto je vždy vstupní pole. Počáteční hodnota pole je SDVER1.

Počáteční hodnoty

Tabulka 732. Pole v MQSD

Název pole	Název konstanty	Hodnota konstanty
SDSID	SDSIDV	'SD¬¬'
SDVER	SDVER1	1
SDOPT	SONDUR	0
SDON	Není	Mezery
SDAU	Není	Mezery
SDASI	SINONE	Hodnoty null

Tabulka 732. Pole v MQSD (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
SDEXP	EIULIM	-1
SDOS	Názvy a hodnoty definované pro MQCHARV	
SDSN	Názvy a hodnoty definované pro MQCHARV	
SDSUD	Názvy a hodnoty definované pro MQCHARV	
SDCID	CINONE	Hodnoty null
SDPRI	PRQDEF	-3
SDACC	ACNONE	Hodnoty null
SDAID	Není	Mezery
SDSL	Není	1
SDRO	Názvy a hodnoty definované v MQCHARV	
Poznámka: 1. Symbol ¬ představuje jeden prázdný znak.		

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQSD Structure
D*
D* Structure identifier
D  SDSID                  1      4
D* Structure version number
D  SDVER                  5      8I 0
D* Options associated with subscribing
D  SDOPT                  9     12I 0
D* Object name
D  SDON                  13     60
D* Alternate user identifier
D  SDAU                  61     72
D* Alternate security identifier
D  SDASI                 73    112
D* Expiry of Subscription
D  SDEXP                113    116I 0
D* Object Long name
D SDOSP                 117    132*
D SDOSO                 133    136I 0
D SDOSS                 137    140I 0
D SDOSL                 141    144I 0
D SDOSC                 145    148I 0
D* Subscription name
D SDSNP                 149    164*
D SDSNO                 165    168I 0
D SDSNS                 169    172I 0
D SDSNL                 173    176I 0
D SDSNC                 177    180I 0
D* Subscription User data
D SDSUDP                181    196*
D SDSUDO                197    200I 0
D SDSUDS                201    204I 0
D SDSUDL                205    208I 0
D SDSUDC                209    212I 0
D* Correlation Id related to this subscription
D SDCID                 213    236
D* Priority set in publications
```

```
D SDPRI                 237    240I 0
D* Accounting Token set in publications
D SDACC                 241    272
D* Appl Identity Data set in publications
D SDAID                 273    304
D* Message Selector
D SDSSP                 305    320*
D SDSSO                 321    324I 0
D SDSSS                 325    328I 0
D SDSSL                 329    332I 0
D SDSSC                 333    336
D* Subscription level
D SDSL                  337    340  0
D* Resolved Long object name
D SDROP                 341    356*
D SDROO                 357    360I 0
D SDROS                 361    364I 0
D SDROL                 365    368I 0
D SDROC                 369    372I 0
```

IBM i MQSMPO (Nastavení voleb vlastností zprávy) na systému IBM i

Struktura **MQSMPO** umožňuje aplikacím určit volby, které řídí způsob nastavení vlastností zpráv.

Přehled

Účel: Struktura je vstupní parametr volání **MQSETMP** .

Znaková sada a kódování: Data v **MQSMPO** musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1221
- "Počáteční hodnoty" na stránce 1222
- "Deklarace RPG" na stránce 1223

Pole

Struktura MQSMPO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

SPOPT (10místné celé číslo se znaménkem)

Volby umístění: Následující volby se vztahují k relativnímu umístění vlastnosti v porovnání s kurzorem vlastnosti:

SPSETF (Nastavení)
Nastaví hodnotu první vlastnosti, která odpovídá zadanému názvu, nebo pokud neexistuje, přidá novou vlastnost za všechny ostatní vlastnosti s odpovídající hierarchií.

SPSETC (nastavení)
Nastaví hodnotu vlastnosti, na kterou ukazuje kurzor vlastnosti. Vlastnost, na kterou ukazuje kurzor vlastnosti, je ta, která byla naposledy dotazována pomocí volby IPINQF nebo IPINQN.

Kurzor vlastnosti je resetován při opětovném použití manipulátoru zprávy nebo při zadání manipulátoru zprávy v poli *HMSG* struktury MQGMO ve volání MQGET nebo ve struktuře MQPMO ve volání MQPUT.

Pokud je tato volba použita, když kurzor vlastnosti ještě nebyl zaveden, nebo pokud byla vlastnost, na kterou ukazoval kurzor vlastnosti, odstraněna, volání selže s kódem dokončení CCFAIL a kódem příčiny RC2471.

Nastavení SPSETA
Nastaví novou vlastnost za vlastnost, na kterou ukazuje kurzor vlastnosti. Vlastnost, na kterou ukazuje kurzor vlastnosti, je ta, která byla naposledy dotazována pomocí volby IPINQF nebo IPINQO.

Kurzor vlastnosti je resetován při opětovném použití manipulátoru zprávy nebo při zadání manipulátoru zprávy v poli *HMSG* struktury MQGMO ve volání MQGET nebo ve struktuře MQPMO ve volání MQPUT.

Pokud je tato volba použita, když kurzor vlastnosti ještě nebyl zaveden, nebo pokud byla vlastnost, na kterou ukazoval kurzor vlastnosti, odstraněna, volání selže s kódem dokončení CCFAIL a kódem příčiny RC2471.

Pokud nepotřebujete žádnou z popsaných voleb, použijte následující volbu:

SPNONE
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SPSETF.

SPSID (10místné celé číslo se znaménkem)

Toto je identifikátor struktury; hodnota musí být:

SPSIDV

Identifikátor pro nastavení struktury voleb vlastností zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **SPSIDV**.

SPVAKCSI (10místné celé číslo se znaménkem)

Znaková sada hodnoty vlastnosti, která má být nastavena, pokud je hodnota znakový řetězec.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **CSAPL**.

SPVALENC (10místné celé číslo se znaménkem)

Kódování hodnoty vlastnosti, která má být nastavena, pokud je hodnota číselná.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **ENNAT**.

SPVER (celé číslo se znaménkem 10 číslic)

Toto je číslo verze struktury; hodnota musí být:

SPVER1
Version-1 nastavte strukturu voleb vlastností zprávy.

Následující konstanta určuje číslo verze aktuální verze:

SPVERC
Aktuální verze struktury vlastností nastavení zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **SPVER1**.

Počáteční hodnoty

Tabulka 733. Pole v objektu MQSMPO

Název pole	Název konstanty	Hodnota konstanty
SPSID	SPSIDV	'SMPO'
SPVER	SPVER1	1
SPOPT	SPNONE	0
SPVALENC	ENNAT	Závisí na prostředí
SPVALCSI	CSAPL	-3

Deklarace RPG

```
D* MQSMPO Structure
D*
D*
D* Structure identifier
D  SPSID                  1      4    INZ('SMPO')
D*
D* Structure version number
D  SPVER                  5      8I 0 INZ(1)
D*
** Options that control the action of
D* MQSETMP
D  SPOPT                  9     12I 0 INZ(0)
D*
D* Encoding of Value
D  SPVALENC              13     16I 0 INZ(273)
D*
D* Character set identifier of Value
D  SPVALCSI              17     20I 0 INZ(-3)
```

IBM i MQSRO (Volby požadavku na odběr) na systému IBM i

Struktura MQSRO umožňuje aplikaci určit volby, které řídí způsob provedení požadavku na odběr.

Přehled

Účel: Struktura je vstupní/výstupní parametr pro volání MQSUBRQ.

Verze: Aktuální verze MQSRO je SRVER1.

- "Pole" na stránce 1223
- "Počáteční hodnoty" na stránce 1224
- "Deklarace RPG" na stránce 1224

Pole

Struktura MQSRO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

SRNMP (10místné celé číslo se znaménkem)

Jedná se o výstupní pole vrácené aplikaci, které určuje počet publikování odeslaných do fronty odběrů v důsledku tohoto volání. Ačkoli byl tento počet publikací odeslán jako výsledek tohoto volání, není zde žádná záruka, že tento počet zpráv bude k dispozici pro získání aplikace, zejména pokud se jedná o dočasné zprávy.

Pokud bylo téma přihlášeno k odběru a obsahovalo zástupný znak, může existovat více než jedna publikace. Pokud v řetězci tématu nebyly při vytvoření odběru reprezentovaného hodnotou *HSUB* přítomny žádné zástupné znaky, bude v důsledku tohoto volání odesláno nejvýše jedno publikování.

SROPT (10místné celé číslo se znaménkem)

Musí být zadána jedna z následujících voleb. Lze zadat pouze jednu volbu.

Další volby: Následující volba řídí, co se stane, když se správce front uvede do klidového stavu:

SRFIQ

Volání MQSUBRQ se nezdaří, pokud je správce front ve stavu uvedení do klidového stavu.

Výchozí volba: Pokud dříve popsaná volba není povinná, musí být použita následující volba:

SRNONE

Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

SRNONE pomáhá s dokumentací programu. Ačkoli není zamýšleno, aby byla tato volba použita s ostatními, protože její hodnota je nula, nelze toto použití zjistit.

SRSID (4bajtový znakový řetězec)

Toto je identifikátor struktury; hodnota musí být:

SRSIDV
Identifikátor pro strukturu SROPT požadavku na odběr.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SRSIDV.

SRVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury; hodnota musí být:

SRVER1
Version-1 Struktura voleb požadavku na odběr.

Následující konstanta určuje číslo verze aktuální verze:

SRVERC
Aktuální verze struktury voleb požadavku na odběr.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SRVER1.

Počáteční hodnoty

Tabulka 734. Pole v MQSRO

Název pole	Název konstanty	Hodnota konstanty
SRSID	SRSIDV	'SRO¬'
SRVER	SRVER1	1
SROPT	SRNONE	0
SRNMP	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězec nebo mezery označuje nulový řetězec v jazyce C a prázdné znaky v jiných programovacích jazycích.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQSRO Structure
D*
D* Structure identifier
D  SRSID                  1      4
D* Structure version number
D  SRVER                  5      8I 0
D* Options that control the action of MQSUBRQ
D  SROPT                  9     12I 0
D* Number of publications sent
D  SRNMP                 13     16I 0
```

IBM i MQSTS (vykazovací struktura stavu) na systému IBM i

Struktura MQSTS popisuje data ve struktuře stavu vrácená příkazem MQSTAT.

Přehled

Znaková sada a kódování: Znaková data v MQSTS jsou ve znakové sadě lokálního správce front; to je dáno atributem správce front *CodedCharSetId* . Číselná data v MQSTS jsou v nativním kódování počítače; toto je dáno hodnotou *ENNAT*.

Použití: Příkaz MQSTAT se používá k načtení informací o stavu. Tyto informace jsou vráceny ve struktuře MQSTS. Informace o MQSTAT viz "MQSTAT (Načtení informací o stavu) na systému IBM i" na stránce 1351.

- "Pole" na stránce 1225
- "Počáteční hodnoty" na stránce 1228
- "Deklarace RPG" na stránce 1228

Pole

Struktura MQSTS obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

STSCC (10místné celé číslo se znaménkem)

Jedná se o kód dokončení, který je výsledkem první chyby ohlášené ve struktuře MQSTS.

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je CCOK.

STSFC (10místné celé číslo se znaménkem)

Jedná se o počet asynchronních volání vložení, která se nezdařila.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0.

STSOBJN (48bajtový znakový řetězec)

Jedná se o lokální název objektu zahrnutého v prvním selhání.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 48 prázdných znaků.

STSOQMGR (48bajtový znakový řetězec)

Jedná se o název správce front, v němž je definován objekt *STSOBJN* . Název, který je zcela prázdný až do prvního znaku null nebo do konce pole, označuje správce front, ke kterému je aplikace připojena (lokální správce front).

Toto je výstupní pole. Počáteční hodnota tohoto pole je 48 prázdných znaků.

STSOO (10místné celé číslo se znaménkem)

STS00 použitý k otevření objektu, pro který se má hlášení provést. K dispozici pouze ve verzi 2 produktu MQSTS nebo vyšší.

Hodnota parametru STS00 závisí na hodnotě parametru MQSTAT **STYPE** .

STATAPT

Nula.

STATREC

Nula.

STATRER

Hodnota STS00 použitá v době, kdy došlo k selhání. Příčina poruchy je hlášena v polích *STSCC* a *STSRC* ve struktuře MQSTS .

STS00 je výstupní pole. Jeho počáteční hodnota je nula.

STSOS (MQCHARV)

Dlouhý název objektu, pro který došlo k selhání. K dispozici pouze ve verzi 2 produktu MQSTS nebo vyšší.

STS0S je pole MQCHARV s maximální délkou 10240. Popis použití struktury MQCHARV viz MQCHARV .

Interpretace parametru STS0S závisí na hodnotě parametru MQSTAT **STYPE** .

STATAPT

Jedná se o dlouhý název objektu fronty nebo tématu použitého v operaci MQPUT , která se nezdařila.

STATREC

Řetězec s nulovou délkou

STATRER

Jedná se o dlouhý název objektu, který způsobil selhání opětovného připojení.

STS0S je výstupní pole. Jeho počáteční hodnota je řetězec s nulovou délkou.

STSOT (10místné celé číslo se znaménkem)

Typ objektu, který je pojmenován v souboru *ObjectName*. Možné hodnoty jsou:

OTALSQ (přenosové fronty)
Fronta alias.

OTLOCQ
Lokální fronta.

OTMODQ
Modelová fronta.

OTQ (dotazů)
Fronta.

OTREMQ:
Vzdálená fronta.

OTTOP
.

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je OTQ.

STSRC (10místné celé číslo se znaménkem)

Toto je kód příčiny, který je výsledkem první chyby ohlášené ve struktuře MQSTS

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je RCNONE.

STSROBJN (48bajtový znakový řetězec)

Jedná se o název cílové fronty pojmenované v souboru *STSOBJN* poté, co lokální správce front přeloží název. Vrácený název je název fronty, která existuje ve správci front identifikovaném pomocí *STSRQMGR*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou otevřenou pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Pokud je otevřený objekt některý z následujících, *STSROBJN* je nastaven na mezery:

- Téma
- Fronta, ale neotevřená pro procházení, vstup nebo výstup

Toto je výstupní pole. Počáteční hodnota tohoto pole je 48 prázdných znaků.

STSRQMGR (48bajtový znakový řetězec)

Jedná se o název cílového správce front poté, co lokální správce front přeloží název. Vrácený název je název správce front, který vlastní frontu označenou jako *STSROBJN*. *STSRQMGR* může být název lokálního správce front.

Pokud je *STSROBJN* sdílená fronta, kterou vlastní skupina sdílení front, do které lokální správce front patří, *STSRQMGR* je název skupiny sdílení front. Pokud je fronta vlastněna jinou skupinou sdílení front,

může být *STSROBJN* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (povaha vrácené hodnoty je určena definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou otevřenou pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Pokud je otevřený objekt některý z následujících, *STSRQMGR* je nastaven na mezery:

- Téma
- Fronta, ale neotevřená pro procházení, vstup nebo výstup
- Fronta klastru se zadaným OOBNDN (nebo s OOBNDQ v platnosti, když má atribut fronty **DefBind** hodnotu OOBNDN)

Toto je výstupní pole. Počáteční hodnota tohoto pole je 48 prázdných znaků.

STSSC (10místné celé číslo se znaménkem)

Jedná se o počet úspěšných asynchronních volání vložení.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0.

STSSID (4bajtový znakový řetězec)

Jedná se o identifikátor struktury. Hodnota musí být:

STSSID
Identifikátor pro strukturu vykazování stavu.

Počáteční hodnota tohoto pole je STSSID.

STSSO (10místné celé číslo se znaménkem)

Soubor STSSO použitý k otevření selhávajícího odběru. K dispozici pouze ve verzi 2 produktu MQSTS nebo vyšší.

Interpretace parametru STSSO závisí na hodnotě parametru MQSTAT **STYPE** .

STATAPT

Nula.

STATREC

Nula.

STATRER

Hodnota STSSO použitá v době, kdy došlo k selhání. Příčina poruchy je hlášena v polích *STSCC* a *STSRC* ve struktuře MQSTS . Pokud selhání nesouvisí s přihlášením k odběru tématu, vrácená hodnota je nula.

STSSO je výstupní pole. Jeho počáteční hodnota je nula.

STSSUN (MQCHARV)

Název selhávajícího odběru. K dispozici pouze ve verzi 2 produktu MQSTS nebo vyšší.

STSSUN je pole MQCHARV s délkou maxiumum 10240. Popis použití struktury MQCHARV viz MQCHARV .

Interpretace parametru STSSUN závisí na hodnotě parametru MQSTAT **STYPE** .

STATAPT

Řetězec s nulovou délkou.

STATREC

Řetězec s nulovou délkou.

STATRER

Název odběru, který způsobil selhání opětovného připojení. Pokud není k dispozici žádný název odběru nebo pokud selhání nesouvisí s odběrem, jedná se o řetězec s nulovou délkou.

STSSUN je výstupní pole. Jeho počáteční hodnota je řetězec s nulovou délkou.

STSVER (10místné celé číslo se znaménkem)

Toto je číslo verze struktury. Hodnota musí být:

STSVR1
Číslo verze pro strukturu hlášení stavu.

Následující konstanta určuje číslo verze aktuální verze:

STSVRC
Aktuální verze struktury hlášení stavu.

Počáteční hodnota tohoto pole je STSVR1.

STSWC (10místné celé číslo se znaménkem)

Jedná se o počet asynchronních volání vložení, která byla dokončena s varováním.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0.

Počáteční hodnoty

Tabulka 735. Pole v MQSTS

Název pole	Název konstanty	Hodnota konstanty
STSSID	STSID	
STSVER	STSVRC	STSVR1
STSCC	CCOK	0
STSRC	RCNONE	0
STSSC	Není	0
STSWC	Není	0
STSFC	Není	0
STSOT	Není	0
STSOBJN	Není	Mezery
STSOQMGR	Není	Mezery
STSROBJN	Není	Mezery
STSRQMGR	Není	Mezery
STSOS	Názvy a hodnoty definované pro MQCHARV	
STSSUN	Názvy a hodnoty definované pro MQCHARV	
STSOO	Není	0
STSSO	Není	0

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQSTS Structure
D*
```

```
D* Structure identifier
D  STSSID                 1      4
D* Structure version number
D  STSVER                 5      8I 0
D* Completion code
D  STSCC                  9     12I 0
D* Reason code
D  STSRC                 13     16I 0
D* Success count
D  STSSC                 17     20I 0
D* Warning count
D  STSWC                 21     24I 0
D* Failure count
D  STSFC                 25     28I 0
D* Object type
D  STSOT                 29     32I 0
D* Object name
D  STSOBJN               33     80
D* Object queue manager
D  STSOQMGR              81     128
D* Resolved object name
D  STSROBJN             129     176
D* Resolved object queue manager name
D  STSRQMGR             177     224
D* Ver:1 **
D* Failing object long name
D* Address of variable length string
D  STSOSCHRP            225    240*
D* Offset of variable length string
D  STSOSCHRO            241    244I 0
D* Size of buffer
D  STSOSVSBS            245    248I 0
D* Length of variable length string
D  STSOSCHRL            249    252I 0
D* CCSID of variable length string
D  STSOSCHRC            253    256I 0
D* Failing subscription name
D* Address of variable length string
D  STSSUNCHRP           257    272*
D* Offset of variable length string
D  STSSUNCHRO           273    276I 0
D* Size of buffer
D  STSSUNVSBS           277    280I 0
D* Length of variable length string
D  STSSUNCHRL           281    284I 0
D* CCSID of variable length string
D  STSSUNCHRC           285    288I 0
D* Failing open options
D  STSOO                289    292I 0
D* Failing subscription options
D  STSSO                293    296I 0
D* Ver:2 **
```

MQTM-zpráva spouštěče

Struktura MQTM popisuje data ve zprávě spouštěče, která je odeslána správcem front do aplikace monitoru spouštěčů při výskytu události spouštěče pro frontu.

Přehled

Účel: Tato struktura je součástí rozhraní IBM MQ Trigger Monitor Interface (TMI), které je jedním z rozhraní rámce IBM MQ .

Název formátu: FMTM.

Znaková sada a kódování: Znaková data v MQTM jsou ve znakové sadě správce front, který generuje MQTM. Číselná data v MQTM jsou v kódování počítače správce front, který generuje MQTM.

Znaková sada a kódování MQTM jsou dány poli *MDCSI* a *MDENC* v:

- MQMD (pokud je struktura MQTM na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQTM (všechny ostatní případy).

Použití: Aplikace monitoru spouštěčů může potřebovat předat některé nebo všechny informace ve zprávě spouštěče aplikaci, která je spuštěna aplikací monitoru spouštěčů. Informace, které může spuštěná

aplikace potřebovat, zahrnují *TMQN*, *TMTD*a *TMUD*. Aplikace monitoru spouštěčů může předat strukturu MQTM přímo spuštěné aplikaci, nebo místo toho předat strukturu MQTMC2 v závislosti na tom, co je povoleno prostředím a vhodné pro spuštěnou aplikaci. Informace o MQTMC2viz "MQTMC2 (dvouznakový formát zprávy spouštěče) v systému IBM i" na stránce 1233.

- V systému IBM iaplikace monitoru spouštěčů dodávané s produktem IBM MQ předává spuštěnou aplikaci strukturu MQTMC2 .

Informace o spouštěčích naleznete v tématu Předpoklady pro spouštění.

- "MQMD pro zprávu spouštěče" na stránce 1230
- "Pole" na stránce 1231
- "Počáteční hodnoty" na stránce 1233
- "Deklarace RPG" na stránce 1233

MQMD pro zprávu spouštěče

Tabulka 736. Nastavení pro pole v deskriptoru MQMD zprávy spouštěče generované správcem front

Pole v deskriptoru MQMD	Použitá hodnota
MDSID	MDSIDV
MDVER	MDVER1
MDREP	RONONE
MDMT	MTDGRM
MDEXP	EIULIM
MDFB	FBNONE
MDENC	ENNAT
MDCSI	Atribut **CodedCharSetId** správce front
MDFMT	FMTM
MDPRI	Atribut **DefPriority** inicializační fronty
MDPER	PENPER
MDMID	Jedinečná hodnota
MDCID	CINONE
MDBOC	0
MDRQ	Mezery
MDRM	Název správce front
MDUID	Mezery
MDACC	ACNONE
MDAID	Mezery
MDPAT	ATQM nebo podle potřeby pro agenta kanálu zpráv
MDPAN	Prvních 28 bajtů názvu správce front
MDPD	Datum odeslání zprávy spouštěče
MDPT	Čas odeslání zprávy spouštěče
MDAOD	Mezery

Aplikaci, která generuje zprávu spouštěče, se doporučuje nastavit podobné hodnoty, s výjimkou následujících:

- Pole *MDPRI* lze nastavit na hodnotu PRQDEF (správce front jej při vložení zprávy změní na výchozí prioritu inicializační fronty).
- Pole *MDRM* lze nastavit na prázdné hodnoty (správce front jej změní na název lokálního správce front při vložení zprávy).
- Pole kontextu by měla být nastavena podle potřeby pro aplikaci.

Pole

Struktura MQTM obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

TMAI (256bajtový znakový řetězec)

Identifikátor aplikace.

Jedná se o znakový řetězec, který identifikuje aplikaci, která má být spuštěna, a je používán aplikací monitoru spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole hodnotou atributu **ApplId** objektu procesu identifikovaného polem *TMPN* . Podrobnosti o tomto atributu naleznete v části “Atributy pro definice procesů v systému IBM i” na stránce 1389 . Obsah těchto dat nemá pro správce front žádný význam.

Význam parametru *TMAI* je určen aplikací pro monitorování spouštěčů. Monitor spouštěčů poskytovaný produktem IBM MQ vyžaduje, aby *TMAI* byl název spustitelného programu.

Délka tohoto pole je dána hodnotou LNPROA. Počáteční hodnota tohoto pole je 256 prázdných znaků.

TMAT (10místné celé číslo se znaménkem)

Typ aplikace.

To identifikuje povahu programu, který se má spustit, a je používán aplikací pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole hodnotou atributu **ApplType** objektu procesu identifikovaného polem *TMPN* . Podrobnosti o tomto atributu naleznete v části “Atributy pro definice procesů v systému IBM i” na stránce 1389 . Obsah těchto dat nemá pro správce front žádný význam.

TMAT může mít jednu z následujících standardních hodnot. Lze také použít uživatelsky definované typy, ale měly by být omezeny na hodnoty v rozsahu ATUFST až ATULST:

rovnoCICS
CICS transakce.

ATVSE
CICS/VSE transakce.

AT400
IBM i .

ATUFST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

ATULST
Nejvyšší hodnota pro typ aplikace definovaný uživatelem.

Počáteční hodnota tohoto pole je 0.

TMED (128bajtový znakový řetězec)

Data prostředí.

Jedná se o znakový řetězec, který obsahuje informace související s prostředím aplikace, která má být spuštěna, a je používán aplikací monitoru spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole hodnotou atributu **EnvData** objektu procesu identifikovaného polem *TMPN* .

Podrobnosti o tomto atributu naleznete v části “Atributy pro definice procesů v systému IBM i” na stránce 1389 . Obsah těchto dat nemá pro správce front žádný význam.

Délka tohoto pole je dána hodnotou LNPROE. Počáteční hodnota tohoto pole je 128 prázdných znaků.

TMPN (48bajtový znakový řetězec)

Název objektu procesu.

Jedná se o název objektu procesu správce front určeného pro spuštěnou frontu a může být použit aplikací monitoru spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **ProcessName** fronty určené polem *TMQN* ; podrobnosti o tomto atributu naleznete v části “Atributy pro fronty” na stránce 1360 .

Názvy, které jsou kratší než definovaná délka pole, jsou vždy zprava vyplněny mezerami; nejsou předčasně ukončeny znakem null.

Délka tohoto pole je dána hodnotou LNPRON. Počáteční hodnota tohoto pole je 48 prázdných znaků.

TMQN (48bajtový znakový řetězec)

Název spuštěné fronty.

Jedná se o název fronty, pro kterou došlo k události spouštěče, a je používán aplikací spuštěnou aplikací monitoru spouštěčů. Správce front inicializuje toto pole s hodnotou atributu **QName** spuštěné fronty. Podrobnosti o tomto atributu naleznete v části “Atributy pro fronty” na stránce 1360 .

Názvy, které jsou kratší než definovaná délka pole, jsou vpravo vyplněny mezerami; nejsou předčasně ukončeny znakem null.

Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

TMSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

TMSIDV
Identifikátor pro strukturu zprávy spouštěče.

Počáteční hodnota tohoto pole je TMSIDV.

TMTD (64bitový znakový řetězec)

Data spouątěče.

Jedná se o data ve volném formátu pro použití aplikací monitoru spouštěčů, která obdrží zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **TriggerData** fronty určené polem *TMQN* ; podrobnosti o tomto atributu naleznete v části “Atributy pro fronty” na stránce 1360 . Obsah těchto dat nemá pro správce front žádný význam.

Délka tohoto pole je dána hodnotou LNTRGD. Počáteční hodnota tohoto pole je 64 prázdných znaků.

TMUD (128bajtový znakový řetězec)

Uživatelská data.

Jedná se o znakový řetězec, který obsahuje informace o uživateli související s aplikací, která má být spuštěna, a je používán aplikací monitoru spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole hodnotou atributu **UserData** objektu procesu identifikovaného polem *TMPN* . Podrobnosti o tomto atributu naleznete v části “Atributy pro definice procesů v systému IBM i” na stránce 1389 . Obsah těchto dat nemá pro správce front žádný význam.

Délka tohoto pole je dána hodnotou LNPROU. Počáteční hodnota tohoto pole je 128 prázdných znaků.

TMVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

TMVER1
Číslo verze pro strukturu zprávy spouštěče.

Následující konstanta určuje číslo verze aktuální verze:

TMVERC
Aktuální verze struktury zprávy spouštěče.

Počáteční hodnota tohoto pole je TMVER1.

Počáteční hodnoty

Tabulka 737. Pole v MQTM

Název pole	Název konstanty	Hodnota konstanty
TMSID	TMSIDV	'TM¬¬'
TMVER	TMVER1	1
TMQN	Není	Mezery
TMPN	Není	Mezery
TMTD	Není	Mezery
TMAT	Není	0
TMAI	Není	Mezery
TMED	Není	Mezery
TMUD	Není	Mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQTM Structure
D*
D* Structure identifier
D  TMSID                  1      4    INZ('TM  ')
D* Structure version number
D  TMVER                  5      8I 0 INZ(1)
D* Name of triggered queue
D  TMQN                   9     56    INZ
D* Name of process object
D  TMPN                  57    104    INZ
D* Trigger data
D  TMTD                 105    168    INZ
D* Application type
D  TMAT                 169    172I 0 INZ(0)
D* Application identifier
D  TMAI                 173    428    INZ
D* Environment data
D  TMED                 429    556    INZ
D* User data
D  TMUD                 557    684    INZ
```

IBM i MQTMC2 (dvouznakový formát zprávy spouštěče) v systému IBM i

Když aplikace monitoru spouštěčů načte zprávu spouštěče (MQTM) z inicializační fronty, může být nutné, aby monitor spouštěčů předal některé nebo všechny informace ve zprávě spouštěče aplikaci, která je spuštěna monitorem spouštěčů.

Přehled

Účel: Informace, které může spuštěná aplikace potřebovat, zahrnují *TC2QN*, *TC2TD*a *TC2UD*. Aplikace monitoru spouštěčů může předat strukturu MQTM přímo spuštěné aplikaci, nebo místo toho předat strukturu MQTMC2 v závislosti na tom, co je povoleno prostředím a vhodné pro spuštěnou aplikaci.

Tato struktura je součástí rozhraní IBM MQ Trigger Monitor Interface (TMI), které je jedním z rozhraní rámce IBM MQ .

Znaková sada a kódování: Znaková data v produktu MQTMC2 se nacházejí ve znakové sadě lokálního správce front. Toto je dáno atributem správce front **CodedCharSetId** .

Použití: Struktura MQTMC2 je podobná formátu struktury MQTM. Rozdíl je v tom, že neznaková pole v produktu MQTM jsou v produktu MQTMC2 změněna na znaková pole stejné délky a na konec struktury je přidán název správce front.

- V systému IBM iaplikace monitoru spouštěčů dodávané s produktem IBM MQ předává spuštěnou aplikaci strukturu MQTMC2 .
- “Pole” na stránce 1234
- “Počáteční hodnoty” na stránce 1235
- “Deklarace RPG” na stránce 1235

Pole

Struktura MQTMC2 obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

TC2AI (256bajtový znakový řetězec)

Identifikátor aplikace.

Viz pole *TMAI* ve struktuře MQTM.

TC2AT (4bajtový znakový řetězec)

Typ aplikace.

Toto pole vždy obsahuje mezery, bez ohledu na hodnotu v poli *TMAT* ve struktuře MQTM původní zprávy spouštěče.

TC2ED (128bajtový znakový řetězec)

Data prostředí.

Viz pole *TMED* ve struktuře MQTM.

TC2PN (48bajtový znakový řetězec)

Název objektu procesu.

Viz pole *TMPN* ve struktuře MQTM.

TC2QMN (48bajtový znakový řetězec)

Název správce front.

Jedná se o název správce front, ve kterém došlo k události spouštěče.

TC2QN (48bajtový znakový řetězec)

Název spuštěné fronty.

Viz pole *TMQN* ve struktuře MQTM.

TC2SID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

TCSIDV
Identifikátor pro strukturu zprávy spouštěče (znakový formát).

TC2TD (64bitový znakový řetězec)

Data spouątěče.

Viz pole *TMTD* ve struktuře MQTM.

TC2UD (128bajtový znakový řetězec)

Uživatelská data.

Viz pole *TMUD* ve struktuře MQTM.

TC2VER (4bajtový znakový řetězec)

Číslo verze struktury.

Hodnota musí být:

TCVER2
Struktura zprávy spouštěče verze 2 (znakový formát).

Následující konstanta určuje číslo verze aktuální verze:

TCVERC
Aktuální verze struktury zprávy spouštěče (znakový formát).

Počáteční hodnoty

Tabulka 738. Pole v tabulce MQTMC2

Název pole	Název konstanty	Hodnota konstanty
TC2SID	TCSIDV	'TMC¬'
TC2VER	TCVER2	'¬¬¬2'
TC2QN	Není	Mezery
TC2PN	Není	Mezery
TC2TD	Není	Mezery
TC2AT	Není	Mezery
TC2AI	Není	Mezery
TC2ED	Není	Mezery
TC2UD	Není	Mezery
TC2QMN	Není	Mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQTMC2 Structure
D*
D* Structure identifier
D  TC2SID                1      4
D* Structure version number
D  TC2VER                5      8
D* Name of triggered queue
D  TC2QN                 9     56
```

```
     D* Name of process object
     D  TC2PN                 57    104
     D* Trigger data
     D  TC2TD                105    168
     D* Application type
     D  TC2AT                169    172
     D* Application identifier
     D  TC2AI                173    428
     D* Environment data
     D  TC2ED                429    556
     D* User data
     D  TC2UD                557    684
     D* Queue manager name
     D  TC2QMN               685    732
```

IBM i MQWIH (záhlaví informací o práci) na IBM i

Struktura MQWIH popisuje informace, které musí být přítomny na začátku zprávy, která má být zpracována správcem pracovní zátěže z/OS .

Přehled

Název formátu: FMWIH.

Znaková sada a kódování: Pole ve struktuře MQWIH jsou ve znakové sadě a kódování dané poli *MDCSI* a *MDENC* ve struktuře záhlaví, která předchází MQWIH, nebo těmito poli ve struktuře MQMD, pokud je MQWIH na začátku dat zprávy aplikace.

Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech front.

Použití: Pokud má správce pracovní zátěže z/OS zprávu zpracovat, musí zpráva začínat strukturou MQWIH.

- "Pole" na stránce 1236
- "Počáteční hodnoty" na stránce 1238
- "Deklarace RPG" na stránce 1238

Pole

Struktura MQWIH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

WICSI (10místné celé číslo se znaménkem)

Identifikátor znakové sady dat, která následují za MQWIH.

Tato volba určuje identifikátor znakové sady dat, která se řídí strukturou MQWIH; nevztahuje se na znaková data v samotné struktuře MQWIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Lze použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následujících* je tato struktura ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Za předpokladu, že nedojde k žádné chybě, není hodnota CSINHT vrácena voláním MQGET.

CSINHT nelze použít, pokud je hodnota pole *MDPAT* v MQMD ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

WIENC (10místné celé číslo se znaménkem)

Číselné kódování dat, která následují za MQWIH.

Tato volba určuje číselné kódování dat, která následují za strukturou MQWIH; nevztahuje se na číselná data v samotné struktuře MQWIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

WIFLG (10místné celé číslo se znaménkem)

Příznaky

Hodnota musí být:

WINONE
Žádné příznaky.

Počáteční hodnota tohoto pole je WINONE.

WIFMT (8bajtový znakový řetězec)

Název formátu dat, která následují za MQWIH.

Tato volba určuje název formátu dat, která se řídí strukturou MQWIH.

Při volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pro pole *MDFMT* v MQMD.

Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

WILEN (10místné celé číslo se znaménkem)

Délka struktury MQWIH.

Hodnota musí být:

WILEN1
Délka struktury záhlaví pracovních informací version-1 .

Následující konstanta určuje délku aktuální verze:

WILENC
Délka aktuální verze struktury záhlaví informací o práci.

Počáteční hodnota tohoto pole je WILEN1.

WIRSV (32bajtový znakový řetězec)

Vyhrazeno.

Toto je vyhrazené pole; musí být prázdné.

WISID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

WISIDV
Identifikátor struktury záhlaví informací o práci.

Počáteční hodnota tohoto pole je WISIDV.

WISNM (32bajtový znakový řetězec)

Název služby.

Jedná se o název služby, která má zpracovat zprávu.

Délka tohoto pole je dána hodnotou LNSVNM. Počáteční hodnota tohoto pole je 32 prázdných znaků.

WISST (8bajtový znakový řetězec)

Název kroku služby.

Jedná se o název kroku *WISNM* , ke kterému se zpráva vztahuje.

Délka tohoto pole je dána LNSVST. Počáteční hodnota tohoto pole je 8 prázdných znaků.

WITOK (16bajtový bitový řetězec)

Token zprávy.

Jedná se o token zprávy, který jednoznačně identifikuje zprávu.

Pro volání MQPUT a MQPUT1 je toto pole ignorováno. Délka tohoto pole je dána hodnotou LNMTOK. Počáteční hodnota tohoto pole je MTKNON.

WIVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

WIVER1
Struktura záhlaví pracovních informací Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

WIVERC
Aktuální verze struktury záhlaví informací o práci.

Počáteční hodnota tohoto pole je WIVER1.

Počáteční hodnoty

Tabulka 739. Pole v MQWIH

Název pole	Název konstanty	Hodnota konstanty
WISID	WISIDV	'WIH¬'
WIVER	WIVER1	1
WILEN	WILEN1	120
WIENC	Není	0
WICSI	CSUNDF	0
WIFMT	FMNONE	Mezery
WIFLG	WINONE	0
WISNM	Není	Mezery
WISST	Není	Mezery
WITOK	MTKNON	Hodnoty null
WIRSV	Není	Mezery

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQWIH Structure
D*
D* Structure identifier
D  WISID                  1      4    INZ('WIH ')
D* Structure version number
D  WIVER                  5      8I 0 INZ(1)
D* Length of MQWIH structure
```

```
     D  WILEN                  9     12I 0 INZ(120)
     D* Numeric encoding of data that followsMQWIH
     D  WIENC                 13     16I 0 INZ(0)
     D* Character-set identifier of data thatfollows MQWIH
     D  WICSI                 17     20I 0 INZ(0)
     D* Format name of data that followsMQWIH
     D  WIFMT                 21     28    INZ('        ')
     D* Flags
     D  WIFLG                 29     32I 0 INZ(0)
     D* Service name
     D  WISNM                 33     64    INZ
     D* Service step name
     D  WISST                 65     72    INZ
     D* Message token
     D  WITOK                 73     88    INZ(X'00000000000000-
     D                                     0000000000000000')
     D* Reserved
     D  WIRSV                 89    120    INZ
```

IBM i MQXQH (záhlaví přenosové fronty) na IBM i

Struktura MQXQH popisuje informace, které mají předponu k datům zpráv aplikace pro zprávy, když jsou v přenosových frontách.

Přehled

Účel: přenosová fronta je speciální typ lokální fronty, která dočasně uchovává zprávy určené pro vzdálené fronty (tj. určené pro fronty, které nepatří lokálnímu správci front). Přenosová fronta je označena atributem fronty **Usage** , který má hodnotu USTRAN.

Název formátu: FMXQH.

Znaková sada a kódování: Data v MQXQH musí být ve znakové sadě určené atributem správce front **CodedCharSetId** a kódování lokálního správce front daného ENNAT pro programovací jazyk C.

Znaková sada a kódování MQXQH musí být nastaveny do polí *MDCSI* a *MDENC* v:

- samostatné MQMD (pokud je struktura MQXQH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQXQH (všechny ostatní případy).

Použití: Zpráva, která je v přenosové frontě, má *dva* deskriptory zpráv:

- Jeden deskriptor zprávy je uložen odděleně od dat zprávy. Tento deskriptor se nazývá *samostatný deskriptor zprávy*a je generován správcem front při umístění zprávy do přenosové fronty. Některá pole v odděleném deskriptoru zpráv jsou zkopírována z deskriptoru zpráv poskytnutého aplikací ve volání MQPUT nebo MQPUT1 .

  Oddělený deskriptor zprávy je ten, který je vrácen aplikaci v parametru **MSGDSC** volání MQGET při odebrání zprávy z přenosové fronty.
- Druhý deskriptor zprávy je uložen ve struktuře MQXQH jako součást dat zprávy; tento deskriptor se nazývá *vložený deskriptor zprávy*a je kopií deskriptoru zprávy, který byl poskytnut aplikací ve volání MQPUT nebo MQPUT1 (s menšími variacemi).

  Vložený deskriptor zprávy je vždy MQMD version-1 . Pokud má zpráva vložená aplikací jiné než výchozí hodnoty pro jedno nebo více polí version-2 v deskriptoru MQMD, následuje struktura MQMDE za MQXQH a je následována daty zprávy aplikace (pokud existují). MQMDE je buď:

  – Generováno správcem front (pokud aplikace k vložení zprávy používá MQMD version-2), nebo
  – Již se nachází na začátku dat zprávy aplikace (pokud aplikace používá k vložení zprávy MQMD version-1).

  Vložený deskriptor zprávy je ten, který je vrácen aplikaci v parametru **MSGDSC** volání MQGET při odebrání zprávy z konečné cílové fronty.

- “Pole v odděleném deskriptoru zprávy” na stránce 1240
- “Pole ve vloženém deskriptoru zprávy” na stránce 1241

- “Vkládání zpráv do vzdálených front” na stránce 1241
- “Vkládání zpráv přímo do přenosových front” na stránce 1242
- “Získávání zpráv z přenosových front” na stránce 1242
- “Pole” na stránce 1242
- “Počáteční hodnoty” na stránce 1243
- “Deklarace RPG” na stránce 1243

Pole v odděleném deskriptoru zprávy

Pole v odděleném deskriptoru zpráv jsou nastavena správcem front, jak je uvedeno v následujícím seznamu. Pokud správce front nepodporuje rozhraní MQMD version-2 , bude použit modul MQMD version-1 bez ztráty funkce.

Tabulka 740. Pole v odděleném deskriptoru zprávy a použité hodnoty

Pole v odděleném deskriptoru MQMD	Použitá hodnota
MDSID	MDSIDV
MDVER	MDVER2
MDREP	Zkopírováno z vloženého deskriptoru zprávy, ale s bity identifikovanými hodnotou ROAUXM nastavenou na nulu. (To zabraňuje generování zprávy sestavy COA nebo COD, když je zpráva umístěna do přenosové fronty nebo odebrána z přenosové fronty.)
MDMT	Zkopírováno z vloženého deskriptoru zprávy.
MDEXP	Zkopírováno z vloženého deskriptoru zprávy.
MDFB	Zkopírováno z vloženého deskriptoru zprávy.
MDENC	ENNAT
MDCSI	Atribut **CodedCharSetId** správce front.
MDFMT	FMXQH
MDPRI	Zkopírováno z vloženého deskriptoru zprávy.
MDPER	Zkopírováno z vloženého deskriptoru zprávy.
MDMID	Správce front vygeneruje novou hodnotu. Tento identifikátor zprávy se liší od identifikátoru *MDMID* , který mohl být vygenerován správcem front pro vložený deskriptor zprávy (viz předchozí popis).
MDCID	*MDMID* z vloženého deskriptoru zprávy.
MDBOC	0
MDRQ	Zkopírováno z vloženého deskriptoru zprávy.
MDRM	Zkopírováno z vloženého deskriptoru zprávy.
MDUID	Zkopírováno z vloženého deskriptoru zprávy.
MDACC	Zkopírováno z vloženého deskriptoru zprávy.
MDAID	Zkopírováno z vloženého deskriptoru zprávy.
MDPAT	ATQM-řízení jakosti
MDPAN	Prvních 28 bajtů názvu správce front.
MDPD	Datum, kdy byla zpráva vložena do přenosové fronty.
MDPT	Čas, kdy byla zpráva vložena do přenosové fronty.

Tabulka 740. Pole v odděleném deskriptoru zprávy a použité hodnoty (pokračování)

Pole v odděleném deskriptoru MQMD	Použitá hodnota
MDAOD	Mezery
MDGID	GINONE
MDSEQ	1
MDOFF	0
MDMFL	MFNONE
MDOLN	OLUNDF

Pole ve vloženém deskriptoru zprávy

Pole ve vloženém deskriptoru zprávy mají stejné hodnoty jako pole v parametru **MSGDSC** volání MQPUT nebo MQPUT1 , s výjimkou následujících:

- Pole *MDVER* má vždy hodnotu MDVER1.
- Pokud má pole *MDPRI* hodnotu PRQDEF, nahradí se hodnotou atributu **DefPriority** fronty.
- Pokud má pole *MDPER* hodnotu PEQDEF, nahradí se hodnotou atributu **DefPersistence** fronty.
- Má-li pole *MDMID* hodnotu MINONE nebo byla-li zadána volba PMNMID nebo je-li zpráva zprávou distribučního seznamu, je hodnota *MDMID* nahrazena novým identifikátorem zprávy generovaným správcem front.

  Když je zpráva distribučního seznamu rozdělena na menší zprávy distribučního seznamu umístěné v různých přenosových frontách, pole *MDMID* v každém z nových vložených deskriptorů zpráv je stejné jako v původní zprávě distribučního seznamu.
- Pokud byla zadána volba PMNCID, je parametr *MDCID* nahrazen novým identifikátorem korelace generovaným správcem front.
- Kontextová pole jsou nastavena podle voleb PM* uvedených v parametru **PMO** ; kontextová pole jsou:
  - *MDACC*
  - *MDAID*
  - *MDAOD*
  - *MDPAN*
  - *MDPAT*
  - *MDPD*
  - *MDPT*
  - *MDUID*
- Pole version-2 (pokud byla přítomna) jsou odebrána z MQMD a přesunuta do struktury MQMDE, pokud jedno nebo více polí version-2 má jinou než výchozí hodnotu.

Vkládání zpráv do vzdálených front

: Když aplikace vloží zprávu do vzdálené fronty (buď zadáním názvu vzdálené fronty přímo, nebo pomocí lokální definice vzdálené fronty), lokální správce front:

- Vytvoří strukturu MQXQH obsahující vložený deskriptor zprávy.
- Připojí MQMDE, je-li potřeba a není-li již přítomen
- Připojí data zprávy aplikace
- Umístí zprávu do odpovídající přenosové fronty.

Vkládání zpráv přímo do přenosových front

Je také možné, aby aplikace vložila zprávu přímo do přenosové fronty. V tomto případě musí aplikace před data zprávy aplikace přidat strukturu MQXQH a inicializovat pole s odpovídajícími hodnotami. Kromě toho pole *MDFMT* v parametru **MSGDSC** volání MQPUT nebo MQPUT1 musí mít hodnotu FMXQH.

Znaková data ve struktuře MQXQH vytvořené aplikací musí být ve znakové sadě lokálního správce front (definované atributem správce front **CodedCharSetId**) a celočíselná data musí být v nativním kódování počítače. Kromě toho musí být znaková data ve struktuře MQXQH vyplněna mezerami na definovanou délku pole; data nesmí být předčasně ukončena pomocí znaku null, protože správce front nepřevede hodnotu null a následné znaky na mezery ve struktuře MQXQH.

Všimněte si však, že správce front nekontroluje, zda je přítomna struktura MQXQH nebo zda byly pro pole zadány platné hodnoty.

Získávání zpráv z přenosových front

Aplikace, které získají zprávy z přenosové fronty, musí informace ve struktuře MQXQH zpracovat odpovídajícím způsobem. Přítomnost struktury MQXQH na začátku dat zprávy aplikace je označena hodnotou FMXQH vrácenou v poli *MDFMT* v parametru **MSGDSC** volání MQGET. Hodnoty vrácené v polích *MDCSI* a *MDENC* v parametru **MSGDSC** označují znakovou sadu a kódování znakových a celočíselných dat ve struktuře MQXQH. Znaková sada a kódování dat zprávy aplikace jsou definovány poli *MDCSI* a *MDENC* ve vloženém deskriptoru zprávy.

Pole

Struktura MQXQH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

XQMD (MQMD1)

Původní deskriptor zprávy.

Jedná se o vložený deskriptor zprávy a jedná se o blízkou kopii deskriptoru zprávy MQMD, který byl zadán jako parametr **MSGDSC** ve volání MQPUT nebo MQPUT1 při původním vložení zprávy do vzdálené fronty.

Poznámka: Jedná se o databázi MQMD version-1 .

Počáteční hodnoty polí v této struktuře jsou stejné jako hodnoty ve struktuře MQMD.

XQRQ (48bajtový znakový řetězec)

Název cílové fronty.

Jedná se o název fronty zpráv, která je zdánlivě eventuálním cílem zprávy (to se může ukázat jako neskutečné eventuální místo určení, pokud je například tato fronta definována v *XQRQM* jako lokální definice jiné vzdálené fronty).

Pokud se jedná o zprávu rozdělovníku (tj. pole *MDFMT* ve vloženém deskriptoru zprávy je FMDH), *XQRQ* je prázdné.

Délka tohoto pole je dána hodnotou LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

XQRQM (48bajtový znakový řetězec)

Název cílového správce front.

Jedná se o název správce front nebo skupiny sdílení front, která vlastní frontu, jež je zdánlivě eventuálním cílem zprávy.

Pokud se jedná o zprávu rozdělovníku, *XQRQM* je prázdné.

Délka tohoto pole je dána hodnotou LNQMN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

XQSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

XQSIDV
Identifikátor pro strukturu záhlaví přenosové fronty.

Počáteční hodnota tohoto pole je XQSIDV.

XQVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

XQVER1
Číslo verze pro strukturu záhlaví přenosové fronty.

Následující konstanta určuje číslo verze aktuální verze:

XQVERC-bezpečnostní
Aktuální verze struktury záhlaví přenosové fronty.

Počáteční hodnota tohoto pole je XQVER1.

Počáteční hodnoty

Tabulka 741. Pole v MQXQH

Název pole	Název konstanty	Hodnota konstanty
XQSID	XQSIDV	'XQH¬'
XQVER	XQVER1	1
XQRQ	Není	Mezery
XQRQM	Není	Mezery
XQMD	Stejné názvy a hodnoty jako MQMD; viz Tabulka 709 na stránce 1146	-

Notes:

1. Symbol ¬ představuje jeden prázdný znak.

Deklarace RPG

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D*
D* MQXQH Structure
D*
D* Structure identifier
D  XQSID                  1      4    INZ('XQH ')
D* Structure version number
D  XQVER                  5      8I 0 INZ(1)
D* Name of destination queue
D  XQRQ                   9     56    INZ
D* Name of destination queue manager
D  XQRQM                 57    104    INZ
D* Original message descriptor
D  XQ1SID               105    108    INZ('MD  ')
D  XQ1VER               109    112I 0 INZ(1)
D  XQ1REP               113    116I 0 INZ(0)
D  XQ1MT                117    120I 0 INZ(8)
D  XQ1EXP               121    124I 0 INZ(-1)
D  XQ1FB                125    128I 0 INZ(0)
D  XQ1ENC               129    132I 0 INZ(273)
D  XQ1CSI               133    136I 0 INZ(0)
D  XQ1FMT               137    144    INZ('        ')
D  XQ1PRI               145    148I 0 INZ(-1)
D  XQ1PER               149    152I 0 INZ(2)
D  XQ1MID               153    176    INZ(X'00000000000000-
```

```
     D                                      00000000000000000000-
     D                                      000000000000')
     D  XQ1CID               177    200     INZ(X'00000000000000-
     D                                      00000000000000000000-
     D                                      000000000000')
     D  XQ1BOC               201    204I 0  INZ(0)
     D  XQ1RQ                205    252     INZ
     D  XQ1RM                253    300     INZ
     D  XQ1UID               301    312     INZ
     D  XQ1ACC               313    344     INZ(X'00000000000000-
     D                                      00000000000000000000-
     D                                      00000000000000000000-
     D                                      000000')
     D  XQ1AID               345    376     INZ
     D  XQ1PAT               377    380I 0  INZ(0)
     D  XQ1PAN               381    408     INZ
     D  XQ1PD                409    416     INZ
     D  XQ1PT                417    424     INZ
     D  XQ1AOD               425    428     INZ
```

IBM i Volání funkcí na systému IBM i

Pomocí těchto informací se seznámíte s voláními funkcí, která jsou k dispozici v programování systému IBM i .

Konvence používané v popisech volání na webu IBM i

Pro každé volání tato kolekce témat poskytuje popis parametrů a použití volání. Následuje typické vyvolání volání a typické deklarace jeho parametrů v programovacím jazyku RPG.

Důležité: Při kódování volání rozhraní API IBM MQ musíte zajistit, aby byly poskytnuty všechny příslušné parametry (jak je popsáno v následujících sekcích). Pokud tak neučiníme, může dojít k nepředvídatelným výsledkům.

Popis každého volání obsahuje následující sekce:

Název volání
Název volání následovaný stručným popisem účelu volání.

Parametry
Pro každý parametr je název následován jeho datovým typem v závorkách () a jeho směr, například:

CMPCOD (9místné desetinné celé číslo)-výstup

Další informace o datových typech struktury v souboru “Základní datové typy” na stránce 993.

Směr parametru může být:

Vstup
Tento parametr musíte zadat vy (programátor).

Výstup
Volání vrátí tento parametr.

Vstup a výstup
Tento parametr musíte zadat, ale je upraven voláním.

K dispozici je také stručný popis účelu parametru spolu se seznamem všech hodnot, které parametr může mít.

Poslední dva parametry v každém volání jsou kód dokončení a kód příčiny. Kód dokončení označuje, zda bylo volání úspěšně, částečně nebo vůbec dokončeno. Další informace o částečném úspěchu nebo selhání volání jsou uvedeny v kódu příčiny.

Poznámky k použití
Další informace o volání, popisující jeho použití a případná omezení jeho použití.

Vyvolání RPG
Typické vyvolání volání a deklarace jeho parametrů v RPG.

Další notářské konvence jsou:

Konstanty
Názvy konstant jsou uvedeny velkými písmeny; například OOOUT.

Pole
V některých voláních jsou parametry pole znakových řetězců s velikostí, která není pevná. V popisech těchto parametrů představuje malá písmena *n* číselnou konstantu. Při kódování deklarace pro tento parametr nahraďte hodnotu *n* požadovanou číselnou hodnotou.

IBM i MQBACK (vrácení změn) na IBM i

Volání MQBACK označuje správci front, že všechny zprávy, které se vyskytly od posledního synchronizačního bodu, mají být vráceny zpět. Zprávy vložené jako součást pracovní jednotky jsou odstraněny; zprávy načtené jako součást pracovní jednotky jsou obnoveny do fronty.

- Toto volání je podporováno v následujících prostředích:
  - AIX AIX
  - IBM i IBM i
  - Windows Windows
- “Syntaxe” na stránce 1245
- “Poznámky k použití” na stránce 1245
- “Parametry” na stránce 1246
- “Prohlášení o RPG” na stránce 1247

Syntaxe

MQBACK *(Hconn, CompCode, Reason)*

Poznámky k použití

Při použití příkazu MQBACK zvažte tyto poznámky k použití.

1. Toto volání lze použít pouze v případě, že správce front sám koordinuje pracovní jednotku. Jedná se o lokální pracovní jednotku, kde změny ovlivňují pouze prostředky IBM MQ .
2. V prostředích, kde správce front nekoordinuje jednotku práce, musí být místo MQBACK použito odpovídající zpětné volání. Prostředí může také podporovat implicitní vrácení zpět způsobené nestandardním ukončením aplikace.
   - V systému IBM ilze toto volání použít pro lokální pracovní jednotky koordinované správcem front. To znamená, že na úrovni úlohy nesmí existovat definice vázaného zpracování, to znamená, že pro úlohu nesmí být vydán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** .
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, odebrání těchto změn závisí na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v části “MQDISC (Odpojit správce front) na systému IBM i” na stránce 1283 .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, správce front uchovává informace týkající se skupiny zpráv a logické zprávy pro poslední úspěšná volání MQPUT a MQGET. Tyto informace jsou přidruženy k popisovači fronty a zahrnují například:
   - Hodnoty polí *MDGID*, *MDSEQ*, *MDOFF*a *MDMFL* v deskriptoru MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo dočasná.

   Správce front uchovává *tři* sady informací o skupinách a segmentech, jednu sadu pro každou z následujících položek:

- Poslední úspěšné volání MQPUT (může být součástí pracovní jednotky).
- Poslední úspěšné volání MQGET, které odebrala zprávu z fronty (může být součástí pracovní jednotky).
- Poslední úspěšné volání MQGET, které procházelo zprávu ve frontě (nemůže být součástí pracovní jednotky).

Pokud aplikace vloží nebo získá zprávy jako součást jednotky práce a pak se rozhodne vrátit zpět jednotku práce, informace o skupině a segmentu se obnoví na hodnotu, kterou měla dříve:

- Informace přidružené k volání MQPUT jsou obnoveny na hodnotu, kterou měly před prvním úspěšným voláním MQPUT pro daný popisovač fronty v aktuální transakci.
- Informace přidružené k volání MQGET jsou obnoveny na hodnotu, kterou měly před prvním úspěšným voláním MQGET pro daný manipulátor fronty v aktuální transakci.

Fronty, které byly aktualizovány aplikací poté, co byla jednotka práce spuštěna, ale mimo rozsah jednotky práce, nemají obnoveny informace o skupinách a segmentech, pokud je jednotka práce odvolána.

Obnova informací o skupině a segmentu na předchozí hodnotu, když je jednotka práce odvolána, umožňuje aplikaci rozložit velkou skupinu zpráv nebo velkou logickou zprávu sestávající z mnoha segmentů do několika jednotek práce a restartovat ji ve správném bodě ve skupině zpráv nebo logické zprávě, pokud jedna z jednotek práce selže. Použití několika pracovních jednotek může být výhodné, pokud má lokální správce front pouze omezené úložiště front. Aplikace však musí udržovat dostatečné informace, aby mohla znovu spustit vkládání nebo získávání zpráv ve správném bodě, dojde-li k selhání systému. Podrobnosti o tom, jak provést restart ve správném bodě po selhání systému, viz volba PMLOGO popsaná v části “MQPMO (Vložit-volby zpráv) na IBM i” na stránce 1168a volba GMLOGO popsaná v části “MQGMO (Volby Get-message) na systému IBM i” na stránce 1073.

Zbývající poznámky k použití platí pouze v případě, že správce front koordinuje pracovní jednotky:

1. Jednotka práce má stejný rozsah jako manipulátor připojení. To znamená, že všechna volání IBM MQ , která ovlivňují konkrétní pracovní jednotku, musí být provedena pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného manipulátoru připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu popisovačů připojení viz parametr **HCONN** popsaný v části “MQCONN (Připojit správce front) na systému IBM i” na stránce 1270 .
2. Toto volání ovlivní pouze zprávy, které byly vloženy nebo načteny jako součást aktuální pracovní jednotky.
3. Přerušitelná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale která nikdy nevydá volání potvrzení nebo odvolání, může způsobit, že se fronty zaplní zprávami, které nejsou k dispozici pro jiné aplikace. V zájmu ochrany před touto možností by měl administrátor nastavit atribut správce front **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránil tomu, že by nepoužívané aplikace zaplňovaly fronty, ale dostatečně vysoká, aby umožňovala, aby očekávané aplikace systému zpráv správně fungovaly.

Parametry

Volání MQBACK má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *COMCOD*.

Pokud je *COMCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *COMCOD* hodnotu CCFAIL:

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2123
(2123, X'84B') Výsledek operace potvrzení nebo vrácení je smíšený.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                   CALLP     MQBACK(HCONN : COMCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQBACK           PR                  EXTPROC('MQBACK')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Completion code
D COMCOD                        10I 0
D* Reason code qualifying COMCOD
D REASON                        10I 0
```

IBM i MQBEGIN (začátek transakce) na IBM i

Volání MQBEGIN zahájí pracovní jednotku, která je koordinována správcem front a která může zahrnovat externí správce prostředků.

- Toto volání je podporováno v následujících prostředích:

-  AIX
-  IBM i
-  Windows

Syntaxe

MQBEGIN *(HCONN, BEGOP, CMPCOD, REASON)*

Poznámky k použití

1. Volání MQBEGIN lze použít ke spuštění pracovní jednotky, která je koordinována správcem front a která může zahrnovat změny prostředků vlastněných jinými správci prostředků. Správce front podporuje tři typy pracovních jednotek:

   Lokální jednotka práce koordinovaná správcem front
   Jedná se o pracovní jednotku, ve které je správce front jediným účastníkem správce prostředků, a proto tento správce front vystupuje jako koordinátor jednotky práce.

   - Chcete-li spustit tento typ pracovní jednotky, je třeba při prvním volání MQPUT, MQPUT1nebo MQGET v pracovní jednotce zadat volbu PMSYP nebo GMSYP.

     Není nutné, aby aplikace vyvolala volání MQBEGIN ke spuštění pracovní jednotky, ale pokud se použije MQBEGIN, volání se dokončí s CCWARN a kódem příčiny RC2121.
   - Chcete-li tento typ práce potvrdit nebo vrátit zpět, musíte použít volání MQCMIT nebo MQBACK.

   Globální jednotka práce koordinovaná správcem front
   Jedná se o pracovní jednotku, ve které správce front vystupuje jako koordinátor jednotky práce, a to jak pro IBM MQ prostředky *, tak pro* prostředky náležející jiným správcům prostředků. Tito správci prostředků spolupracují se správcem front, aby zajistili, že všechny změny prostředků v pracovní jednotce budou potvrzeny nebo vráceny zpět společně.

   - Chcete-li spustit tento typ pracovní jednotky, musíte použít volání MQBEGIN.
   - Chcete-li potvrdit nebo vrátit tento typ pracovní jednotky, musíte použít volání MQCMIT a MQBACK.

   Externě koordinovaná globální jednotka práce
   Jedná se o pracovní jednotku, ve které je správce front účastníkem, ale správce front nepůsobí jako koordinátor pracovní jednotky. Místo toho existuje externí koordinátor pracovních jednotek, se kterým správce front spolupracuje.

   - Chcete-li spustit tento typ pracovní jednotky, musíte použít příslušné volání poskytnuté externím koordinátorem pracovní jednotky.

     Pokud se volání MQBEGIN používá k pokusu o spuštění pracovní jednotky, volání selže s kódem příčiny RC2012.
   - Chcete-li potvrdit nebo vrátit tento typ pracovní jednotky, musí být použita volání potvrzení a vrácení poskytnutá externím koordinátorem pracovní jednotky.

     Pokud se volání MQCMIT nebo MQBACK používá k pokusu o potvrzení nebo vrácení transakce, volání selže s kódem příčiny RC2012.
2. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, odebrání těchto změn závisí na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v části "MQDISC (Odpojit správce front) na systému IBM i" na stránce 1283 .

3. Aplikace se může v daném okamžiku účastnit pouze jedné jednotky práce. Volání MQBEGIN se nezdaří s kódem příčiny RC2128 , pokud již pro aplikaci existuje jednotka práce, bez ohledu na to, jaký typ práce je.
4. Volání MQBEGIN není platné v prostředí klienta IBM MQ . Pokus o použití volání selže s kódem příčiny RC2012.
5. Pokud správce front vystupuje jako koordinátor pracovních jednotek pro globální pracovní jednotky, jsou správci prostředků, kteří se mohou podílet na transakci, definováni v konfiguračním souboru správce front.
6. V systému IBM ijsou podporovány tři typy jednotek práce:
   - **Lokální jednotky práce koordinované správcem front** lze použít pouze v případě, že na úrovni úlohy neexistuje definice vázaného zpracování, tj. že pro úlohu nebyl zadán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** .
   - **Globální pracovní jednotky koordinované správcem front** nejsou podporovány.
   - **Externě koordinované globální pracovní jednotky** lze použít pouze v případě, že na úrovni úlohy existuje definice vázaného zpracování, to znamená, že pro úlohu musí být zadán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** . Pokud k tomu došlo, operace IBM i COMMIT a ROLLBACK se vztahují na prostředky IBM MQ i na prostředky patřící ostatním zúčastněným správcům prostředků.

Parametry

Volání MQBEGIN má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

BEGOP (MQBO)-vstup/výstup

Volby, které řídí akci MQBEGIN.

Podrobnosti viz “MQBO (začátek voleb) na IBM i” na stránce 1015.

Nejsou-li vyžadovány žádné volby, mohou programy napsané v jazyku C nebo sestavovacím modulu S/390 zadat adresu parametru s hodnotou Null namísto adresy struktury MQBO.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCWARN:

RC2121
(2121, X'849 ') Nejsou registrováni žádní zúčastnění správci prostředků.

RC2122
(2122, X'84A') Participující správce prostředků není k dispozici.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2134
(2134, X'856 ') Struktura začátku voleb není platná.

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2012
(2012, X'7DC') Volání není v prostředí platné.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2128
(2128, X'850 ') Jednotka práce již byla spuštěna.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQBEGIN(HCONN : BEGOP : CMPCOD :
C                                       REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQBEGIN          PR                  EXTPROC('MQBEGIN')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Options that control the action of MQBEGIN
D BEGOP                         12A
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CMPCOD
D REASON                        10I 0
```

IBM i MQBUFMH (Převést vyrovnávací paměť na popisovač zprávy) na IBM i

Volání funkce MQBUFMH převádí vyrovnávací paměť na manipulátor zprávy a je inverzní funkcí volání MQMHBUF.

Toto volání vezme deskriptor zprávy a vlastnosti MQRFH2 do vyrovnávací paměti a zpřístupní je prostřednictvím manipulátoru zprávy. Vlastnosti MQRFH2 v datech zprávy jsou volitelně odebrány. Pole

Encoding, *CodedCharSetId*a *Format* deskriptoru zprávy jsou v případě potřeby aktualizována tak, aby správně popisovala obsah vyrovnávací paměti po odebrání vlastností.

Syntaxe

MQBUFMH *(Hconn, Hmsg, BufMsgHOpts, MsgDesc, Buffer, BufferLength, DataLength, CompCode, Reason)*

Poznámky k použití

Volání MQBUFMH nelze zachytit prostřednictvím uživatelských procedur rozhraní API-vyrovnávací paměť je převedena na manipulátor zprávy v prostoru aplikace; volání nedosáhne správce front.

Parametry

Volání MQBUFMH má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podprocesu, který převádí vyrovnávací paměť na popisovač zprávy. Pokud není ustanoveno platné připojení, volání selže s RC2009.

HMSG (20místné celé číslo se znaménkem)-vstup

Tento manipulátor je manipulátor zpráv, pro který je vyžadována vyrovnávací paměť. Hodnota byla vrácena předchozím voláním MQCRTMH.

BMHOPT (MQBMHO)-vstup

Struktura MQBMHO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou manipulátory zpráv vytvářeny z vyrovnávacích pamětí.

Podrobnosti viz "MQBMHO (volby vyrovnávací paměti pro obsluhu zpráv) v systému IBM i" na stránce 1013.

MSGDSC (MQMD)-vstupní/výstupní

Struktura *MSGDSC* obsahuje vlastnosti deskriptoru zpráv a popisuje obsah oblasti vyrovnávací paměti.

Při výstupu z volání jsou vlastnosti volitelně odebrány z oblasti vyrovnávací paměti a v tomto případě je deskriptor zprávy aktualizován tak, aby správně popisoval oblast vyrovnávací paměti.

Data v této struktuře musí být ve znakové sadě a kódování aplikace.

BUFLEN (10místné celé číslo se znaménkem)-vstup

BUFLEN je délka oblasti vyrovnávací paměti v bajtech.

Hodnota *BUFLEN* s nulovým počtem bajtů je platná a označuje, že oblast vyrovnávací paměti neobsahuje žádná data.

BUFFER (1bajtový bitový řetězec x BUFLEN)-vstup/výstup

BUFFER definuje oblast obsahující vyrovnávací paměť zpráv. Pro většinu dat je nutné vyrovnávací paměť zarovnat na 4bajtovou hranici.

Pokud *BUFFER* obsahuje znaková nebo číselná data, nastavte pole *CodedCharSetId* a *Encoding* v parametru **MSGDSC** na hodnoty odpovídající datům; to v případě potřeby umožní převod dat.

Pokud jsou ve vyrovnávací paměti zpráv nalezeny vlastnosti, jsou volitelně odebrány; později budou k dispozici v popisovači zprávy při návratu z volání.

V programovacím jazyku C je parametr deklarován jako ukazatel na prázdno, což znamená, že jako parametr lze zadat adresu libovolného typu dat.

Je-li parametr **BUFLEN** nulový, na *BUFFER* se neodkazuje. V tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo v sestavovacím modulu System/390 hodnotu null.

DATLEN (10místné celé číslo se znaménkem)-výstup

DATLEN je délka (v bajtech) vyrovnávací paměti, která může mít odebrané vlastnosti.

CMPCOD (10místné celé číslo se znaménkem)-výstup

CCOK
Úspěšné dokončení.

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

RC2489
(2489, X'09B9') Struktura voleb vyrovnávací paměti pro obsluhu zpráv je neplatná.

RC2004
(2004, X'07D4') Parametr vyrovnávací paměti není platný.

RC2005
(2005, X'07D5') Parametr délky vyrovnávací paměti není platný.

RC2219
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

RC2009
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

RC2460
(2460, X'099C') popisovač zprávy není platný.

RC2026
(2026, X'07EA') Deskriptor zpráv není platný.

RC2499
(2499, X'09C3') Obsluha zprávy je již používána.

RC2046
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2334
(2334, X'091E') Struktura MQRFH2 není platná.

RC2421
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nelze analyzovat.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQBUFMH(HCONN : HMSG : BMHOPT :
                                        MSGDSC : BUFLEN : BUFFER :
                                        DATLEN : CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQBUFMH          PR                  EXTPROC('MQBUFMH')
D* Connection handle
D HCONN                         10I 0
D* Message handle
D HMSG                          10I 0
D* Options that control the action of MQBUFMH
D BMHOPT                        12A   VALUE
D* Message descriptor
D MSGDSC                       364A
D* Length in bytes of the Buffer area
D BUFLEN                        10I 0
D* Area to contain the message buffer
D BUFFER                          *   VALUE
D* Length of the output buffer
D DATLEN                        10I 0
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CompCode
D REASON                        10I 0
```

IBM i MQCB (Správa zpětného volání) na IBM i

Volání MQCB znovu registruje zpětné volání pro určený manipulátor objektu a řídí aktivaci a změny zpětného volání.

Zpětné volání je část kódu (určená buď jako název funkce, která může být dynamicky propojena, nebo jako ukazatel funkce), který je volán funkcí IBM MQ , když dojde k určitým událostem.

Chcete-li používat MQCB a MQCTL na klientu V7 , musíte být připojeni k serveru V7 a parametr **SHARECNV** kanálu musí mít nenulovou hodnotu.

Informace o globálních jednotkách práce viz: Globální jednotky práce.

Typy zpětného volání, které lze definovat, jsou:

Spotřebitel zpráv.
Funkce zpětného volání spotřebitele zpráv je volána, když je zpráva splňující zadaná kritéria výběru k dispozici na popisovači objektu.

Pro každý popisovač objektu lze registrovat pouze jednu funkci zpětného volání. Má-li být přečtena jedna fronta s více výběrovými kritérii, musí být fronta otevřena vícekrát a na každém popisovači musí být registrována funkce spotřebitele.

obslužná rutina událostí
Obslužná rutina událostí je volána pro podmínky, které ovlivňují celé prostředí zpětného volání.

Funkce je volána při výskytu podmínky události, například při zastavení nebo uvedení do klidového stavu správce front nebo připojení.

Funkce není volána pro podmínky, které jsou specifické pro jednoho spotřebitele zpráv, například RC2016; je však volána, pokud funkce zpětného volání nekončí normálně.

Syntaxe

MQCB *(HCONN, OPERATN, HOBJ, CBDSC, MSGDSC, GMO, CMPCOD, REASON)*

Poznámky k použití pro MQCB

1. MQCB se používá k definování akce, která má být vyvolána pro každou zprávu, odpovídající zadaným kritériím, která je k dispozici ve frontě. Když je akce zpracována, buď je zpráva odebrána z fronty a předána definovanému spotřebiteli zpráv, nebo je poskytnut token zprávy, který se používá k načtení zprávy.
2. Funkci MQCB lze použít k definování rutin zpětného volání před spuštěním spotřeby s funkcí MQCTL nebo ji lze použít z rutiny zpětného volání.
3. Chcete-li používat funkci MQCB mimo rutinu zpětného volání, musíte nejprve pozastavit spotřebu zpráv pomocí funkce MQCTL a poté obnovit spotřebu.

Posloupnost zpětného volání spotřebitele zpráv

Spotřebitele můžete nakonfigurovat tak, aby vyvolal zpětné volání v klíčových bodech během životního cyklu spotřebitele. Příklad:

- kdy je spotřebitel poprvé registrován,
- když je připojení spuštěno,
- když je připojení zastaveno a
- když je odběratel odregistrován, buď explicitně, nebo implicitně pomocí MQCLOSE.

Tabulka 742. Definice příkazových slov MQCTL

Sloveso	**Význam**
MQCTL (začátek)	Volání MQCTL pomocí operace CTLSR
MQCTL (STOP)	Volání MQCTL pomocí operace CTLSP
MQCTL (WAIT)	Volání MQCTL pomocí operace CTLSW

Umožňuje spotřebiteli udržovat stav přidružený ke spotřebiteli. Když aplikace požaduje zpětné volání, pravidla pro vyvolání spotřebitele jsou následující:

Registrovat
Jedná se vždy o první typ vyvolání zpětného volání.

Je vždy volána ve stejném podprocesu jako volání MQCB (CBREG).

SPUSTIT
Je vždy volána synchronně se slovesem MQCTL (START).

- Všechna zpětná volání START jsou dokončena před vrácením příkazu MQCTL (START).

Je na stejném podprocesu jako doručení zprávy, pokud je požadováno CTLTHR.

Volání se spuštěním není zaručeno, pokud například předchozí zpětné volání během MQCTL (START) vydá příkaz MQCTL (STOP).

ZASTAVIT
Po tomto volání nebudou doručeny žádné další zprávy nebo události, dokud nebude připojení restartováno.

Příkaz STOP je zaručen v případě, že aplikace byla dříve volána pro příkaz START, zprávu nebo událost.

DEREGISTER (REGISTROVAT)
Jedná se vždy o poslední typ vyvolání zpětného volání.

Ujistěte se, že vaše aplikace provádí inicializaci založenou na podprocesech a vyčištění ve zpětných voláních START a STOP. Pomocí zpětných volání REGISTER a DEREGISTER můžete provést inicializaci a vyčištění bez použití podprocesů.

Nedělejte žádné jiné předpoklady o životnosti a dostupnosti závitu než to, co je uvedeno. Nespoléhejte se například na to, že podproces zůstane aktivní i po posledním volání příkazu DEREGISTER. Podobně, pokud jste se rozhodli nepoužívat CTLTHR, nepředpokládejte, že podproces existuje při každém spuštění připojení.

Pokud má vaše aplikace konkrétní požadavky na charakteristiku podprocesu, může vždy vytvořit podproces odpovídajícím způsobem a poté použít MQCTL (WAIT). Tento krok *daruje* podproces IBM MQ pro asynchronní doručování zpráv.

Využití připojení spotřebitele zpráv

Obvykle, když aplikace zadá další volání MQI, zatímco je jedno nevyřízené, volání selže s kódem příčiny RC2219.

Existují však zvláštní případy, kdy aplikace musí před dokončením předchozího volání vydat další volání MQI. Spotřebitele lze například vyvolat během volání MQCB s CBRE.

V takové instanci platí, že pokud je v důsledku aplikace vydávající příkaz MQCB nebo MQCTL aplikace volána zpět, může aplikace zadat další volání MQI. Tato instance znamená, že můžete například zadat volání MQOPEN ve funkci spotřebitele při volání s typem CBCCALLT CBCTRC. Jakékoli volání MQI, s výjimkou volání MQDISC, je povoleno.

Parametry pro MQCB

Volání MQCB má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Spravovat funkci zpětného volání-parametr HCONN.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

OPERATN (10místné celé číslo se znaménkem)-vstup

Spravovat funkci zpětného volání-parametr OPERATN.

Operace, která se zpracovává na zpětném volání definovaném pro uvedený popisovač objektu. Musíte uvést jednu z následujících voleb; pokud je požadována více než jedna volba, hodnoty lze přidat (nepřidávat stejnou konstantu více než jednou) nebo kombinovat pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Kombinace, které nejsou platné, jsou uvedeny; všechny ostatní kombinace jsou platné.

CBREG
Definujte funkci zpětného volání pro zadaný popisovač objektu. Tato operace definuje funkci, která se má volat, a kritéria výběru, která se mají použít.

Je-li pro manipulátor objektu již definována funkce zpětného volání, je definice nahrazena. Je-li zjištěna chyba při nahrazování zpětného volání, je zrušena registrace funkce.

Pokud je zpětné volání registrováno ve stejné funkci zpětného volání, ve které bylo dříve odregistrováno, je toto považováno za operaci nahrazení; žádná počáteční nebo konečná volání nejsou vyvolána.

Můžete použít CBREG s CTLSU nebo CTLRE.

CBUNR
Zastavte spotřebu zpráv pro manipulátor objektu a odeberte manipulátor z těch, které jsou vhodné pro zpětné volání.

Pokud je přidružený popisovač zavřený, je automaticky zrušena registrace zpětného volání.

Je-li funkce CBUNR volána ze spotřebitele a zpětné volání má definováno volání zastavení, je vyvolána při návratu ze spotřebitele.

Pokud je tato operace vydána pro *Hobj* bez registrovaného spotřebitele, volání se vrátí s RC2448.

CTLSU
Pozastaví spotřebu zpráv pro popisovač objektu.

Pokud je tato operace použita na obslužnou rutinu událostí, obslužná rutina událostí nezíská události během pozastavení a všechny události, které zmeškaly v pozastaveném stavu, nejsou při obnovení operace poskytnuty.

Zatímco je funkce spotřebitele pozastavena, pokračuje v získání zpětných volání typu řízení.

CTLRE
Obnovte spotřebu zpráv pro popisovač objektu.

Pokud je tato operace použita na obslužnou rutinu událostí, obslužná rutina událostí nezíská události během pozastavení a všechny události, které zmeškaly v pozastaveném stavu, nejsou při obnovení operace poskytnuty.

CBDSC (MQCBD)-vstup

Správa funkce zpětného volání-parametr CBDSC.

Jedná se o strukturu, která identifikuje funkci zpětného volání, která je registrována aplikací, a volby použité při její registraci.

Podrobnosti o struktuře viz “MQCBD-Deskriptor zpětného volání” na stránce 290 .

Deskriptor zpětného volání je vyžadován pouze pro volbu CBREG; pokud deskriptor není požadován, předaná adresa parametru může být null.

HOBJ (10místné celé číslo se znaménkem)-vstup

Spravovat funkci zpětného volání-parametr HOBJ.

Tento popisovač představuje přístup, který byl ustanoven k objektu, ze kterého má být zpráva využita. Jedná se o popisovač, který byl vrácen z předchozího volání MQOPEN nebo MQSUB (v parametru **HOBJ**).

Parametr *HOBJ* není vyžadován při definování rutiny obslužné rutiny událostí (CBTEH) a musí být určen jako HONONE.

Pokud byl tento příkaz *Hobj* vrácen z volání MQOPEN, musí být fronta otevřena s jednou nebo více z následujících voleb:

- OOINPS
- OOINPX
- OOINPQ
- OOBRW

MSGDSC (MQMD)-vstup

Spravovat funkci zpětného volání -MSGDSC parametr.

Tato struktura popisuje atributy požadované zprávy a atributy načtené zprávy.

Parametr **MsgDesc** definuje atributy zpráv požadovaných spotřebitelem a verzi MQMD, která má být předána spotřebiteli zpráv.

Volby *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber*a *Offset* v deskriptoru MQMD se používají pro výběr zpráv v závislosti na volbách zadaných v parametru **GetMsgOpts** .

Hodnoty *Encoding* a *CodedCharSetId* se používají pro převod zpráv, pokud zadáte volbu GMCONV.

Podrobnosti viz MQMD .

MsgDesc se používá pouze pro CBREG, a pokud požadujete jiné hodnoty než výchozí pro libovolná pole. *MsgDesc* se nepoužívá pro obslužnou rutinu událostí.

Pokud deskriptor není požadován, předaná adresa parametru může být null.

Všimněte si, že pokud je pro stejnou frontu registrováno více spotřebitelů s překrývajícími se selektory, není vybraný spotřebitel pro každou zprávu definován.

GMO (MQGMO)-vstup

Spravovat funkci zpětného volání-parametr GMO.

Volby, které řídí způsob, jakým spotřebitel zpráv získává zprávy.

Všechny volby mají význam popsaný v části "MQGMO (Volby Get-message) na systému IBM i" na stránce 1073, jsou-li použity pro volání MQGET, s výjimkou následujících:

GMSSIG
Tato volba není povolena.

GMBRWF, GMBRWN, GMMBH, GMMBC
Pořadí zpráv doručených spotřebiteli procházení je určeno kombinacemi těchto voleb. Významné kombinace jsou:

GMBRWF
První zpráva ve frontě je opakovaně doručena spotřebiteli. To je užitečné, když spotřebitel destruktivně spotřebuje zprávu ve zpětném volání. Tuto volbu používejte opatrně.

GMBRWN
Odběratel obdrží každou zprávu ve frontě od aktuální pozice kurzoru až po dosažení konce fronty.

GMBRWF + GMBRWN
Kurzor se resetuje na začátek fronty. Odběratel pak obdrží každou zprávu, dokud kurzor nedosáhne konce fronty.

GMBRWF + GMMBH nebo GMMBC
Počínaje začátkem fronty je spotřebiteli předána první neoznačená zpráva ve frontě, která je poté označena pro tohoto spotřebitele. Tato kombinace zajišťuje, že spotřebitel může přijímat nové zprávy přidané za aktuální bod kurzoru.

GMBRWN + GMMBH nebo GMMBC
Počínaje pozicí kurzoru je spotřebiteli poskytnuta další neoznačená zpráva ve frontě, která je pak označena pro tohoto spotřebitele. Tuto kombinaci používejte opatrně, protože zprávy lze přidat do fronty za aktuální pozici kurzoru.

GMBRWF + GMBRWN + GMMBH nebo GMMBC
Tato kombinace není povolena, je-li použita, volání vrátí hodnotu RC2046.

GMNWT, GMWT a GMWI
Tyto volby řídí způsob vyvolání spotřebitele.

GMNWT
Odběratel není nikdy volán s RC2033. Spotřebitel je vyvolán pouze pro zprávy a události

GMWT s nulovým GMWI
Kód RC2033 je předán spotřebiteli pouze v případě, že neexistují žádné zprávy a

- spotřebitel byl spuštěn
- spotřebitel byl doručen alespoň jednu zprávu od posledního kódu příčiny bez zpráv.

To zabrání odběrateli v systému výzev v vytížené smyčce, když je uveden nulový interval čekání.

GMWT a pozitivní GMWI
Uživatel je vyvolán po uvedeném intervalu čekání s kódem příčiny RC2033. Toto volání se provádí bez ohledu na to, zda byly spotřebiteli doručeny nějaké zprávy. To umožňuje uživateli provést zpracování prezenčního signálu nebo dávkového zpracování.

GMWT a GMWI z WIULIM
Uvádí nekonečné čekání před návratem RC2033. Odběratel není nikdy volán s RC2033.

GMO se používá pouze pro CBREG, a pokud požadujete jiné hodnoty než výchozí pro libovolná pole. *GMO* se nepoužívá pro obslužnou rutinu událostí.

Pokud nejsou volby požadovány, předaná adresa parametru může být null.

Je-li ve struktuře MQGMO poskytnut popisovač vlastností zprávy, je kopie poskytnuta ve struktuře MQGMO, která je předána do zpětného volání spotřebitele. Při návratu z volání MQCB může aplikace odstranit popisovač vlastností zprávy.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Spravovat funkci zpětného volání-parametr CMPCOD.

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Spravovat funkci zpětného volání-parametr REASON.

Následující kódy příčiny jsou kódy, které může správce front vrátit pro parametr **REASON** .

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu CCFAIL:

RC2204
(2204, X'89C') Adaptér není k dispozici.

RC2133
(2133, X'855 ') Nelze načíst moduly služeb převodu dat.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2374
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

RC2183
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

RC2157
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

RC2005
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

RC2219
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

RC2487
(2487, X'9B7') Nesprávné pole typu zpětného volání.

RC2448
(2448, X' 990 ') Nelze zrušit registraci, pozastavit nebo obnovit, protože neexistuje žádné registrované zpětné volání.

RC2486
(2486, X'9B6') Musí být uveden buď *CallbackFunction* , nebo *CallbackName* , ale ne obojí.

RC2483
(2483, X'9B3') Nesprávné pole typu zpětného volání.

RC2484
(2484, X'9B4') Chybné pole voleb MQCBD.

RC2140
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2217
(2217, X'8A9') Není autorizováno pro připojení.

RC2202
(2202, X'89A') Uklidení připojení.

RC2203
(2203, X'89B') Probíhá ukončování připojení.

RC2207
(2207, X'89F') Chyba identifikátoru korelace.

RC2010
(2010, X'7DA') Parametr délky dat není platný.

RC2016
(2016, X'7E0') Získá blokováno pro frontu.

RC2351
(2351, X'92F') Globální konflikt pracovních jednotek.

RC2186
(2186, X'88A') Struktura voleb Get-message není platná.

RC2353
(2353, X' 931 ') Rukojeť používaná pro globální jednotku práce.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2019
(2019, X'7E3') popisovač objektu není platný.

RC2259
(2259, X'8D3') Nekonzistentní specifikace procházení.

RC2245
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

RC2246
(2246, X'8C6') Zpráva pod kurzorem není platná pro načtení.

RC2352
(2352, X' 930 ') Globální jednotka práce je v konfliktu s lokální jednotkou práce.

RC2247
(2247, X'8C7') Volby shody nejsou platné.

RC2485
(2485, X'9B4') Chybné *MaxMsgLength* pole.

RC2026
(2026, X'7EA') Deskriptor zpráv není platný.

RC2497
(2497, X'9C1') Zadaný vstupní bod funkce nebyl v modulu nalezen.

RC2496
(2496, X'9C0') Modul byl nalezen, ale je nesprávného typu; není 32bitový, 64bitový nebo má platnou knihovnu dynamického odkazu.

RC2495
(2495, X'9BF') Modul nebyl nalezen ve vyhledávací cestě nebo není autorizován k načtení.

RC2250
(2250, X'8CA') Pořadové číslo zprávy není platné.

RC2331
(2331, X'91B') Použití tokenu zprávy není platné.

RC2033
(2033, X'7F1') Není k dispozici žádná zpráva.

RC2034
(2034, X'7F2') Kurzor procházení není umístěn na zprávě.

RC2036
(2036, X'7F4') Fronta není otevřena pro procházení.

RC2037
(2037, X'7F5') Fronta není otevřena pro vstup.

RC2041
(2041, X'7F9') Definice objektu se od doby otevření změnila.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2206
(2206, X'89E') Chybný kód operace ve volání rozhraní API.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2193
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2394
(2394, X'95A') Fronta má chybný typ indexu.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2161
(2161, X'871 ') Správce front je uveden do klidového stavu.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2069
(2069, X'815 ') Pro tuto rukojeť je signál vynikající.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2109
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

RC2024
(2024, X'7E8') V rámci aktuální pracovní jednotky nelze zpracovat žádné další zprávy.

RC2072
(2072, X'818 ') Podpora synchronizační funkce není k dispozici.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2354
(2354, X' 932 ') Registrace v globální pracovní jednotce se nezdařila.

RC2355
(2355, X' 933 ') Směs volání jednotky práce není podporována.

RC2255
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

RC2090
(2090, X'82A') Interval čekání v MQGMO není platný.

RC2256
(2256, X'8D0') Dodána nesprávná verze MQGMO.

RC2257
(2257, X'8D1') Dodána nesprávná verze MQMD.

RC2298
(2298, X'8FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                    CALLP     MQCB(HCONN : OPERATN : CBDSC :
                               HOBJ : MSGDSC : GMO :
                               DATLEN : CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQCB             PR                  EXTPROC('MQCB')
D* Connection handle
D HCONN                          10I 0 VALUE
D* Operation
D OPERATN                        10I 0 VALUE
D* Callback descriptor
D CBDSC                         180A
D* Object handle
D HOBJ                           10I 0 VALUE
D* Message Descriptor
D MSGDSC                        364A
D* Get options
D GMO                           112A
D* Completion code
D CMPCOD                         10I 0
* Reason code qualifying CompCode
D REASON                         10I 0
```

IBM i MQCLOSE (Zavřít objekt) na IBM i

Volání MQCLOSE znovu ukončí přístup k objektu a je inverzní k volání MQOPEN.

- "Prohlášení o RPG" na stránce 1267

Syntaxe

MQCLOSE *(HCONN, HOBJ, OPTS, CMPCOD, REASON)*

Poznámky k použití

1. Když aplikace zadá volání MQDISC nebo skončí buď normálně, nebo abnormálně, všechny objekty, které byly otevřeny aplikací a jsou stále otevřené, se automaticky zavřou s volbou CONONE.
2. Následující body platí v případě, že zavřený objekt je *fronta*:
   - Pokud jsou operace ve frontě prováděny jako součást pracovní jednotky, lze frontu zavřít před nebo po výskytu synchronizačního bodu bez ovlivnění výsledku synchronizačního bodu.
   - Pokud byla fronta otevřena s volbou OOBRW, je kurzor procházení zničen. Pokud je fronta později znovu otevřena s volbou OOBRW, vytvoří se nový kurzor procházení (viz volba OOBRW popsaná v MQOPEN).
   - Je-li pro tento manipulátor v době volání MQCLOSE aktuálně uzamknuta zpráva, bude zámek uvolněn (viz volba GMLK popsaná v tématu "MQGMO (Volby Get-message) na systému IBM i" na stránce 1073).
3. Následující body platí v případě, že zavřený objekt je *dynamická fronta* (buď trvalá, nebo dočasná):
   - Pro dynamickou frontu lze zadat volby CODEL nebo COPURG bez ohledu na volby zadané v odpovídajícím volání MQOPEN.
   - Když je dynamická fronta odstraněna, všechna volání MQGET s volbou GMWT, která jsou vůči frontě nevyřízená, jsou zrušena a vrátí se kód příčiny RC2052 . Viz volba GMWT popsaná v části "MQGMO (Volby Get-message) na systému IBM i" na stránce 1073.

     Po odstranění dynamické fronty selže jakékoli volání (jiné než MQCLOSE), které se pokusí odkazovat na frontu pomocí dříve získaného popisovače *HOBJ* , s kódem příčiny RC2052.

     Mějte na paměti, že ačkoli aplikace nemají přístup k odstraněné frontě, fronta není odebrána ze systému a přidružené prostředky nejsou uvolněny, dokud nebudou uzavřeny všechny manipulátory, které odkazují na frontu, a všechny jednotky práce, které mají vliv na frontu, nebyly potvrzeny nebo vráceny zpět.
   - Pokud při odstranění trvalé dynamické fronty není manipulátor *HOBJ* určený ve volání MQCLOSE vrácen voláním MQOPEN, které vytvořilo frontu, je provedena kontrola, zda je identifikátor uživatele, který byl použit k ověření volání MQOPEN, autorizován k odstranění fronty. Pokud byla ve volání MQOPEN zadána volba OOALTU, je kontrolovaný identifikátor uživatele *ODAU*.

     Tato kontrola se neprovádí, pokud:
     - Uvedený popisovač je ten, který vrátil volání MQOPEN, které vytvořilo frontu.
     - Odstraňovaná fronta je dočasná dynamická fronta.
   - Pokud je při zavření dočasné dynamické fronty manipulátor *HOBJ* určený ve volání MQCLOSE vrácen voláním MQOPEN, které vytvořilo frontu, bude fronta odstraněna. K tomu dochází bez ohledu na volby zavření zadané ve volání MQCLOSE. Pokud jsou ve frontě zprávy, jsou vyřazeny; žádné zprávy sestavy se nevygenerují.

     Pokud existují nepotvrzené jednotky práce, které ovlivňují frontu, fronta a její zprávy jsou stále odstraněny, ale to nezpůsobí selhání jednotek práce. Avšak, jak bylo popsáno dříve, prostředky přidružené k jednotkám práce se neuvolní, dokud nebudou jednotlivé jednotky práce buď potvrzeny, nebo vráceny zpět.
4. Následující body platí, pokud je uzavírána *distribuční seznam*:
   - Jediná platná volba zavření pro rozdělovník je CONONE; volání selže s kódem příčiny RC2046 nebo RC2045 , pokud jsou uvedeny jiné volby.

- Když je rozdělovník uzavřen, individuální kódy dokončení a kódy příčiny nejsou vráceny pro fronty v seznamu-pro diagnostické účely jsou k dispozici pouze parametry **CMPCOD** a **REASON** volání.

  Dojde-li k selhání při zavírání jedné z front, bude správce front pokračovat ve zpracování a pokusí se zavřít zbývající fronty v distribučním seznamu. Parametry **CMPCOD** a **REASON** volání jsou poté nastaveny tak, aby vracely informace popisující selhání. Proto je možné, aby kód dokončení byl CCFAIL, i když většina front byla úspěšně uzavřena. Fronta, která zjistila chybu, není identifikována.

  Pokud dojde k selhání ve více než jedné frontě, není definováno, které selhání je ohlášeno v parametrech **CMPCOD** a **REASON** .

Parametry

Volání MQCLOSE má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOBJ (10místné celé číslo se znaménkem)-vstup/výstup

Popisovač objektu.

Tento popisovač představuje objekt, který se zavírá. Objekt může být libovolného typu. Hodnota *HOBJ* byla vrácena předchozím voláním MQOPEN.

Při úspěšném dokončení volání správce front nastaví tento parametr na hodnotu, která není platným popisovačem pro dané prostředí. Tato hodnota je:

HOUNUH
Nepoužitelný popisovač objektu.

OPTS (10místné celé číslo se znaménkem)-vstup

Volby, které řídí akci MQCLOSE.

Parametr **OPTS** řídí způsob zavření objektu. Více než jedním způsobem lze zavřít pouze trvalé dynamické fronty a odběry. Trvalé dynamické fronty lze buď zachovat, nebo odstranit; jedná se o fronty s atributem **DefinitionType** , který má hodnotu QDPERM (viz atribut **DefinitionType** popsaný v části “Atributy pro fronty” na stránce 1360). Volby zavření jsou shrnuty v tabulce v tomto tématu.

Trvalé odběry lze buď zachovat, nebo odebrat. Tyto odběry jsou vytvořeny pomocí volání MQSUB s volbou SODUR.

Při zavírání manipulátoru do spravovaného místa určení (což je parametr **Hobj** vrácený při volání MQSUB, které použilo volbu SOMAN) správce front vyčistí všechna nenačtená publikování po odebrání také přidruženého odběru. To se provádí pomocí volby CORMSB v parametru **Hsub** vráceném ve volání MQSUB. Všimněte si, že CORMSB je výchozí chování v MQCLOSE pro netrvalý odběr.

Při zavírání manipulátoru do nespravovaného místa určení odpovídáte za vyčištění fronty, kam jsou odesílána publikování. Doporučuje se nejprve zavřít odběr pomocí CORMSB a poté zpracovat zprávy mimo frontu, dokud nezbývají žádné zprávy.

Musí být uveden jeden (a pouze jeden) z následujících:

Volby uzavření dynamické fronty

Tyto volby řídí způsob zavírání trvalých dynamických front:

CODEL
Odstraňte frontu.

Fronta se odstraní, pokud je splněna některá z následujících podmínek:

- Jedná se o trvalou dynamickou frontu vytvořenou předchozím voláním MQOPEN a ve frontě nejsou žádné zprávy a neexistují žádné nepotvrzené nevyřízené požadavky get nebo put pro frontu (buď pro aktuální úlohu, nebo pro jinou úlohu).
- Jedná se o dočasnou dynamickou frontu vytvořenou voláním MQOPEN, které vrátilo hodnotu *HOBJ*. V tomto případě jsou všechny zprávy ve frontě vymazány.

Ve všech ostatních případech, včetně případu, kdy byla funkce *Hobj* vrácena při volání MQSUB, volání selže s kódem příčiny RC2045a objekt nebude odstraněn.

COPURG-UVOLNĚNÍ
Odstraňte frontu a vymažte v ní všechny zprávy.

Fronta se odstraní, pokud je splněna některá z následujících podmínek:

- Jedná se o trvalou dynamickou frontu vytvořenou předchozím voláním MQOPEN a pro tuto frontu neexistují žádné nepotvrzené neprovedené požadavky get nebo put (buď pro aktuální úlohu, nebo pro jinou úlohu).
- Jedná se o dočasnou dynamickou frontu vytvořenou voláním MQOPEN, které vrátilo hodnotu *HOBJ*.

Ve všech ostatních případech, včetně případu, kdy byla funkce *Hobj* vrácena při volání MQSUB, volání selže s kódem příčiny RC2045a objekt nebude odstraněn.

Další tabulka zobrazuje, které volby zavření jsou platné a zda je objekt zachován nebo odstraněn.

Tabulka 743. Platné volby zavření pro použití se zachovanými nebo odstraněnými objekty

Typ objektu nebo fronty	**CONONE**	**CODEL**	**COPURG-UVOLNĚNÍ**
Objekt jiný než fronta	Zachováno	Neplatné	Neplatné
Předdefinovaná fronta	Zachováno	Neplatné	Neplatné
permanentní dynamická fronta	Zachováno	Odstraněno, pokud jsou prázdné a žádné nevyřízené aktualizace	Zprávy odstraněny; fronta odstraněna, pokud nejsou žádné nevyřízené aktualizace
Dočasná dynamická fronta (volání vydané tvůrcem fronty)	Odstraněno	Odstraněno	Odstraněno
Dočasná dynamická fronta (volání nebylo vydáno tvůrcem fronty)	Zachováno	Neplatné	Neplatné
Distribuční seznam	Zachováno	Neplatné	Neplatné
Cíl spravovaného odběru	Zachováno	Neplatné	Neplatné
Distribuční seznam (odběr byl odebrán)	Zprávy odstraněny; fronta odstraněna	Neplatné	Neplatné

Volby uzavření odběru

Tyto volby určují, zda mají být při zavření manipulátoru odebrány trvalé odběry a zda mají být vyčištěna publikování čekající na čtení aplikací. Tyto volby jsou platné pouze pro použití s popisovačem objektu vráceným v parametru **HSUB** volání MQSUB.

COKPSB
Popisovač pro odběr je uzavřen, ale provedený odběr je zachován. Publikování budou nadále odesílána do místa určení určeného v odběru. Tato volba je platná pouze v případě, že byl odběr proveden s volbou SODUR. COKPSB je výchozí, pokud je odběr trvalý

CORMSB
Odběr je odebrán a popisovač odběru je uzavřen.

Parametr **Hobj** volání MQSUB není zneplatněn uzavřením parametru **Hsub** a může být nadále používán pro MQGET nebo MQCB pro příjem zbývajících publikování. Je-li parametr **Hobj** volání MQSUB také zavřen, budou v případě, že se jednalo o spravované místo určení, odebrána všechna nenačtená publikování.

CORMSB je výchozí, pokud je odběr netrvalý.

Tyto volby uzavření odběru jsou shrnuty v následujících tabulkách:

Chcete-li zavřít popisovač trvalého odběru, ale opustit odběr, použijte následující volby uzavření odběru:

Tabulka 744. Volby úlohy pro uzavření popisovače trvalého odběru a opuštění předplatného

Úloha	**Volba uzavření odběru**
Zachovat publikování na popisovači MQOPENed	COKPSB
Odebrat publikování na popisovači MQOPENed	Akce není povolena
Udržujte publikace na rukojeti s SOMAN	COKPSB
Odstranění publikací na rukojeti pomocí systému SOMAN	Akce není povolena

Chcete-li zrušit odběr, buď zavřením popisovače trvalého odběru a jeho zrušením, nebo zavřením popisovače netrvalého odběru, použijte následující volby uzavření odběru:

Tabulka 745. Volby úlohy pro zrušení odběru

Úloha	**Volba uzavření odběru**
Zachovat publikování na popisovači MQOPENed	CORMSB
Odebrat publikování na popisovači MQOPENed	Akce není povolena
Udržujte publikace na rukojeti s SOMAN	CORMSB
Odstranění publikací na rukojeti pomocí systému SOMAN	COPGSB

Volby dopředného čtení

Následující volby řídí, co se stane s dočasnými zprávami, které byly odeslány klientovi před tím, než je aplikace vyžádala a dosud nebyly aplikací spotřebovány. Tyto zprávy jsou uloženy ve vyrovnávací paměti dopředného čtení klienta, která čeká na vyžádání aplikací, a mohou být buď vyřazeny, nebo spotřebovány z fronty před dokončením operace MQCLOSE.

COIMM
Objekt je okamžitě uzavřen a všechny zprávy, které byly klientovi odeslány před tím, než je aplikace požadovala, jsou vyřazeny a nejsou k dispozici pro využití žádnou aplikací. Toto je výchozí hodnota.

COQSC
Je učiněn požadavek na zavření objektu, ale pokud se nějaké zprávy, které byly odeslány klientovi před tím, než je aplikace požadovala, stále nacházejí ve vyrovnávací paměti dopředného čtení klienta, volání MQCLOSE se vrátí s varovným kódem RC2458a popisovač objektu zůstane platný.

Aplikace pak může pokračovat v používání popisovače objektu k načtení zpráv, dokud nebudou k dispozici žádné další, a poté objekt znovu nezavře. Klientovi se neodešlou žádné další zprávy před aplikací, která by požadovala, a čtení napřed se nyní vypne.

Aplikacím se doporučuje používat COQSC místo toho, aby se pokusily dosáhnout bodu, ve kterém již nejsou žádné další zprávy ve vyrovnávací paměti pro dopředné čtení klienta, protože mezi

posledním voláním MQGET a následujícím voláním MQCLOSE by mohla být přijata zpráva, která by byla vyřazena, pokud by byl použit modul COIMM.

Pokud je příkaz MQCLOSE s COQSC vydán z funkce asynchronního zpětného volání, použije se stejné chování při čtení dopředných zpráv. Je-li vrácen varovný kód RC2458 , bude funkce zpětného volání volána alespoň jednou. Když byla poslední zbývající zpráva, která byla přečtena napřed, předána do funkce zpětného volání, pole CBCFLG je nastaveno na CBCFBE.

Výchozí volba

Pokud nevyžadujete žádnou z dříve popsaných voleb, můžete použít následující volbu:

CONONE
Není vyžadováno žádné volitelné zpracování zavření.

Toto musí být uvedeno pro:

- Jiné objekty než fronty
- Předdefinované fronty
- Dočasné dynamické fronty (ale pouze v těch případech, kdy *HOBJ* není manipulátor vrácený voláním MQOPEN, které vytvořilo frontu).
- Distribuční seznamy

Ve všech předchozích případech je objekt zachován a není odstraněn.

Je-li tato volba zadána pro dočasnou dynamickou frontu:

- Fronta je odstraněna, pokud byla vytvořena voláním MQOPEN, které vrátilo hodnotu *HOBJ* ; všechny zprávy, které jsou ve frontě, jsou vymazány.
- Ve všech ostatních případech se fronta (a všechny zprávy na ní) uchovávají.

Je-li tato volba uvedena pro trvalou dynamickou frontu, fronta se zachová a neodstraní.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCWARN:

RC2241
(2241, X'8C1') Skupina zpráv není dokončena.

RC2242
(2242, X'8C2') Logická zpráva není dokončena.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2019
(2019, X'7E3') popisovač objektu není platný.

RC2035
(2035, X'7F3') Není autorizováno pro přístup.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2045
(2045, X'7FD') Volba není platná pro typ objektu.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2055
(2055, X'807 ') Fronta obsahuje jednu nebo více zpráv nebo nepotvrzených požadavků na vložení nebo získání.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2063
(2063, X'80F') Došlo k chybě zabezpečení.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQCLOSE(HCONN : HOBJ : OPTS :
C                                       CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQCLOSE          PR                  EXTPROC('MQCLOSE')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Object handle
D HOBJ                          10I 0
D* Options that control the action of MQCLOSE
D OPTS                          10I 0 VALUE
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CMPCOD
D REASON                        10I 0
```

IBM i MQCMIT (Potvrdit změny) na IBM i

Volání MQCMIT informuje správce front o tom, že aplikace dosáhla synchronizačního bodu a že všechny operace získání a vložení zpráv, které se vyskytly od posledního synchronizačního bodu, mají být trvalé. Zprávy vkládané jako součást pracovní jednotky jsou zpřístupněny jiným aplikacím; zprávy načtené jako součást pracovní jednotky jsou odstraněny.

Syntaxe

MQCMIT *(HCONN, COMCOD, REASON)*

Poznámky k použití

Při používání produktu MQCMIT zvažte tyto poznámky k použití.

1. Toto volání lze použít pouze v případě, že správce front sám koordinuje pracovní jednotku. Jedná se o lokální pracovní jednotku, kde změny ovlivňují pouze prostředky IBM MQ .
2. V prostředích, kde správce front nekoordinuje pracovní jednotku, musí být místo MQCMIT použito příslušné volání potvrzení. Prostředí může také podporovat implicitní potvrzení způsobené normálním ukončením aplikace.
   - V systému IBM ilze toto volání použít pro lokální pracovní jednotky koordinované správcem front. To znamená, že na úrovni úlohy nesmí existovat definice vázaného zpracování, to znamená, že pro úlohu nesmí být vydán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** .
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, odebrání těchto změn závisí na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v části "MQDISC (Odpojit správce front) na systému IBM i" na stránce 1283 .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, správce front uchovává informace týkající se skupiny zpráv a logické zprávy pro poslední úspěšná volání MQPUT a MQGET. Tyto informace jsou přidruženy k popisovači fronty a zahrnují například:
   - Hodnoty polí *MDGID*, *MDSEQ*, *MDOFF*a *MDMFL* v deskriptoru MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo dočasná.

   Po potvrzení transakce správce front zachová informace o skupině a segmentu a aplikace může pokračovat v vkládání nebo získávání zpráv do aktuální skupiny zpráv nebo logické zprávy.

   Uchování informací o skupině a segmentu při potvrzení transakce umožňuje aplikaci rozložit velkou skupinu zpráv nebo velkou logickou zprávu sestávající z mnoha segmentů do několika pracovních jednotek. Použití několika pracovních jednotek může být výhodné, pokud má lokální správce front pouze omezené úložiště front. Aplikace však musí udržovat dostatečné informace, aby mohla znovu spustit vkládání nebo získávání zpráv ve správném bodě, dojde-li k selhání systému. Podrobnosti o tom, jak provést restart ve správném bodě po selhání systému, viz volba PMLOGO popsaná v části "MQPMO (Vložit-volby zpráv) na IBM i" na stránce 1168a volba GMLOGO popsaná v části "MQGMO (Volby Get-message) na systému IBM i" na stránce 1073.

Zbývající poznámky k použití platí pouze v případě, že správce front koordinuje pracovní jednotky:

1. Jednotka práce má stejný rozsah jako manipulátor připojení. To znamená, že všechna volání IBM MQ , která ovlivňují konkrétní pracovní jednotku, musí být provedena pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného manipulátoru připojení (například volání vydaná jinou aplikací)

ovlivňují jinou jednotku práce. Informace o rozsahu popisovačů připojení viz parametr **HCONN** popsaný v MQCONN.

2. Toto volání ovlivní pouze zprávy, které byly vloženy nebo načteny jako součást aktuální pracovní jednotky.
3. Přerušitelná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale která nikdy nevydá potvrzení nebo zpětné volání, může způsobit, že se fronty naplní zprávami, které nejsou k dispozici jiným aplikacím. V zájmu ochrany před touto možností by měl administrátor nastavit atribut správce front **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránil tomu, že by nepoužívané aplikace zaplňovaly fronty, ale dostatečně vysoká, aby umožňovala, aby očekávané aplikace systému zpráv správně fungovaly.

Parametry

Volání MQCMIT má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

COMCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *COMCOD*.

Pokud je *COMCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *COMCOD* hodnotu CCWARN:

RC2003
(2003, X'7D3') Jednotka práce byla vrácena zpět.

RC2124
(2124, X'84C') Výsledek operace potvrzení je nevyřízený.

Má-li parametr *COMCOD* hodnotu CCFAIL:

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2123
(2123, X'84B') Výsledek operace potvrzení nebo vrácení je smíšený.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                   CALLP     MQCMIT(HCONN : COMCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQCMIT           PR                  EXTPROC('MQCMIT')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Completion code
D COMCOD                        10I 0
D* Reason code qualifying COMCOD
D REASON                        10I 0
```

IBM i MQCONN (Připojit správce front) na systému IBM i

Volání MQCONN připojí aplikační program ke správci front. Poskytuje manipulátor připojení správce front, který aplikace používá při následných voláních fronty zpráv.

- Aplikace musí používat volání MQCONN nebo MQCONNX pro připojení ke správci front a volání MQDISC pro odpojení od správce front.

V systému IBM MQ for Multiplatformsse každý podproces v aplikaci může připojit k různým správcům front. V jiných systémech musí být všechna souběžná připojení v rámci procesu ke stejnému správci front.

- "Syntaxe" na stránce 1270
- "Poznámky k použití" na stránce 1270
- "Parametry" na stránce 1271
- "Prohlášení o RPG" na stránce 1273

Syntaxe

MQCONN *(QMNAME, HCONN, CMPCOD, REASON)*

Poznámky k použití

1. Správce front, ke kterému je vytvořeno připojení pomocí volání MQCONN, se nazývá *lokální správce front*.
2. Fronty vlastněné lokálním správcem front se zobrazují aplikaci jako lokální fronty. Je možné vkládat zprávy a získávat zprávy z těchto front.

   Sdílené fronty vlastněné skupinou sdílení front, do které patří lokální správce front, se v aplikaci zobrazují jako lokální fronty. Je možné vkládat zprávy a získávat zprávy z těchto front.

Fronty vlastněné vzdálenými správci front se zobrazují jako vzdálené fronty. Je možné vkládat zprávy do těchto front, ale není možné získat zprávy z těchto front.

3. Pokud dojde k selhání správce front v době, kdy je aplikace spuštěna, musí aplikace znovu spustit volání MQCONN, aby získala nový manipulátor připojení pro použití v následných voláních IBM MQ . Aplikace může pravidelně provádět volání MQCONN, dokud nebude volání úspěšné.

   Pokud si aplikace není jistá, zda je připojena ke správci front, může bezpečně spustit volání MQCONN, aby získala manipulátor připojení. Pokud je aplikace již připojena, vrácený popisovač je stejný jako popisovač vrácený předchozím voláním MQCONN, ale s kódem dokončení CCWARN a kódem příčiny RC2002.
4. Po dokončení aplikace používající volání IBM MQ by aplikace měla k odpojení od správce front použít volání MQDISC.
5. V systému IBM inejsou programy, které skončí abnormálně, automaticky odpojeny od správce front. Aplikace by proto měly být napsány tak, aby umožňovaly volání MQCONN nebo MQCONNX vracející kód dokončení CCWARN a kód příčiny RC2002. Manipulátor připojení vrácený v této situaci lze použít jako normální.

Parametry

Volání MQCONN má následující parametry:

QMNAME (48bajtový znakový řetězec)-vstup

Název správce front.

Jedná se o název správce front, ke kterému se chce aplikace připojit. Název může obsahovat následující znaky:

- Velká písmena abecedy (A až Z)
- Malá písmena abecedy (a až z)
- Číselné číslice (0 až 9)
- Tečka (.), dopředné lomítko (/), podtržítko (_), procento (%)

Název nesmí obsahovat úvodní nebo vložené mezery, ale může obsahovat koncové mezery. Znak null lze použít k označení konce důležitých dat v názvu; hodnota null a všechny následující znaky jsou považovány za mezery. V označených prostředích platí následující omezení:

- V systému IBM imusí být názvy obsahující malá písmena, dopředné lomítko nebo procenta uzavřeny v uvozovkách, pokud jsou zadány v příkazech. Tyto uvozovky nesmí být uvedeny v parametru **QMNAME** .

Pokud se název skládá zcela z mezer, použije se název *výchozího* správce front.

Název zadaný pro *QMNAME* musí být názvem *připojitelného* správce front.

Skupiny sdílení front:V systémech, kde existuje několik správců front a jsou konfigurováni tak, aby tvořili skupinu sdílení front, lze název skupiny sdílení front zadat pro produkt *QMNAME* namísto názvu správce front. To umožňuje aplikaci připojit se k *libovolnému* správci front, který je k dispozici ve skupině sdílení front. Systém lze také nakonfigurovat tak, že prázdný soubor *QMNAME* způsobí připojení ke skupině sdílení front namísto k výchozímu správci front.

Pokud parametr *QMNAME* určuje název skupiny sdílení front, ale v systému existuje také správce front s tímto názvem, vytvoří se připojení k posledně jmenované skupině přednostně k prvnímu. Připojení k jednomu ze správců front ve skupině sdílení front, o které jste se pokusili, je pouze v případě, že se připojení nezdaří.

Je-li připojení úspěšné, lze manipulátor vrácený voláním MQCONN nebo MQCONNX použít pro přístup k *všem* prostředkům (sdíleným i nesdíleným), které patří ke konkrétnímu správci front, ke kterému bylo vytvořeno připojení. Přístup k těmto prostředkům podléhá typickým ovládacím prvkům autorizace.

Pokud aplikace zadá dvě volání MQCONN nebo MQCONNX za účelem vytvoření souběžných připojení a jedno nebo obě volání určují název skupiny sdílení front, druhé volání může vrátit kód dokončení CCWARN a kód příčiny RC2002. K tomu dochází, když se druhé volání připojí ke stejnému správci front jako první volání.

Skupiny sdílení front jsou podporovány pouze v systému z/OS. Připojení ke skupině sdílení front je podporováno pouze v dávkovém prostředí, v dávkovém prostředí RRS a v prostředí TSO.

IBM MQ klientské aplikace: V případě aplikací IBM MQ MQI client se provádí pokus o připojení pro každou definici kanálu připojení klienta s určeným názvem správce front, dokud není úspěšná. Správce front však musí mít stejný název jako zadaný název. Je-li zadán prázdný název, bude každý kanál připojení klienta s prázdným názvem správce front zkoušen až do úspěšného názvu; v tomto případě nebude provedena žádná kontrola skutečného názvu správce front.

IBM MQ Skupiny správců front klienta: Pokud uvedený název začíná hvězdičkou (*), skutečný správce front, ke kterému je vytvořeno připojení, může mít název, který se liší od názvu určeného aplikací. Zadaný název (bez hvězdičky) definuje *skupinu* správců front, kteří jsou vhodní pro připojení. Implementace vybere jednu ze skupiny tak, že postupně vyzkouší každou z nich v abecedním pořadí, dokud není nalezena jedna, ke které lze vytvořit připojení. Pokud není žádný ze správců front ve skupině k dispozici pro připojení, volání se nezdaří. Každý správce front je vyzkoušen pouze jednou. Je-li pro název zadána pouze hvězdička, bude použita výchozí skupina správců front definovaná implementací.

Skupiny správců front jsou podporovány pouze pro aplikace spuštěné v prostředí MQ-client; volání se nezdaří, pokud neklientská aplikace určuje název správce front začínající hvězdičkou. Skupina je definována zadáním několika definic kanálů připojení klienta se stejným názvem správce front (zadaný název bez hvězdičky) pro komunikaci s jednotlivými správci front ve skupině. Výchozí skupina je definována zadáním jedné nebo více definic kanálů připojení klienta, každá s prázdným názvem správce front (zadání prázdného názvu má proto stejný účinek jako zadání jediné hvězdičky pro název klientské aplikace).

Po připojení k jednomu správci front ve skupině může aplikace v polích s názvem správce front ve zprávě a v deskriptorech objektů zadat mezery typickým způsobem, čímž se rozumí název správce front, k němuž se aplikace skutečně připojila (*lokální správce front*). Pokud aplikace potřebuje znát tento název, lze zadat volání MQINQ s dotazem na atribut správce front **QMgrName** .

Předpona hvězdičky k názvu připojení znamená, že aplikace není závislá na připojení ke konkrétnímu správci front ve skupině. Vhodné aplikace by byly:

- Aplikace, které vkládají zprávy, ale nezískají zprávy.
- Aplikace, které vkládají zprávy požadavků a pak získají zprávy odpovědí z *dočasné dynamické* fronty.

Nevhodné aplikace by byly ty, které potřebují získat zprávy z konkrétní fronty v konkrétním správci front; tyto aplikace by neměly před název přidávat hvězdičku.

Všimněte si, že pokud je uvedena hvězdička, maximální délka zbytku názvu je 47 znaků.

Délka tohoto parametru je dána hodnotou LNQMN.

HCONN (10místné celé číslo se znaménkem)-výstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Musí být uveden ve všech následných voláních fronty zpráv vydaných aplikací. Přestane být platný, když je vydáno volání MQDISC nebo když je ukončena jednotka zpracování, která definuje rozsah popisovače.

Rozsah rukojeti je omezen na nejmenší jednotku paralelní zpracování podporované platformou, na které je aplikace spuštěna; manipulátor není platný mimo jednotku paralelního zpracování, ze které bylo vydáno volání MQCONN.

- V systému IBM ije rozsahem manipulátoru úloha vydávající volání.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCWARN:

RC2002
(2002, X'7D2') Aplikace je již připojena.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2267
(2267, X'8DB') Nelze načíst uživatelskou proceduru pracovní zátěže klastru.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2035
(2035, X'7F3') Není autorizováno pro přístup.

RC2137
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2161
(2161, X'871 ') Správce front je uveden do klidového stavu.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2063
(2063, X'80F') Došlo k chybě zabezpečení.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
     C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
```

```
     C                     CALLP     MQCONN(QMNAME : HCONN : CMPCOD :
     C                                      REASON)
```

Definice prototypu pro volání je:

```
     D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     DMQCONN           PR                  EXTPROC('MQCONN')
     D* Name of queue manager
     D QMNAME                        48A
     D* Connection handle
     D HCONN                         10I 0
     D* Completion code
     D CMPCOD                        10I 0
     D* Reason code qualifying CMPCOD
     D REASON                        10I 0
```

IBM i MQCONNX (správce front připojení (rozšířený)) na systému IBM i

Volání MQCONNX připojí aplikační program ke správci front. Poskytuje manipulátor připojení správce front, který aplikace používá při následných voláních IBM MQ .

Volání MQCONNX je podobné volání MQCONN s tím rozdílem, že MQCONNX umožňuje zadat volby pro řízení způsobu, jakým volání funguje.

V systému IBM MQ for Multiplatformsse každý podproces v aplikaci může připojit k různým správcům front. V jiných systémech musí být všechna souběžná připojení v rámci procesu ke stejnému správci front.

- “Syntaxe” na stránce 1274
- “Parametry” na stránce 1274
- “Prohlášení o RPG” na stránce 1275

Syntaxe

MQCONNX *(QMNAME, CNOPT, HCONN, CMPCOD, REASON)*

Parametry

Volání MQCONNX má následující parametry:

QMNAME (48bajtový znakový řetězec)-vstup

Název správce front.

Podrobnosti viz parametr **QMNAME** popsaný v části “MQCONN (Připojit správce front) na systému IBM i” na stránce 1270 .

CNOPT (MQCNO)-vstupní/výstupní

Volby, které řídí akci MQCONNX.

Podrobnosti viz “MQCNO (Volby připojení) na systému IBM i” na stránce 1044.

HCONN (10místné celé číslo se znaménkem)-výstup

Manipulátor připojení.

Podrobnosti viz parametr **HCONN** popsaný v části “MQCONN (Připojit správce front) na systému IBM i” na stránce 1270 .

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Podrobnosti viz parametr **CMPCOD** popsaný v části “MQCONN (Připojit správce front) na systému IBM i” na stránce 1270 .

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Podrobnosti o možných kódech příčiny viz parametr **REASON** popsaný v části “MQCONN (Připojit správce front) na systému IBM i” na stránce 1270 .

Volání MQCONNX může vrátit následující další kódy příčiny:

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2278
(2278, X'8E6') Pole připojení klienta nejsou platná.

RC2139
(2139, X'85B') Struktura voleb připojení je neplatná.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQCONN(QMNAME : HCONN : CMPCOD :
C                                      REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQCONN           PR                  EXTPROC('MQCONN')
D* Name of queue manager
D QMNAME                         48A
D* Options that control the action of MQCONNX
D HCONN                         224A
D* Connection handle
D HCONN                          10I 0
D* Completion code
D CMPCOD                         10I 0
D* Reason code qualifying CMPCOD
D REASON                         10I 0
```

IBM i MQCRTMH (Vytvoření popisovače zprávy) na IBM i

Volání MQCRTMH vrací popisovač zprávy.

Aplikace jej může použít při následných voláních řazení zpráv do fronty:

- Pomocí volání MQSETMP nastavte vlastnost manipulátoru zprávy.
- Pomocí volání MQINQMP můžete zjistit hodnotu vlastnosti manipulátoru zprávy.
- Volání MQDLTMP slouží k odstranění vlastnosti manipulátoru zprávy.

Manipulátor zprávy lze použít pro volání MQPUT a MQPUT1 k přidružení vlastností manipulátoru zprávy k vlastnostem vkládané zprávy. Podobně lze při zadání manipulátoru zprávy ve volání MQGET přistupovat k vlastnostem načítané zprávy pomocí manipulátoru zprávy po dokončení volání MQGET.

K odstranění manipulátoru zprávy použijte příkaz MQDLTMH .

Syntaxe

MQCRTMH *(Hconn, CrtMsgHOpts, Hmsg, CompCode, Reason)*

Parametry

Volání MQCRTMH má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX. Pokud připojení ke správci front přestane být platné a na manipulátoru zprávy nepracuje žádné volání IBM MQ , bude pro odstranění zprávy implicitně voláno volání MQDLTMH .

Případně můžete zadat následující hodnotu:

HCUNAS
Manipulátor připojení nepředstavuje připojení k žádnému konkrétnímu správci front.

Při použití této hodnoty musí být manipulátor zprávy odstraněn s explicitním voláním funkce MQDLTMH , aby se uvolnila paměť, která je mu přidělena; IBM MQ manipulátor zprávy nikdy implicitně neodstraní.

Musí existovat alespoň jedno platné připojení ke správci front vytvořenému v podprocesu, který vytváří popisovač zprávy, jinak volání selže s volbou RC2018.

CRTOPT (MQCMHO)-vstup

Volby, které řídí akci MQCRTMH. Podrobnosti viz MQCMHO .

HMSG (20místné celé číslo se znaménkem)-výstup

Na výstupu je vrácen manipulátor zprávy, který lze použít k nastavení, dotazování a odstranění vlastností manipulátoru zprávy. Na počátku popisovač zprávy neobsahuje žádné vlastnosti.

Popisovač zprávy má také přidružený deskriptor zprávy. Na počátku tento deskriptor zprávy obsahuje výchozí hodnoty. Hodnoty přidružených polí deskriptoru zpráv lze nastavit a zjišťovat pomocí volání MQSETMP a MQINQMP. Volání MQDLTMP resetuje pole deskriptoru zprávy zpět na výchozí hodnotu.

Je-li parametr *HCONN* uveden jako hodnota HCUNAS, pak lze vrácený popisovač zprávy použít pro volání MQGET, MQPUT nebo MQPUT1 s jakýmkoli připojením v rámci jednotky zpracování, ale může být používán pouze jedním voláním IBM MQ v daném okamžiku. Pokud se manipulátor používá, když se druhé volání IBM MQ pokusí použít stejný manipulátor zprávy, druhé volání IBM MQ selže s kódem příčiny RC2499.

Pokud parametr *HCONN* není HCUNAS, pak lze vrácený ovladač zprávy použít pouze na uvedeném připojení.

Stejná hodnota parametru *HCONN* musí být použita pro následná volání MQI, kde je použit tento popisovač zprávy:

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMHBUF
- MQBUFMH

Vrácený popisovač zprávy přestane být platný, když je pro popisovač zprávy vydáno volání MQDLTMH, nebo když je ukončena jednotka zpracování, která definuje rozsah popisovače. Funkce MQDLTMH je volána implicitně, pokud je při vytvoření manipulátoru zprávy zadáno specifické připojení a připojení ke správci front přestane být platné, například pokud je volán modul MQDBC.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

RC2219
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

RC2461
(2461, X'099D') Struktura voleb popisovače zprávy není platná.

RC2273
(2273, X'7D9') Připojení ke správci front bylo ztraceno.

RC2017
(2017, X'07E1') Nejsou k dispozici žádné další popisovače.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2460
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

RC2046
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Další informace viz část “Návratové kódy pro IBM i (ILE RPG)” na stránce 1418.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQCRTMH(HCONN : CRTOPT : HMSG :
                                        CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQCRTMH          PR                  EXTPROC('MQCRTMH')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Options that control the action of MQCRTMH
D CRTOPT                        12A
D* Message handle
D HMSG                          20I 0
D* Completion code
D CMPCOD                        10I 0
```

```
     D* Reason code qualifying CompCode
     D REASON                        10I 0
```

IBM i MQCTL (Řídicí zpětné volání) na systému IBM i

Volání MQCTL provádí řídicí akce na manipulátorů objektů otevřených pro připojení.

- "Syntaxe" na stránce 1278
- "Poznámky k použití" na stránce 1278
- "Parametry" na stránce 1278
- "Prohlášení o RPG" na stránce 1283

Syntaxe

MQCTL *(Hconn, Operation, ControlOpts, CompCode, Reason)*

Poznámky k použití

1. Rutiny zpětného volání musí zkontrolovat odezvy ze všech služeb, které vyvolávají, a pokud rutina zjistí podmínku, kterou nelze vyřešit, musí zadat příkaz MQCB (CBREG), který zabrání opakovaným voláním rutiny zpětného volání.

Parametry

Volání MQCTL má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

OPERATN (10místné celé číslo se znaménkem)-vstup

Operace, která se zpracovává na zpětném volání definovaném pro uvedený popisovač objektu. Musíte zadat pouze jednu z následujících voleb:

CTLSR

Spustit spotřebu zpráv pro všechny definované funkce spotřebitele zpráv pro určený manipulátor připojení.

Zpětná volání se spouštějí na podprocesu spuštěném systémem, který se liší od všech podprocesů aplikace.

Tato operace poskytuje řízení poskytovaného manipulátoru připojení systému. Jediná volání MQI, která mohou být vydána jiným podprocesem než podprocesem spotřebitele, jsou:

- MQCTL s operací CTLSP
- MQCTL s operací CTLSU
- MQDISC-Provede MQCTL s operací CTLSP před odpojením připojení HConn.

Volání RC2500 je vráceno, pokud je vydáno volání rozhraní API IBM MQ , když je spuštěn manipulátor připojení, a volání nepochází z funkce spotřebitele zpráv.

Dojde-li k selhání připojení, konverzace bude ukončena co nejdříve. Proto je možné, že volání rozhraní API IBM MQ vydané na hlavním podprocesu na chvíli obdrží návratový kód RC2500 následovaný návratovým kódem RC2009 , když se připojení vrátí do zastaveného stavu.

Tuto funkci lze zadat ve funkci spotřebitele. Pro stejné připojení jako rutina zpětného volání je jediným účelem zrušení dříve vydané operace CTLSP.

Tato volba není podporována, pokud je aplikace svázána s knihovnou IBM MQ bez podprocesů.

CTLSW
Spustit spotřebu zpráv pro všechny definované funkce spotřebitele zpráv pro určený manipulátor připojení.

Spotřebitelé zpráv jsou spouštěny ve stejném podprocesu a řízení není vráceno volajícímu modulu MQCTL, dokud:

- Uvolněno pomocí operací MQCTL CTLSP nebo CTLSU, nebo
- Všechny rutiny spotřebitele byly odregistrovány nebo pozastaveny.

Pokud jsou všichni spotřebitelé odregistrováni nebo pozastaveni, je vydána implicitní operace CTLSP.

Tuto volbu nelze použít v rámci rutiny zpětného volání ani pro aktuální manipulátor připojení, ani pro žádný jiný manipulátor připojení. Pokud se o volání pokusíte, vrátí se s RC2012.

Pokud kdykoli během operace CTLSW nejsou registrováni žádní nepozastavení spotřebitelé, volání selže s kódem příčiny RC2446.

Pokud je během operace CTLSW připojení pozastaveno, volání MQCTL vrátí varovný kód příčiny RC2521; připojení zůstane 'spuštěno'.

Aplikace se může rozhodnout vydat CTLSP nebo CTLRE. V této instanci se blokuje operace CTLRE.

Tato volba není podporována v jednom klientovi s podporou podprocesů.

CTLSP
Zastavte spotřebu zpráv a počkejte, až všichni spotřebitelé dokončí své operace před dokončením této volby. Tato operace uvolní manipulátor připojení.

Je-li zadána z rutiny zpětného volání, tato volba se projeví až po ukončení rutiny. Po dokončení rutin spotřebitele pro zprávy, které již byly načteny, a po provedení volání zastavení (je-li požadováno) pro rutiny zpětného volání nejsou volány žádné další rutiny spotřebitele zpráv.

Je-li vydán mimo rutinu zpětného volání, řízení se nevrátí volajícímu, dokud nejsou dokončeny rutiny spotřebitele pro již přečtené zprávy a po provedení volání zastavení (je-li požadováno) pro zpětná volání. Samotná zpětná volání však zůstávají registrována.

Tato funkce nemá žádný vliv na zprávy dopředného čtení. Musíte zajistit, aby spotřebitelé spouštěli příkaz MQCLOSE (COQSC) z funkce zpětného volání, abyste určili, zda jsou k dispozici další zprávy, které mají být doručeny.

CTLSU
Pozastavit spotřebu zpráv. Tato operace uvolní manipulátor připojení.

To nemá vliv na dopředné čtení zpráv pro aplikaci. Chcete-li ukončit spotřebu zpráv po dlouhou dobu, zvažte zavření fronty a její opětovné otevření, když musí spotřeba pokračovat.

Je-li vydán z rutiny zpětného volání, neprojeví se, dokud se rutina neukončí. Po ukončení aktuální rutiny nebudou volány žádné další rutiny spotřebitele zpráv.

Je-li vydán mimo zpětné volání, ovládací prvek se nevrátí volajícímu, dokud není dokončena aktuální rutina spotřebitele a nejsou volána žádná další.

CTLRE
Obnovte spotřebu zpráv.

Tato volba je obvykle vydána z hlavního podprocesu aplikace, ale lze ji také použít z rutiny zpětného volání ke zrušení dřívějšího požadavku na pozastavení vydaného ve stejné rutině.

Pokud se CTLRE použije k obnovení CTLSW, pak se operace zablokuje.

PCTLOP (MQCTLO)-vstup

Volby, které řídí akci MQCTL

Podrobnosti o struktuře viz MQCTLO .

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Následující kódy příčiny jsou ty, které může správce front vrátit pro parametr **Reason** .

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2133
(2133, X'855 ') Nelze načíst moduly služeb převodu dat.

RC2204
(2204, X'89C') Adaptér není k dispozici.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2374
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API.

RC2183
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

RC2157
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

RC2005
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

RC2487
(2487, X'9B7') Nelze volat rutinu zpětného volání

RC2448
(2448, X' 990 ') Nelze zrušit registraci, pozastavit nebo obnovit, protože neexistuje žádné registrované zpětné volání

RC2486
(2486, X'9B6') Buď byly zadány volby CallbackFunction i CallbackName ve volání CBREG, nebo byl zadán jeden z parametrů CallbackFunction nebo CallbackName , ale neodpovídá aktuálně registrované funkci zpětného volání.

RC2483
(2483, X'9B3') Nesprávné pole typu CallBack.

RC2219
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

RC2444
(2444, X'98C') Blok voleb je chybný.

RC2484
(2484, X'9B4') Chybné pole voleb MQCBD.

RC2140
(2140, X'85C') Požadavek na čekání byl odmítnut produktem CICS.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2217
(2217, X'8A9') Není autorizováno pro připojení.

RC2202
(2202, X'89A') Uklidení připojení.

RC2203
(2203, X'89B') Probíhá ukončování připojení.

RC2207
(2207, X'89F') Chyba identifikátoru korelace.

RC2016
(2016, X'7E0') Získá blokováno pro frontu.

RC2351
(2351, X'92F') Globální konflikt pracovních jednotek.

RC2186
(2186, X'88A') Struktura voleb Get-message není platná.

RC2353
(2353, X' 931 ') Rukojeť používaná pro globální jednotku práce.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2019
(2019, X'7E3') popisovač objektu není platný.

RC2259
(2259, X'8D3') Nekonzistentní specifikace procházení.

RC2245
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

RC2246
(2246, X'8C6') Zpráva pod kurzorem není platná pro načtení.

RC2352
(2352, X' 930 ') Globální jednotka práce je v konfliktu s lokální jednotkou práce.

RC2247
(2247, X'8C7') Volby shody nejsou platné.

RC2485
(2485, X'9B5') Chybné pole MaxMsg

RC2026
(2026, X'7EA') Deskriptor zpráv není platný.

RC2497
(2497, X'9C1') Zadaný vstupní bod funkce nebyl v modulu nalezen.

RC2496
(2496, X'9C0') Modul je nalezen, ale je nesprávného typu (32bitový nebo 64bitový) nebo není platný dll.

RC2495
(2495, X'9BF') Modul nebyl nalezen ve vyhledávací cestě nebo není autorizován k načtení.

RC2206
(2206, X'89E') Chyba identifikátoru zprávy.

RC2250
(2250, X'8CA') Pořadové číslo zprávy není platné.

RC2331
(2331, X'91B') Použití tokenu zprávy není platné.

RC2036
(2036, X'7F4') Fronta není otevřena pro procházení.

RC2037
(2037, X'7F5') Fronta není otevřena pro vstup.

RC2041
(2041, X'7F9') Definice objektu se od doby otevření změnila.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2488
(2488, X'9B8') Chybný kód operace ve volání rozhraní API

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2193
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2394
(2394, X'95A') Fronta má chybný typ indexu.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2161
(2161, X'871 ') Správce front je uveden do klidového stavu.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2069
(2069, X'815 ') Pro tuto rukojeť je signál vynikající.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2109
(2109, X'83D') Volání bylo potlačeno uživatelským programem.

RC2072
(2072, X'818 ') Podpora synchronizační funkce není k dispozici.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2354
(2354, X' 932 ') Registrace v globální pracovní jednotce se nezdařila.

RC2355
(2355, X' 933 ') Směs volání jednotky práce není podporována.

RC2255
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

RC2090
(2090, X'82A') Interval čekání v MQGMO není platný.

RC2256
(2256, X'8D0') Dodána nesprávná verze MQGMO.

RC2257
(2257, X'8D1') Dodána nesprávná verze MQMD.

RC2298
(2298, X'8FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQCTL(HCONN : OPERATN : PCTLOP :
                                      CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQCTL            PR                  EXTPROC('MQCTL')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Operation
D OPERATN                       10I 0 VALUE
D* Control options
D PCTLOP                        32A
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CompCode
D REASON                        10I 0
```

IBM i MQDISC (Odpojit správce front) na systému IBM i

Volání MQDISC přeruší připojení mezi správcem front a aplikačním programem a jedná se o inverzní volání MQCONN nebo MQCONNX.

Syntaxe

MQDISC *(HCONN, CMPCOD, REASON)*

Poznámky k použití

1. Pokud je vyvoláno volání MQDISC, když má aplikace stále otevřené objekty, správce front tyto objekty zavře s volbami zavření nastavenými na CONONE.
2. Pokud aplikace skončí s nepotvrzenými změnami v transakci, odebrání těchto změn závisí na tom, jak aplikace skončí:
   a. Pokud aplikace před ukončením vydá volání MQDISC:
      - V případě koordinované pracovní jednotky správce front zadá správce front jménem aplikace volání MQCMIT. Jednotka práce je potvrzena, pokud je to možné, a vrácena zpět, pokud ne.
      - U externě koordinované jednotky práce nedochází ke změně stavu jednotky práce; správce front však bude označovat, že jednotka práce by měla být potvrzena, když bude požádána koordinátorem jednotky práce.
   b. Pokud aplikace skončí normálně, ale bez volání MQDISC, jednotka práce se odvolá.
   c. Pokud aplikace ukončí *abnormálně* bez volání MQDISC, jednotka práce se odvolá.

Parametry

Volání MQDISC má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup/výstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

Při úspěšném dokončení volání správce front nastaví hodnotu *HCONN* na hodnotu, která není platným popisovačem pro dané prostředí. Tato hodnota je:

HCUNUH
Nepoužitelný manipulátor připojení.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                   CALLP     MQDISC(HCONN : CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQDISC           PR                  EXTPROC('MQDISC')
D* Connection handle
D HCONN                         10I 0
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CMPCOD
D REASON                        10I 0
```

IBM i MQDLTMH (Odstranění popisovače zprávy) na IBM i

Volání MQDLTMH odstraní manipulátor zprávy a je inverzní k volání MQCRTMH.

- "Syntaxe" na stránce 1285
- "Poznámky k použití" na stránce 1285
- "Parametry" na stránce 1286
- "Prohlášení o RPG" na stránce 1288

Syntaxe

MQDLTMH *((Hconn, Hmsg, DltMsgHOpts, CompCode, Reason)*

Poznámky k použití

1. Toto volání lze použít pouze v případě, že správce front sám koordinuje jednotku práce. Může se jednat o:
   - Lokální pracovní jednotka, kde změny ovlivňují pouze prostředky IBM MQ .
   - Globální jednotka práce, kde změny mohou ovlivnit prostředky patřící jiným správcům prostředků a prostředky IBM MQ .

   Další podrobnosti o lokálních a globálních jednotkách práce viz "MQBEGIN (začátek transakce) na IBM i" na stránce 1247.
2. V prostředích, kde správce front nekoordinuje pracovní jednotku, použijte místo MQBACK příslušné zpětné volání. Prostředí může také podporovat implicitní vrácení zpět způsobené nestandardním ukončením aplikace.
   - V systému z/OSpoužijte následující volání:
     - Dávkové programy (včetně IMS dávkových programů DL/I) mohou používat volání MQBACK, pokud pracovní jednotka ovlivňuje pouze prostředky IBM MQ . Pokud však pracovní jednotka ovlivňuje prostředky IBM MQ i prostředky patřící jiným správcům prostředků (například Db2), použijte volání SRRBACK poskytované službou RRS (z/OS Obnovitelný Resource Service). Volání SRRBACK vrací změny prostředků náležejících správcům prostředků, kteří mají povolenou koordinaci RRS.
     - Aplikace systému CICS musí použít příkaz `EXEC CICS SYNCPOINT ROLLBACK` k vrácení pracovní jednotky zpět. Nepoužívejte volání MQBACK pro aplikace CICS .
     - Aplikace IMS (jiné než dávkové programy DL/I) musí používat IMS volání, jako např. `ROLB` , aby se vrátila jednotka práce. Nepoužívejte volání MQBACK pro aplikace IMS (jiné než dávkové programy DL/I).
   - V systému IBM ipoužijte toto volání pro lokální pracovní jednotky koordinované správcem front. To znamená, že na úrovni úlohy nesmí existovat definice vázaného zpracování, to znamená, že pro úlohu nesmí být vydán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** .
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, odebrání těchto změn závisí na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v části "MQDISC (Odpojit správce front) na systému IBM i" na stránce 1283 .

4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, správce front uchovává informace týkající se skupiny zpráv a logické zprávy pro poslední úspěšná volání MQPUT a MQGET. Tyto informace jsou přidruženy k popisovači fronty a zahrnují například:
   - Hodnoty polí *GroupId*, *MsgSeqNumber*, *Offset*a *MsgFlags* v deskriptoru MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo dočasná.

   Správce front uchovává tři sady informací o skupinách a segmentech, jednu sadu pro každou z následujících položek:
   - Poslední úspěšné volání MQPUT (může být součástí pracovní jednotky).
   - Poslední úspěšné volání MQGET, které odebrala zprávu z fronty (může být součástí pracovní jednotky).
   - Poslední úspěšné volání MQGET, které procházelo zprávu ve frontě (nemůže být součástí pracovní jednotky).

   Pokud aplikace vloží nebo získá zprávy jako součást jednotky práce a pak vrátí zpět jednotku práce, informace o skupině a segmentu se obnoví na hodnotu, kterou měla dříve:
   - Informace přidružené k volání MQPUT jsou obnoveny na hodnotu, kterou měly před prvním úspěšným voláním MQPUT pro daný popisovač fronty v aktuální transakci.
   - Informace přidružené k volání MQGET jsou obnoveny na hodnotu, kterou měly před prvním úspěšným voláním MQGET pro daný manipulátor fronty v aktuální transakci.

   Fronty, které byly aktualizovány aplikací po spuštění jednotky práce, ale mimo rozsah jednotky práce, nemají obnoveny informace o skupinách a segmentech, pokud je jednotka práce odvolána.

   Obnova informací o skupině a segmentu na předchozí hodnotu, když je jednotka práce odvolána, umožňuje aplikaci rozložit velkou skupinu zpráv nebo velkou logickou zprávu sestávající z mnoha segmentů do několika jednotek práce a restartovat ji ve správném bodě ve skupině zpráv nebo logické zprávě, pokud jedna z jednotek práce selže. Použití několika pracovních jednotek může být výhodné, pokud má lokální správce front pouze omezené úložiště front. Aplikace však musí udržovat dostatek informací, aby mohla znovu spustit vkládání nebo získávání zpráv na správném místě, pokud dojde k selhání systému.

   Podrobnosti o tom, jak provést restart ve správném bodě po selhání systému, viz volba PMLOGO popsaná v části PMOPT (10místné celé číslo se znaménkem)a volba GMLOGO popsaná v části GMOPT (10místné celé číslo se znaménkem).

   Zbývající poznámky k použití platí pouze v případě, že správce front koordinuje pracovní jednotky:
5. Jednotka práce má stejný rozsah jako manipulátor připojení. Všechna volání IBM MQ , která ovlivňují konkrétní jednotku práce, musí být provedena pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného manipulátoru připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu manipulátorů připojení viz HCONN (10místné celé číslo se znaménkem)-výstup .
6. Toto volání ovlivní pouze zprávy, které byly vloženy nebo načteny jako součást aktuální pracovní jednotky.
7. Přerušitelná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale nikdy nevydá volání potvrzení nebo odvolání, může zaplnit fronty zprávami, které nejsou k dispozici pro jiné aplikace. Chcete-li se vyhnout této možnosti, administrátor musí nastavit atribut správce front **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila tomu, že by běžné aplikace zaplňovaly fronty, ale dostatečně vysoká, aby umožňovala správné fungování očekávaných aplikací systému zpráv.

Parametry

Volání MQDLTMH má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **HMSG** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, pak musí být ustanoveno platné připojení na podprocesu, který odstranil popisovač zprávy, jinak volání selže s RC2009 .

HMSG (20místné celé číslo se znaménkem)-vstup/výstup

Jedná se o popisovač zprávy, který má být odstraněn. Hodnota byla vrácena předchozím voláním MQCRTMH.

Při úspěšném dokončení volání je popisovač nastaven na neplatnou hodnotu pro dané prostředí. Tato hodnota je:

HMUNUH
Nepoužitelný popisovač zprávy.

Popisovač zprávy nelze odstranit, pokud probíhá jiné volání IBM MQ , kterému byl předán stejný popisovač zprávy.

DLTOPT (MQDMHO)-vstup

Podrobnosti viz MQDMHO .

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Je-li hodnota *CMPCOD* CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

RC2219
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

RC2009
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

RC2462
(2462, X'099E') Struktura voleb popisovače zprávy odstranění není platná.

RC2460
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

RC2499
(2499, X'09C3') Obsluha zprávy je již používána.

RC2046
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Další informace viz část "Návratové kódy pro IBM i (ILE RPG)" na stránce 1418.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQDLTMH(HCONN : HMSG : DLTOPT :
                                   CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQDLTMH          PR                  EXTPROC('MQDLTMH')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Message handle
D HMSG                          20I 0
D* Options that control the action of MQDLTMH
D DLTOPT                        12A
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CompCode
D REASON                        10I 0
```

MQDLTMP-Odstranit vlastnost zprávy

Volání MQDLTMP odstraní vlastnost z manipulátoru zprávy a je inverzní k volání MQSETMP.

Syntaxe

MQDLTMP *(Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason)*

Parametry

Volání MQDLTMP má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **HMSG** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, pak musí být ustanoveno platné připojení na podprocesu, který odstranil popisovač zprávy, jinak volání selže s RC2009.

HMSG (20místné celé číslo se znaménkem)-vstup

Jedná se o popisovač zprávy obsahující vlastnost, která má být odstraněna. Hodnota byla vrácena předchozím voláním MQCRTMH.

DLTOPT (MQDMPO)-Vstup

Podrobnosti viz datový typ MQDMPO .

PRNAME (MQCHARV)-vstup

Název vlastnosti, která se má odstranit. Další informace o názvech vlastností viz Názvy vlastností .

V názvu vlastnosti nejsou povoleny zástupné znaky.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCWARN:

RC2471
(2471, X'09A7') Vlastnost není k dispozici.

RC2421
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nelze analyzovat.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'0852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'086D') Primární a domovská ASID se liší.

RC2219
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

RC2009
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

RC2481
(2481, X'09B1') Struktura voleb vlastností zprávy pro odstranění je neplatná.

RC2460
(2460, X'099C') popisovač zprávy není platný.

RC2499
(2499, X'09C3') Obsluha zprávy je již používána.

RC2046
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2442
(2442, X'098A') Neplatný název vlastnosti.

RC2111
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

RC2195
(2195, X'0893 ') Došlo k neočekávané chybě.

Další informace o těchto kódech naleznete v tématu Kódy dokončení a příčiny rozhraní API.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQDLTMP(HCONN : HMSG : DLTOPT :
                                    PRNAME : CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQDLTMP          PR                  EXTPROC('MQDLTMP')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Message handle
D HMSG                          20I 0 VALUE
D* Options that control the action of MQDLTMP
D DLTOPT                        12A
D* Property name
D PRNAME                        32A
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CompCode
D REASON                        10I 0
```

IBM i MQGET (Získat zprávu) na IBM i

Volání MQGET načte zprávu z lokální fronty, která byla otevřena pomocí volání MQOPEN.

Syntaxe

MQGET *(HCONN, HOBJ, MSGDSC, GMO, BUFLEN, BUFFER, DATLEN, CMPCOD, REASON)*

Poznámky k použití

1. Načtená zpráva je obvykle odstraněna z fronty. K tomuto odstranění může dojít jako součást samotného volání MQGET nebo jako součást synchronizačního bodu. K odstranění zprávy nedojde, pokud je v parametru **GMO** uvedena volba GMBRWF nebo GMBRWN (viz pole *GMOPT* popsané v části "MQGMO (Volby Get-message) na systému IBM i" na stránce 1073).
2. Je-li zadána volba GMLK s jednou z voleb procházení, procházená zpráva je uzamčena tak, aby byla viditelná pouze pro tento popisovač.

   Je-li uvedena volba GMUNLK, dříve zamčená zpráva se odemkne. V tomto případě se nenačte žádná zpráva a parametry **MSGDSC**, **BUFLEN**, **BUFFER** a **DATLEN** se nezkontrolují ani nezmění.
3. Pokud je aplikace vydávající volání MQGET spuštěna jako IBM MQ MQI client, je možné, že načtená zpráva bude ztracena v případě, že během zpracování volání MQGET dojde k nestandardnímu ukončení volání IBM MQ MQI client nebo k přerušení připojení klienta. K tomu dochází proto, že náhradník, který je spuštěn na platformě správce front a který zadá volání MQGET jménem klienta, nemůže zjistit ztrátu klienta, dokud náhradník nevrátí zprávu klientovi; k tomu dojde po odebrání zprávy z fronty. Tato situace může nastat jak pro trvalé zprávy, tak pro přechodné zprávy.

   Riziko ztráty zpráv tímto způsobem lze eliminovat vždy načtením zpráv v rámci pracovních jednotek (tj. zadáním volby GMSYP ve volání MQGET a použitím volání MQCMIT nebo MQBACK k potvrzení nebo vrácení pracovní jednotky po dokončení zpracování zprávy). Je-li zadán parametr GMSYP

a klient se ukončí nestandardně nebo je připojení přerušeno, náhradní jednotka práce ve správci front a zpráva je znovu uvedena do fronty.

V zásadě může dojít ke stejné situaci s aplikacemi spuštěnými na platformě správce front, ale v tomto případě je okno, během kterého může být zpráva ztracena, malé. Avšak stejně jako v případě produktu IBM MQ MQI clients lze riziko eliminovat načtením zprávy v rámci pracovní jednotky.

4. Pokud aplikace vloží posloupnost zpráv na konkrétnífronty v rámci jedné pracovní jednotky a poté úspěšně potvrdí tuto pracovní jednotku, budou zprávy k dispozici pro načtení následujícím způsobem:
   - Pokud se jedná o *nesdílenou frontu* (tj. lokální frontu), budou všechny zprávy v rámci pracovní jednotky k dispozici současně.
   - Pokud se jedná o frontu *sdílenou frontu*, budou zprávy v rámci pracovní jednotky k dispozici v pořadí, v jakém byly vloženy, ale ne všechny najednou. Když je systém silně zatížený, je možné, že první zpráva v pracovní jednotce bude úspěšně načtena, ale volání MQGET pro druhou nebo následnou zprávu v pracovní jednotce selže s RC2033. Pokud k tomu dojde, musí aplikace chvíli počkat a poté operaci zopakovat.
5. Pokud aplikace vloží posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, pokud jsou splněny určité podmínky. Podrobnosti naleznete v poznámkách k použití v popisu volání MQPUT. Jsou-li splněny podmínky, jsou zprávy předkládány přijímající žádosti v pořadí, v jakém byly odeslány, pokud:
   - Pouze jeden zásobník získává zprávy z fronty.

     Pokud existují dvě nebo více aplikací, které získávají zprávy z fronty, musí souhlasit s odesílatelem mechanizmu, který má být použit k identifikaci zpráv, které patří do posloupnosti. Odesílatel může například nastavit všechna pole MDCID ve zprávách v posloupnosti na hodnotu, která je pro danou posloupnost zpráv jedinečná.
   - Příjemce záměrně nezmění pořadí načtení, například uvedením konkrétního MDMID nebo MDCID.

   Pokud odesílající aplikace vloží zprávy jako skupinu zpráv, budou zprávy prezentovány přijímající aplikaci ve správném pořadí, pokud přijímající aplikace zadá volbu GMLOGO pro volání MQGET. Další informace o skupinách zpráv viz:
   - Pole MDMFL v deskriptoru MQMD
   - Volba PMLOGO v MQPMO
   - Volba GMLOGO v MQGMO
6. Aplikace testují kód zpětné vazby FBQUIT v poli MDFB parametru **MSGDSC** . Je-li tato hodnota nalezena, aplikace skončí. Další informace viz pole MDFB popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105 .
7. Pokud byla fronta označená HOBJ otevřena s volbou OOSAVA a kód dokončení z volání MQGET je CCOK nebo CCWARN, je kontext přidružený k popisovači fronty HOBJ nastaven na kontext zprávy, která byla načtena (pokud není nastavena volba GMBRWF nebo GMBRWN, v takovém případě je kontext označen jako nedostupný). Tento kontext lze použít při následném volání MQPUT nebo MQPUT1 zadáním voleb PMPASI nebo PMPASA. To umožňuje, aby byl kontext přijaté zprávy zcela nebo částečně přenesen do jiné zprávy (například když je zpráva postoupena do jiné fronty). Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.
8. Pokud je volba GMCONV zahrnuta v parametru **GMO** , data zprávy aplikace se převedou na znázornění požadované přijímající aplikací před umístěním dat do parametru **BUFFER** :
   - Pole MDFMT v řídicích informacích ve zprávě identifikuje strukturu dat aplikace a pole MDCSI a MDENC v řídicích informacích ve zprávě uvádí svůj identifikátor znakové sady a kódování.
   - Aplikace vydávající volání MQGET určuje v polích MDCSI a MDENC v parametru **MSGDSC** identifikátor znakové sady a kódování, do kterého musí být data zprávy aplikace převedena.

   Je-li převod dat zprávy nezbytný, provede převod buď samotný správce front, nebo uživatelská procedura, v závislosti na hodnotě pole MDFMT v řídicích informacích zprávy:
   - Následující formáty jsou automaticky převedeny správcem front; tyto formáty se nazývají "vestavěné" formáty:

FMADMN	FMMDE
FMCICS	Prostředek FMPCF
FMCMD1	FMRMH
FMCMD2	FMRFH
FMDLH	FMRFH2
FMDH	FMSTR
FMEVNT	FMTM
FMIMS	FMXQH
FMIMVS	

- Název formátu FMNONE je speciální hodnota, která označuje, že povaha dat ve zprávě není definována. V důsledku toho se správce front nepokusí o převod, když je zpráva načtena z fronty.

  Poznámka: Je-li ve volání MQGET pro zprávu s názvem formátu FMNONE zadána hodnota GMCONV a znaková sada nebo kódování zprávy se liší od kódování uvedeného v parametru **MSGDSC** , je zpráva stále vrácena v parametru **BUFFER** (nepředpokládá se žádné další chyby), ale volání se dokončí s kódem dokončení CCWARN a kódem příčiny RC2110.

  FMNONE lze použít buď v případě, kdy povaha dat zprávy znamená, že nevyžadují převod, nebo v případě, že se odesílající a přijímající žádosti mezi sebou dohodly na formuláři, ve kterém by měla být data zprávy odeslána.
- Všechny ostatní názvy formátů způsobí, že zpráva bude předána uživatelské proceduře pro převod. Uživatelská procedura má stejný název jako formát, kromě přidání specifických pro prostředí. Názvy formátů zadané uživatelem nesmí začínat písmeny "MQ", protože tyto názvy mohou být v konfliktu s názvy formátů podporovanými v budoucnosti.

Uživatelská data ve zprávě lze převést mezi všemi podporovanými znakovými sadami a kódováním. Mějte však na paměti, že pokud zpráva obsahuje jednu nebo více struktur záhlaví IBM MQ , nelze zprávu převést ze znakové sady nebo na znakovou sadu, která má dvoubajtové nebo vícebajtové znaky pro všechny znaky platné v názvech front. Výsledkem je kód příčiny RC2111 nebo RC2115 , pokud se o to pokusíte, a zpráva se vrátí nepřevedená. Příkladem takové znakové sady je znaková sada Unicode UTF-16 .

Při návratu z příkazu MQGET následující kód příčiny označuje, že zpráva byla úspěšně převedena:

- RCNONE

Následující kód příčiny označuje, že zpráva mohla být úspěšně převedena; aplikace musí zkontrolovat pole `MDCSI` a `MDENC` v parametru **MSGDSC** , aby zjistila:

- RC2079

Všechny ostatní kódy příčiny označují, že zpráva nebyla převedena.

Poznámka: Interpretace kódu příčiny popsaná v tomto příkladu je pravdivá pro konverze prováděné uživatelskými výstupními procedurami pouze v případě, že uživatelská procedura odpovídá pokynům pro zpracování.

9. Pro dříve uvedené vestavěné formáty může správce front při zadání volby GMCONV provést výchozí převod znakových řetězců ve zprávě. Výchozí převod umožňuje správci front při převodu řetězcových dat použít výchozí znakovou sadu určenou instalací, která se blíží skutečné znakové sadě. Výsledkem je, že volání MQGET může být úspěšné s kódem dokončení CCOK, namísto dokončení s kódem příčiny RC2111 nebo RC2115.

   Poznámka: Výsledkem použití přibližné znakové sady pro převod řetězcových dat je, že některé znaky mohou být nesprávně převedeny. Tomu se lze vyhnout tím, že v řetězci budou použity pouze znaky, které jsou společné jak pro skutečnou znakovou sadu, tak pro výchozí znakovou sadu.

   Výchozí převod se vztahuje jak na data zprávy aplikace, tak na znaková pole ve strukturách MQMD a MQMDE:

- Výchozí převod dat zprávy aplikace se provádí pouze v případě, že jsou splněny všechny následující podmínky:
  - Aplikace uvádí GMCONV.
  - Zpráva obsahuje data, která musí být převedena buď ze znakové sady, nebo na znakovou sadu, která není podporována.
  - Při instalaci nebo restartování správce front byl povolen výchozí převod.
- Výchozí převod znakových polí ve strukturách MQMD a MQMDE se provádí podle potřeby, pokud je pro správce front povolen výchozí převod. Převod se provede i v případě, že aplikace ve volání MQGET nezadá volbu GMCONV.

10. Parametr **BUFFER** zobrazený v programovacím příkladu RPG je deklarován jako řetězec; to omezuje maximální délku parametru na 256 bajtů. Je-li vyžadována větší vyrovnávací paměť, musí být parametr deklarován jako struktura nebo jako pole ve fyzickém souboru.

    Deklarování parametru jako struktury zvýší maximální možnou délku na 9999 bajtů, zatímco deklarování parametru jako pole ve fyzickém souboru zvýší maximální možnou délku na přibližně 32 kB.

Parametry

Volání MQGET má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota HCONN byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOBJ (10místné celé číslo se znaménkem)-vstup

Popisovač objektu.

Tento manipulátor představuje frontu, ze které má být načtena zpráva. Hodnota HOBJ byla vrácena předchozím voláním MQOPEN. Fronta musí být otevřena s jednou nebo více z následujících voleb (podrobnosti viz “MQOPEN (Otevřít objekt) na IBM i” na stránce 1314):

- OOINPS
- OOINPX
- OOINPQ
- OOBRW

MSGDSC (MQMD)-vstupní/výstupní

Deskriptor zprávy.

Tato struktura popisuje atributy požadované zprávy a atributy načtené zprávy. Podrobnosti viz “MQMD (deskriptor zprávy) na IBM i” na stránce 1105.

Je-li hodnota BUFLEN menší než délka zprávy, správce front stále zadá MSGDSC , zda je v parametru **GMO** uvedeno GMATM (viz pole GMOPT popsané v tématu “MQGMO (Volby Get-message) na systému IBM i” na stránce 1073).

Pokud aplikace poskytuje version-1 MQMD, má vrácená zpráva předponu MQMDE pro data zprávy aplikace, ale pouze v případě, že jedno nebo více polí v MQMDE má jinou než výchozí hodnotu. Pokud mají všechna pole v prostředí MQMDE výchozí hodnoty, je prostředí MQMDE vynecháno. Název formátu FMMDE v poli MDFMT v MQMD označuje přítomnost MQMDE.

GMO (MQGMO)-vstup/výstup

Volby, které řídí akci MQGET.

Podrobnosti viz “MQGMO (Volby Get-message) na systému IBM i” na stránce 1073.

BUFLEN (10místné celé číslo se znaménkem)-vstup

Délka oblasti BUFFER v bajtech.

Nula může být uvedena pro zprávy, které nemají žádná data, nebo pokud má být zpráva odebrána z fronty a data vyřazena (GMATM musí být uvedeno v tomto případě).

Poznámka: Délka nejdelší zprávy, kterou lze číst z fronty, je dána atributem fronty **MaxMsgLength** ; viz “Atributy pro fronty” na stránce 1360.

BUFFER (1bajtový bitový řetězec x BUFLEN)-výstup

Oblast, která má obsahovat data zprávy.

Vyrovnávací paměť musí být zarovnána na hranici odpovídající povaze dat ve zprávě. 4bajtové zarovnání musí být vhodné pro většinu zpráv (včetně zpráv obsahujících struktury záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8bajtové zarovnání.

Je-li hodnota BUFLEN menší než délka zprávy, přesune se co největší část zprávy do adresáře BUFFER ; to se stane, pokud je GMATM uvedeno v parametru **GMO** (další informace viz pole GMOPT popsané v části “MQGMO (Volby Get-message) na systému IBM i” na stránce 1073).

Znaková sada a kódování dat v souboru **BUFFER** jsou dány poli MDCSI a MDENC vrácenými v parametru **MSGDSC** . Pokud se tyto hodnoty liší od hodnot požadovaných příjemcem, musí příjemce převést data zprávy aplikace na znakovou sadu a požadované kódování. Volbu GMCONV lze použít s uživatelskou procedurou pro provedení konverze dat zprávy (podrobnosti o této volbě viz “MQGMO (Volby Get-message) na systému IBM i” na stránce 1073).

Poznámka: Všechny ostatní parametry volání MQGET jsou ve znakové sadě a kódování lokálního správce front (dané atributem správce front **CodedCharSetId** a ENNAT).

Pokud volání selže, obsah vyrovnávací paměti se mohl ještě změnit.

DATLEN (10místné celé číslo se znaménkem)-výstup

Délka zprávy.

Jedná se o délku dat aplikace ve zprávě v bajtech. Pokud je tato délka zprávy větší než BUFLEN, v parametru **BUFFER** se vrátí pouze BUFLEN bajtů (to znamená, že se zpráva ořízne). Je-li hodnota nula, znamená to, že zpráva neobsahuje žádná data aplikace.

Je-li hodnota BUFLEN menší než délka zprávy, správce front stále zadává hodnotu DATLEN , zda je v parametru **GMO** zadána hodnota GMATM (další informace viz pole GMOPT popsané v tématu “MQGMO (Volby Get-message) na systému IBM i” na stránce 1073). To umožňuje aplikaci určit velikost vyrovnávací paměti požadované pro uložení dat zprávy a poté znovu zadat volání s vyrovnávací pamětí odpovídající velikosti.

Pokud je však zadána volba GMCONV a převedená data zprávy jsou příliš dlouhá na to, aby se vešla do souboru BUFFER, je hodnota vrácená pro parametr DATLEN :

- Délka nepřevedených dat pro formáty definované správcem front.

  V tomto případě, pokud povaha dat způsobí, že se během převodu rozšíří, musí aplikace přidělit vyrovnávací paměť větší, než je hodnota vrácená správcem front pro DATLEN.
- Hodnota vrácená uživatelskou procedurou pro převod dat pro formáty definované aplikací.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující CMPCOD.

Následující kódy příčiny jsou ty, které může správce front vrátit pro parametr **REASON** . Pokud aplikace uvádí volbu GMCONV a je vyvolána uživatelská procedura pro převod některých nebo všech dat zprávy, je to uživatelská procedura, která rozhodne, jaká hodnota se vrátí pro parametr **REASON** . V důsledku toho jsou možné jiné hodnoty než hodnoty zdokumentované později v této sekci.

Pokud je CMPCOD CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr CMPCOD hodnotu CCWARN:

RC2120
(2120, X'848 ') Převedená data jsou příliš velká pro vyrovnávací paměť.

RC2190
(2190, X'88E') Převedený řetězec je pro pole příliš velký.

RC2150
(2150, X'866 ') Řetězec DBCS není platný.

RC2110
(2110, X'83E') Formát zprávy není platný.

RC2243
(2243, X'8C3') Segmenty zpráv mají různé CCSID.

RC2244
(2244, X'8C4') Segmenty zpráv mají odlišné kódování.

RC2209
(2209, X'8A1') Žádná zpráva není uzamčena.

RC2119
(2119, X'847 ') Data zprávy nebyla převedena.

RC2272
(2272, X'8E0') Data zprávy byla částečně převedena.

RC2145
(2145, X'861 ') Parametr zdrojové vyrovnávací paměti není platný.

RC2111
(2111, X'83F') Identifikátor zdrojové kódované znakové sady není platný.

RC2113
(2113, X'841 ') kódování Packed-decimal ve zprávě nebylo rozpoznáno.

RC2114
(2114, X'842 ') kódování s pohyblivou řádovou čárkou ve zprávě nebylo rozpoznáno.

RC2112
(2112, X'840 ') kódování zdrojového celého čísla nebylo rozpoznáno.

RC2143
(2143, X'85F') Parametr délky zdroje není platný.

RC2146
(2146, X'862 ') Parametr cílové vyrovnávací paměti není platný.

RC2115
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

RC2117
(2117, X'845 ') kódování Packed-decimal určené příjemcem nebylo rozpoznáno.

RC2118
(2118, X'846 ') kódování s pohyblivou řádovou čárkou určené příjemcem nebylo rozpoznáno.

RC2116
(2116, X'844 ') kódování cílového celého čísla nebylo rozpoznáno.

RC2079
(2079, X'81F') Vrácena zkrácená zpráva (zpracování dokončeno).

RC2080
(2080, X'820 ') Vrácena zkrácená zpráva (zpracování nebylo dokončeno).

Má-li parametr CMPCOD hodnotu CCFAIL:

RC2004
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

RC2005
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2010
(2010, X'7DA') Parametr délky dat není platný.

RC2016
(2016, X'7E0') Získá blokováno pro frontu.

RC2186
(2186, X'88A') Struktura voleb Get-message není platná.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2019
(2019, X'7E3') popisovač objektu není platný.

RC2241
(2241, X'8C1') Skupina zpráv není dokončena.

RC2242
(2242, X'8C2') Logická zpráva není dokončena.

RC2259
(2259, X'8D3') Nekonzistentní specifikace procházení.

RC2245
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

RC2246
(2246, X'8C6') Zpráva pod kurzorem není platná pro načtení.

RC2247
(2247, X'8C7') Volby shody nejsou platné.

RC2026
(2026, X'7EA') Deskriptor zpráv není platný.

RC2250
(2250, X'8CA') Pořadové číslo zprávy není platné.

RC2033
(2033, X'7F1') Není k dispozici žádná zpráva.

RC2034
(2034, X'7F2') Kurzor procházení není umístěn na zprávě.

RC2036
(2036, X'7F4') Fronta není otevřena pro procházení.

RC2037
(2037, X'7F5') Fronta není otevřena pro vstup.

RC2041
(2041, X'7F9') Definice objektu se od doby otevření změnila.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2161
(2161, X'871 ') Správce front je uveden do klidového stavu.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2024
(2024, X'7E8') V rámci aktuální pracovní jednotky nelze zpracovat žádné další zprávy.

RC2072
(2072, X'818 ') Podpora synchronizační funkce není k dispozici.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2255
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

RC2090
(2090, X'82A') Interval čekání v MQGMO není platný.

RC2256
(2256, X'8D0') Dodána nesprávná verze MQGMO.

RC2257
(2257, X'8D1') Dodána nesprávná verze MQMD.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQGET(HCONN : HOBJ : MSGDSC : GMO :
C                                     BUFLEN : BUFFER : DATLEN :
C                                     CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQGET            PR                  EXTPROC('MQGET')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Object handle
D HOBJ                          10I 0 VALUE
D* Message descriptor
D MSGDSC                       364A
```

```
     D* Options that control the action of MQGET
     D GMO                          112A
     D* Length in bytes of the Buffer area
     D BUFLEN                       10I 0 VALUE
     D* Area to contain the message data
     D BUFFER                         *   VALUE
     D* Length of the message
     D DATLEN                       10I 0
     D* Completion code
     D CMPCOD                       10I 0
     D* Reason code qualifying CMPCOD
     D REASON                       10I 0
```

IBM i MQINQ (dotazovat se na atributy objektu) na IBM i

Volání MQINQ vrací pole celých čísel a sadu znakových řetězců obsahujících atributy objektu.

Následující typy objektů jsou platné:

- Fronta
- Seznam názvů
- Definice procesu
- Správce front

- "Syntaxe" na stránce 1298
- "Poznámky k použití" na stránce 1298
- "Parametry" na stránce 1299
- "Prohlášení o RPG" na stránce 1306

Syntaxe

MQINQ *(HCONN, HOBJ, SELCNT, SELS, IACNT, INTATR, CALEN, CHRATR, CMPCOD, REASON)*

Poznámky k použití

1. Vrácené hodnoty jsou snímkem vybraných atributů. Neexistuje žádná záruka, že atributy nebudou změněny, než aplikace bude moci pracovat na vrácených hodnotách.
2. Když otevřete modelovou frontu, vytvoří se dynamická lokální fronta. To platí i v případě, že otevřete modelovou frontu a dotazujete se na její atributy.

   Atributy dynamické fronty (s určitými výjimkami) jsou stejné jako atributy modelové fronty v době vytvoření dynamické fronty. Pokud poté v této frontě použijete volání MQINQ, správce front vrátí atributy dynamické fronty, nikoli atributy modelové fronty. Podrobnosti o atributech modelové fronty, které jsou zděděny dynamickou frontou, viz Tabulka 1 .
3. Je-li dotazovaným objektem fronta aliasů, hodnoty atributů vrácené voláním MQINQ jsou hodnoty alias fronty a nikoli hodnoty základní fronty, na kterou se alias interpretuje.
4. Pokud je dotazovaný objekt frontou klastru, atributy, které lze dotazovat, závisí na tom, jak je fronta otevřena:
   - Pokud je fronta klastru otevřena pro dotazování plus jeden nebo více vstupů, procházení nebo nastavení, musí existovat lokální instance fronty klastru, aby bylo otevření úspěšné. V tomto případě jsou atributy, které lze zjišťovat, platné pro lokální fronty.
   - Pokud je fronta klastru otevřena pro samotný dotaz nebo dotaz a výstup, lze zjistit pouze následující atributy; atribut **QType** má v tomto případě hodnotu QTCLUS:
     - CAQD
     - CAQN
     - IADBND
     - IADPER

- IADPRI
- IAIPUT
- IAQTYP

Je-li fronta klastru otevřena bez pevné vazby (tj. OOBNDN zadané ve volání MQOPEN nebo OOBNDQ zadané v případě, že má atribut **DefBind** hodnotu BNDNOT), mohou následná volání MQINQ pro frontu zjišťovat různé instance fronty klastru, ačkoli obvykle mají všechny instance stejné hodnoty atributů.

Další informace o frontách klastru naleznete v tématu Konfigurace klastru správců front.

5. Má-li být zjišťován počet atributů a některé z nich mají být nastaveny pomocí volání MQSET, může být vhodné umístit na začátek polí selektoru atributy, které mají být nastaveny, aby bylo možné pro MQSET použít stejná pole (se sníženým počtem).
6. Pokud se vyskytne více než jedna z varovných situací (viz parametr **CMPCOD**), vrácený kód příčiny je *první* v následujícím seznamu, který platí:
   a. RC2068
   b. RC2022
   c. RC2008
7. Další informace o atributech objektu viz:
   - “Atributy pro fronty” na stránce 1360
   - “Atributy pro seznamy názvů” na stránce 1388
   - “Atributy pro definice procesů v systému IBM i” na stránce 1389
   - “Atributy pro správce front v systému IBM i” na stránce 1391
8. Nová lokální fronta SYSTEM.ADMIN.COMMAND.EVENT se používá pro řazení zpráv do fronty, které jsou generovány při každém zadání příkazů. Zprávy jsou vkládány do této fronty pro většinu příkazů v závislosti na nastavení atributu správce front CMDEV:
   - ENABLED-zprávy událostí příkazu jsou generovány a vkládány do fronty pro všechny úspěšné příkazy.
   - Zprávy událostí NODISPLAY-command jsou generovány a vkládány do fronty pro všechny úspěšné příkazy kromě příkazu DISPLAY (MQSC) a příkazu Inquire (PCF).
   - DISABLED-zprávy událostí příkazu nejsou generovány (jedná se o počáteční výchozí hodnotu správce front).

Parametry

Volání MQINQ má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOBJ (10místné celé číslo se znaménkem)-vstup

Popisovač objektu.

Tento popisovač představuje objekt (libovolného typu) s požadovanými atributy. Popisovač musí být vrácen předchozím voláním MQOPEN, které určilo volbu OOINQ.

SELCNT (10místné celé číslo se znaménkem)-vstup

Počet selektorů.

Jedná se o počet selektorů, které jsou dodány v poli *SELS* . Jedná se o počet atributů, které mají být vráceny. Nula je platná hodnota. Maximální povolený počet je 256.

SELS (10místné celé číslo se znaménkem x SELCNT)-vstup

Pole selektorů atributů.

Toto je pole selektorů atributů **SELCNT** ; každý selektor identifikuje atribut (celé číslo nebo znak) s hodnotou, která je povinná.

Každý selektor musí být platný pro typ objektu, který *HOBJ* představuje, jinak volání selže s kódem dokončení CCFAIL a kódem příčiny RC2067.

Ve zvláštním případě front:

- Pokud selektor není platný pro fronty typu *any* , volání selže s kódem dokončení CCFAIL a kódem příčiny RC2067.
- Pokud je selektor použitelný pouze pro fronty jiného typu nebo typů, než je typ objektu, volání uspěje s kódem dokončení CCWARN a kódem příčiny RC2068.
- Je-li dotazovaná fronta frontou klastru, budou platné selektory záviset na tom, jak byla fronta vyřešena; další podrobnosti naleznete v poznámce o použití 4.

Selektory lze zadat v libovolném pořadí. Hodnoty atributů, které odpovídají selektorům celočíselných atributů (selektory IA*), jsou vráceny v *INTATR* ve stejném pořadí, v jakém se tyto selektory vyskytují v *SELS*. Hodnoty atributů, které odpovídají selektorům znakových atributů (selektory CA*), jsou vráceny v produktu *CHRATR* ve stejném pořadí, v jakém se tyto selektory vyskytují. Selektory IA* mohou být prokládány se selektory CA*; důležité je pouze relativní pořadí v rámci každého typu.

Poznámka:

1. Selektory celočíselných a znakových atributů jsou přiděleny ve dvou různých rozsazích; selektory IA* jsou umístěny v rozsahu IAFRST až IALAST a selektory CA* v rozsahu CAFRST až CALAST.

   Pro každý rozsah konstanty IALSTU a CALSTU definují nejvyšší hodnotu, kterou správce front přijímá.
2. Pokud se nejprve vyskytnou všechny selektory IA*, lze k adresování odpovídajících prvků v polích *SELS* a *INTATR* použít stejná čísla prvků.

Atributy, které lze zjišťovat, jsou uvedeny v následujících tabulkách. Pro selektory CA* je konstanta, která definuje délku výsledného řetězce v *CHRATR* v bajtech, uvedena v závorkách.

Tabulka 746. Selektory atributů MQINQ pro fronty

Selektor	Popis	Poznámka
CAALTD	Datum poslední změny (LNDATE).	1
CAALTT	Čas poslední změny (LNTIME).	1
CABRQN	Nadměrný název backout-requeue (LNQN).	5
CABASQ	Název fronty, na kterou se alias interpretuje (LNQN).	
CACFSN	Název struktury prostředku Coupling Facility (LNCFSN).	3
CACLN	Název klastru (LNCLUN).	1
CACLNL	Seznam názvů klastru (LNNLN).	1
CACRTD (všechny)	Datum vytvoření fronty (LNCRTD).	
CACRTT (není k)	Čas vytvoření fronty (LNCRTT).	
CAINIQ	Název inicializační fronty (LNQN).	
CAPRON	Název definice procesu (LNPRON).	
CAQD	Popis fronty (LNQD).	

Tabulka 746. Selektory atributů MQINQ pro fronty (pokračování)		
Selektor	**Popis**	**Poznámka**
CAQN	Název fronty (LNQN).	
CARQMN	Název vzdáleného správce front (LNQMN).	
CARQN-počet	Název vzdálené fronty, jak je znám ve vzdáleném správci front (LNQN).	
CATRGD	Data spouštěče (LNTRGD).	5
CAXQN	Název přenosové fronty (LNQN).	
IABTHR	Prahová hodnota vyřazených zpráv.	5
IACDEP	Počet zpráv ve frontě.	
IADBND	Výchozí vazba.	1
IADINP	Výchozí volba open-for-input.	5
IADPER	Výchozí perzistence zpráv.	
IADPRI	Výchozí priorita zprávy.	5
IADEFT	Typ definice fronty.	
IADIST	Podpora distribučního seznamu.	2
IAHGB	Zda se má zatvrdit počet vrácení.	5
IAIGET	Zda jsou povoleny operace získání.	
IAIPUT	Zda jsou povoleny operace vložení.	
IAMLEN-více informací	Maximální délka zprávy.	
IAMDEP	Maximální počet zpráv povolených ve frontě.	
IAMDS	Zda je priorita zprávy relevantní.	5
IAOIC	Počet volání MQOPEN, která mají otevřenou frontu pro vstup.	
IAOOC	Počet volání MQOPEN, která mají otevřenou frontu pro výstup.	
IAQDHE	Řídicí atribut pro události vysoké hloubky fronty.	4, 5
IAQDHL	Horní limit pro hloubku fronty.	4, 5
IAQDLE	Řídicí atribut pro události nízké hloubky fronty.	4, 5
IAQDLL	Dolní limit pro hloubku fronty.	4, 5
IAQDME	Řídicí atribut pro maximální počet událostí délky fronty.	4, 5
IAQSI	Limit pro interval služby fronty.	4, 5
IAQSIE	Řídicí atribut pro události intervalu služby fronty.	4, 5
IAQTYP	Typ fronty.	
IAQSGD	Dispozice skupiny sdílení front.	3
IARINT	Interval uchování fronty.	5
IASCOP	Rozsah definice fronty.	4, 5
IASHAR	Zda lze frontu sdílet pro vstup.	

Tabulka 746. Selektory atributů MQINQ pro fronty (pokračování)		
Selektor	**Popis**	**Poznámka**
IATRGC-změna pořadí	Ovládací prvek spouštěče.	
IATRGD	Hloubka spouątěče.	5
IATRGP-řízení	Priorita zprávy prahové hodnoty pro spouątěče.	5
IATRGT-další	Typ spouątěče.	
IAUSAG	Použití.	
CLWLUSEQ	Použít vzdálené fronty.	

Poznámka:

1. Podporováno na následujících platformách:
   - AIX
   - IBM i
   - Windows
   - z/OS

   a pro systém IBM MQ MQI clients připojený k těmto systémům.
2. Podporováno na následujících platformách:
   - AIX
   - IBM i
   - Windows

   a pro klienty IBM MQ připojené k těmto systémům.
3. z/OS Podporováno na systému z/OS.
4. z/OS Není podporováno na systému z/OS.
5. Není podporováno na systému VSE/ESA.

Tabulka 747. Selektory atributů MQINQ pro seznamy názvů		
Selektor	**Popis**	**Poznámka**
CAALTD	Datum poslední změny (LNDATE)	1
CAALTT	Čas poslední změny (LNTIME)	1
CALSTD	Popis seznamu názvů (LNNLD)	1
CALSTN	Název objektu seznamu názvů (LNNLN)	1
CANAMS	Názvy v seznamu názvů (LNQN x *Počet názvů v seznamu)*	1
IANAMC	Počet jmen v seznamu názvů	1
IAQSGD	Dispozice skupiny sdílení front	3

Tabulka 748. Selektory atributů MQINQ pro definice procesů		
Selektor	**Popis**	**Poznámka**
CAALTD	Datum poslední změny (LNDATE)	1
CAALTT	Čas poslední změny (LNTIME)	1
CAAPPI	Identifikátor aplikace (LNPROA)	5
CAENVD	Data prostředí (LNPROE)	5
CAPROD-základní model	Popis definice procesu (LNPROD)	5
CAPRON	Název definice procesu (LNPRON)	5
CAUSRD	Uživatelská data (LNPROU)	5
IAAPPT (mezinárodní smlouva)	Typ aplikace	5
IAQSGD	Dispozice skupiny sdílení front	3

Tabulka 749. Selektory atributů MQINQ pro správce front		
Selektor	**Popis**	**Poznámka**
CAALTD	Datum poslední změny (LNDATE)	1
CAALTT	Čas poslední změny (LNTIME)	1
CACADX-PŘIDÁNÍ	Název uživatelské procedury automatické definice kanálu (LNEXN)	1
CACLWD-pracovní místo	Data předaná uživatelské proceduře pracovní zátěže klastru (LNEXDA)	1
CACLWX-pracovní místo	Název uživatelské procedury pracovní zátěže klastru (LNEXN)	1
CACMDQ	Název vstupní fronty systémového příkazu (LNQN)	5
CADLQ-počet	Název fronty nedoručených zpráv (LNQN)	5
CADXQN	Výchozí název přenosové fronty (LNQN)	5
CAQMD	Popis správce front (LNQMD)	5
CAQMID	Identifikátor správce front (LNQMID)	1
CAQMN	Název lokálního správce front (LNQMN)	5
CAQSGN	Název skupiny sdílení front (LNQSGN)	3
CARPN	Název klastru, pro který správce front poskytuje služby úložiště (LNQMN)	1
CARPNL	Název objektu seznamu názvů obsahujícího názvy klastrů, pro které správce front poskytuje služby úložiště (LNNLN)	1
CMDEV	Řídicí atribut, který určuje, zda jsou zprávy generované při zadávání příkazů vkládány do fronty.	8
IAAUTE	Atribut řízení pro události oprávnění	4, 5

Tabulka 749. Selektory atributů MQINQ pro správce front (pokračování)		
Selektor	**Popis**	**Poznámka**
IACAD	Řídicí atribut pro automatickou definici kanálu	2
IACADE	Řídicí atribut pro události automatické definice kanálu	2
IACLXQ	Výchozí typ přenosové fronty klastru	4
IACLWL-pracovní	Délka pracovní zátěže klastru	1
IACCSI	Identifikátor znakové sady	5
IACMDL	Úroveň příkazů podporovaná správcem front	5
IACFGE	Atribut řízení pro události konfigurace	3
IADIST	Podpora seznamu distribuce	2
IAINHE	Řídicí atribut pro blokování událostí	4, 5
IALCLE-rozšíření	Atribut řízení pro lokální události	4, 5
IAMHND	Maximální počet popisovačů	5
IAMLEN-více informací	Maximální délka zprávy	5
IAMPRI	Maximální priorita	5
IAMUNC	Maximální počet nepotvrzených zpráv v rámci pracovní jednotky	5
IAPFME	Atribut řízení pro události výkonu	4, 5
IAPLAT	Platforma, na které je umístěn správce front	5
IARMTE	Atribut řízení pro vzdálené události	4, 5
IASSE	Řídicí atribut pro události zastavení spuštění	4, 5
IASYNC	Dostupnost synchronizačního bodu	5
IATRLFT-řízení	Životnost nepoužívaných neadministrativních témat	
IATRGI	Interval spouštěče	5

IACNT (10místné celé číslo se znaménkem)-vstup

Počet celočíselných atributů.

Jedná se o počet prvků v poli INTATR . Nula je platná hodnota.

Pokud se jedná alespoň o počet selektorů IA* v parametru **SELS** , vrátí se všechny požadované celočíselné atributy.

INTATR (10místné celé číslo se znaménkem x IACNT)-výstup

Pole celočíselných atributů.

Toto je pole celočíselných hodnot atributu *IACNT* .

Hodnoty celočíselných atributů jsou vráceny ve stejném pořadí jako selektory IA* v parametru **SELS** . Pokud pole obsahuje více prvků, než je počet selektorů IA*, přebytečné prvky se nezmění.

Pokud HOBJ představuje frontu, ale selektor atributů není použitelný pro tento typ fronty, specifická hodnota IAVNA se vrátí pro odpovídající prvek v poli INTATR .

CALEN (10místné celé číslo se znaménkem)-vstup

Délka vyrovnávací paměti atributů znaků.

Jedná se o délku parametru **CHRATR** v bajtech.

Musí se jednat alespoň o součet délek požadovaných znakových atributů (viz SELS). Nula je platná hodnota.

CHRATR (1 bajtový znakový řetězec x CALEN)-výstup

Znakové atributy.

Jedná se o vyrovnávací paměť, ve které jsou vráceny atributy znaků, které jsou zřetězeny dohromady. Délka vyrovnávací paměti je dána parametrem **CALEN** .

Znakové atributy jsou vráceny ve stejném pořadí jako selektory CA* v parametru **SELS** . Délka každého řetězce atributu je pevná pro každý atribut (viz SELS) a hodnota v něm je v případě potřeby vyplněna vpravo mezerami. Pokud je vyrovnávací paměť větší, než je potřeba, aby obsahovala všechny požadované atributy znaků (včetně výplně), bajty za poslední vrácenou hodnotou atributu se nezmění.

Pokud HOBJ představuje frontu, ale selektor atributů není použitelný pro tento typ fronty, znakový řetězec sestávající výhradně z hvězdiček (*) se vrátí jako hodnota tohoto atributu v souboru CHRATR.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující CMPCOD.

Pokud je CMPCOD CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCWARN:

RC2008
(2008, X'7D8') Pro atributy znaků není povolen dostatek místa.

RC2022
(2022, X'7E6') Není dostatek místa pro celočíselné atributy.

RC2068
(2068, X'814 ') Selektor nelze použít pro typ fronty.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2006
(2006, X'7D6') Délka znakových atributů není platná.

RC2007
(2007, X'7D7') Řetězec znakových atributů není platný.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2019
(2019, X'7E3') popisovač objektu není platný.

RC2021
(2021, X'7E5') Počet celočíselných atributů není platný.

RC2023
(2023, X'7E7') Pole atributů Integer je neplatné.

RC2038
(2038, X'7F6') Fronta není otevřena pro zjišťování.

RC2041
(2041, X'7F9') Definice objektu se od doby otevření změnila.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2065
(2065, X'811 ') Počet selektorů není platný.

RC2067
(2067, X'813 ') Selektor atributů není platný.

RC2066
(2066, X'812 ') Počet selektorů je příliš velký.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                    CALLP     MQINQ(HCONN : HOBJ : SELCNT :
C                                     SELS(1) : IACNT : INTATR(1) :
C                                     CALEN : CHRATR : CMPCOD :
C                                     REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQINQ            PR                  EXTPROC('MQINQ')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Object handle
D HOBJ                          10I 0 VALUE
D* Count of selectors
D SELCNT                        10I 0 VALUE
D* Array of attribute selectors
D SELS                          10I 0
```

```
     D* Count of integer attributes
     D IACNT                         10I 0 VALUE
     D* Array of integer attributes
     D INTATR                        10I 0
     D* Length of character attributes buffer
     D CALEN                         10I 0 VALUE
     D* Character attributes
     D CHRATR                          *   VALUE
     D* Completion code
     D CMPCOD                        10I 0
     D* Reason code qualifying CMPCOD
     D REASON                        10I 0
```

IBM i MQINQMP (vlastnost dotazové zprávy) na IBM i

Volání MQINQMP vrací hodnotu vlastnosti zprávy.

- “Syntaxe” na stránce 1307
- “Parametry” na stránce 1307
- “Prohlášení o RPG” na stránce 1311

Syntaxe

MQINQMP *(Hconn, Hmsg, InqPropOpts, Name, PropDesc, Type, ValueLength, Value, DataLength, CompCode, Reason)*

Parametry

Volání MQINQMP má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, pak musí být ustanoveno platné připojení na podprocesu dotazující vlastnost popisovače zprávy, jinak volání selže s RC2009.

HMSG (20místné celé číslo se znaménkem)-vstup

Jedná se o popisovač zprávy, který má být dotazován. Hodnota byla vrácena předchozím voláním funkce **MQCRTMH** .

INQOPT (MQIMPO)-vstup

Podrobnosti viz datový typ MQIMPO .

PRNAME (MQCHARV)-vstup

Popisuje název vlastnosti, která se má dotázat.

Pokud nelze nalézt žádnou vlastnost s tímto názvem, volání selže s příčinou RC2471.

Na konci názvu vlastnosti můžete použít znak procenta (%). Zástupný znak odpovídá žádnému nebo více znakům, včetně znaku tečky (.). To umožňuje aplikaci zjišťovat hodnotu mnoha vlastností. Vyvolejte MQINQMP s volbou IPINQF, abyste získali první odpovídající vlastnost, a znovu s volbou IPINQN, abyste získali další odpovídající vlastnost. Když nejsou k dispozici žádné další odpovídající vlastnosti, volání selže s RC2471. Je-li pole *ReturnedName* struktury InqPropOpts inicializováno s adresou nebo offsetem pro vrácený název vlastnosti, je tato operace dokončena při návratu z MQINQMP s názvem vlastnosti, pro kterou byla nalezena shoda. Pokud je pole *VSBufSize ReturnedName* ve struktuře InqPropOpts menší než délka vráceného názvu vlastnosti, kód dokončení je nastaven na CCFAIL s příčinou RC2465.

Vlastnosti, které mají známá synonyma, jsou vráceny následujícím způsobem:

1. Vlastnosti s předponou "mqps." jsou vráceny s názvem vlastnosti IBM MQ . Například "MQTopicString" je vrácený název spíše než "mqps.Top".
2. Vlastnosti s předponou "jms." nebo "mcd." jsou vráceny jako název pole záhlaví JMS . Například "JMSExpiration" je vrácený název spíše než "jms.Exp".
3. Vlastnosti s předponou "usr." jsou vráceny bez této předpony. Vrátí se například hodnota "Color", nikoli hodnota "usr.Color".

Vlastnosti se synonymy jsou vráceny pouze jednou.

V programovacím jazyku RPG jsou pro zjišťování všech vlastností a všech vlastností, které začínají řetězcem "usr.", definovány následující proměnné maker:

INQALL
Informujte se o všech vlastnostech zprávy.

INQUSR
Dotázat se na všechny vlastnosti zprávy, která začíná "usr.". Vrácené jméno je vráceno bez "usr." předpona.

Je-li uvedeno IPINQN, ale název se změnil od předchozího volání, nebo se jedná o první volání, pak je IPINQF implicitní.

Další informace o použití názvů vlastností viz Názvy vlastností a Omezení názvů vlastností .

PRPDSC (MQPD)-výstup

Tato struktura se používá k definování atributů vlastnosti, včetně toho, co se stane, když vlastnost není podporována, do jakého kontextu zprávy vlastnost patří a do jakých zpráv by měla být vlastnost zkopírována. Podrobnosti o této struktuře viz MQPD .

TYPE (10místné celé číslo se znaménkem)-vstup/výstup

Při návratu z volání MQINQMP je tento parametr nastaven na datový typ *Hodnota*. Datový typ může být libovolný z následujících:

Počet TYPOBOLŮ
Logická hodnota.

TYPBST
bajtový řetězec.

TYPI8
8bitové celé číslo se znaménkem.

TYPI16
16bitové celé číslo se znaménkem.

TYPI32
32bitové celé číslo se znaménkem.

TYPI64
64bitové celé číslo se znaménkem.

TYPF32
32bitové číslo s pohyblivou řádovou čárkou.

TYPF64
64bitové číslo s pohyblivou řádovou čárkou.

TYPSTR
Znakový řetězec.

TYPNUL
Vlastnost existuje, ale má hodnotu null.

Pokud není datový typ hodnoty vlastnosti rozpoznán, vrátí se TYPSTR a do oblasti *Hodnota* se umístí řetězcová reprezentace hodnoty. Řetězcovou reprezentaci datového typu lze nalézt v poli *IPTYP* parametru *IPOPT* . Vrátí se varovný kód dokončení s příčinou RC2467.

Kromě toho, pokud je uvedena volba IPCTYP, je požadován převod hodnoty vlastnosti. Jako vstup použijte *Typ* , chcete-li určit datový typ, jako který má být vlastnost vrácena. Podrobnosti o převodu datového typu viz popis volby IPCTYP v souboru "MQIMPO (Volby vlastnosti dotazové zprávy) na systému IBM i" na stránce 1098 .

Pokud nepožadujete převod typu, můžete na vstupu použít následující hodnotu:

TYP ZAŘÍZENÍ
Hodnota vlastnosti je vrácena bez převodu jejího datového typu.

VALLEN (10místné celé číslo se znaménkem)-vstup

Délka oblasti hodnot v bajtech.

Zadejte nulu pro vlastnosti, pro které nepožadujete vrácenou hodnotu. Může se jednat o vlastnosti navržené aplikací tak, aby měly hodnotu null nebo prázdný řetězec. Uveďte také nulu, pokud byla uvedena volba IPQLEN; v tomto případě se nevrátí žádná hodnota.

VALUE (1bajtový bit stringxVALLEN)-výstup

Jedná se o oblast, která má obsahovat požadovanou hodnotu vlastnosti. Vyrovnávací paměť by měla být zarovnána na hranici odpovídající vracené hodnotě. Pokud tak neučiníte, může dojít k chybě při pozdějším přístupu k hodnotě.

Je-li hodnota *VALLEN* menší než délka hodnoty vlastnosti, přesune se co největší část hodnoty vlastnosti do *VALUE* a volání selže s kódem dokončení CCFAIL a příčinou RC2469.

Znaková sada dat v *VALUE* je dána polem IPRETCSI v parametru INQOPT. Kódování dat v *VALUE* je dáno polem IPRETENC v parametru INQOPT.

Je-li parametr *VALLEN* nulový, na hodnotu VALUE se neodkazuje.

DATLEN (10místné celé číslo se znaménkem)-výstup

Jedná se o délku (v bajtech) skutečné hodnoty vlastnosti vrácené v oblasti *Hodnota* .

Pokud je *DataLength* menší než délka hodnoty vlastnosti, *DataLength* je stále zadán při návratu z volání MQINQMP. To umožňuje aplikaci určit velikost vyrovnávací paměti potřebné k uložení hodnoty vlastnosti a poté znovu zadat volání s vyrovnávací pamětí odpovídající velikosti.

Také mohou být vráceny následující hodnoty.

Pokud je parametr *Typ* nastaven na TYPSTR nebo TYPBST:

VLEMP
Vlastnost existuje, ale neobsahuje žádné znaky ani bajty.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CompCode*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu CCWARN:

RC2492
(2492, X'09BC') Vrácený název vlastnosti nebyl převeden.

RC2466
(2466, X'09A2') Hodnota vlastnosti nebyla převedena.

RC2467
(2467, X'09A3') Datový typ vlastnosti není podporován.

RC2421
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nelze analyzovat.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'0852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'086D') Primární a domovská ASID se liší.

RC2004
(2004, X'07D4') Parametr hodnoty není platný.

RC2005
(2005, X'07D5') Parametr délky hodnoty není platný.

RC2219
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

RC2009
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

RC2010
(2010, X'07DA') Parametr délky dat není platný.

RC2464
(2464, X'09A0') Struktura voleb vlastností dotazové zprávy je neplatná.

RC2460
(2460, X'099C') popisovač zprávy není platný.

RC2499
(2499, X'09C3') Obsluha zprávy je již používána.

RC2064
(2046, X'07F8') Volby jsou neplatné nebo nekonzistentní.

RC2482
(2482, X'09B2') Struktura deskriptoru vlastnosti není platná.

RC2470
(2470, X'09A6') Převod ze skutečného na požadovaný datový typ není podporován.

RC2442
(2442, X'098A') Neplatný název vlastnosti.

RC2465
(2465, X'09A1') Název vlastnosti je příliš velký pro vrácenou vyrovnávací paměť názvů.

RC2471
(2471, X'09A7) Vlastnost není k dispozici.

RC2469
(2469, X'09A5') Hodnota vlastnosti je pro oblast Hodnota příliš velká.

RC2472
(2472, X'09A8') Došlo k chybě formátu čísla v datech hodnoty.

RC2473
(2473, X'09A9') Neplatný požadovaný typ vlastnosti.

RC2111
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

RC2071
(2071, X'0871 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'0893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech viz:

- položky IBM MQ for z/OS zprávy, dokončení, a kódy příčiny pro IBM MQ for z/OS
- Zprávy a kódy příčiny pro všechny ostatní IBM MQ platformy

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQINQMP(HCONN : HMSG : INQOPT :
                                        PRNAME : PRPDSC : TYPE :
                                        VALLEN : VALUE : DATLEN :
                                        CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQINQMP          PR                  EXTPROC('MQINQMP')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Message handle
D HMSG                          20I 0 VALUE
D* Options that control the action of MQINQMP
D INQOPT                        72A
D* Property name
D PRNAME                        32A
D* Property descriptor
D PRPDSC                        24A
D* Property data type
D TYPE                          10I 0
D* Length in bytes of the Value area
D VALLEN                        10I 0 VALUE
D* Property value
D VALUE                           *   VALUE
D* Length of the property value
D DATLEN                        10I 0
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CompCode
D REASON                        10I 0
```

IBM i MQMHBUF (Převést popisovač zprávy do vyrovnávací paměti) na IBM i

Modul MQMHBUF převádí manipulátor zprávy na vyrovnávací paměť a jedná se o inverzní volání MQBUFMH.

- “Syntaxe” na stránce 1311
- “Poznámky k použití” na stránce 1312
- “Parametry” na stránce 1312
- “Prohlášení o RPG” na stránce 1314

Syntaxe

MQMHBUF *(Hconn, Hmsg, MsgHBufOpts, Name, MsgDesc, BufferLength, Buffer, DataLength, CompCode, Reason)*

Poznámky k použití

MQMHBUF převádí manipulátor zprávy na vyrovnávací paměť.

Můžete jej použít s uživatelskou procedurou rozhraní MQGET API pro přístup k určitým vlastnostem pomocí rozhraní API vlastností zprávy a poté předat tyto vlastnosti ve vyrovnávací paměti zpět aplikaci navržené pro použití záhlaví MQRFH2 namísto popisovačů zpráv.

Toto volání je inverzní volání MQBUFMH, které lze použít k analýze vlastností zprávy z vyrovnávací paměti do manipulátoru zprávy.

Parametry

Volání MQMHBUF má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota *HCONN* musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **HMSG** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podprocesu, který odstranil popisovač zprávy. Pokud není ustanoveno platné připojení, volání selže s RC2009.

HMSG (20místné celé číslo se znaménkem)-vstup

Tento manipulátor je manipulátor zpráv, pro který je vyžadována vyrovnávací paměť.

Hodnota byla vrácena předchozím voláním MQCRTMH.

MHBOPT (MQMHBO)-vstup

Struktura MQMHBO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou vyrovnávací paměti vytvářeny z manipulátorů zpráv.

Podrobnosti viz "MQBMHO (volby vyrovnávací paměti pro obsluhu zpráv) v systému IBM i" na stránce 1013.

PRNAME (MQCHARV)-vstup

Název vlastnosti nebo vlastností, které mají být vloženy do vyrovnávací paměti.

Pokud nelze nalézt žádnou vlastnost odpovídající názvu, volání selže s RC2471.

zástupné znaky

Chcete-li do vyrovnávací paměti vložit více než jednu vlastnost, můžete použít zástupný znak. Chcete-li tak učinit, použijte znak procenta (%) na konci názvu vlastnosti. Tento zástupný znak neodpovídá žádnému nebo více znakům, včetně znaku tečky (.).

Další informace o použití názvů vlastností viz Názvy vlastností a Omezení názvů vlastností .

MSGDSC (MQMD)-vstupní/výstupní

Struktura *MSGDSC* popisuje obsah oblasti vyrovnávací paměti.

Ve výstupu jsou pole *Encoding*, *CodedCharSetId* a *Format* nastavena tak, aby správně popisovala kódování, identifikátor znakové sady a formát dat v oblasti vyrovnávací paměti, jak jsou zapsána voláním.

Data v této struktuře jsou ve znakové sadě a kódování aplikace.

BUFLEN (10místné celé číslo se znaménkem)-vstup

BUFFLEN je délka oblasti vyrovnávací paměti v bajtech.

BUFFER (1bajtový bitový řetězec x BUFLEN)-vstup/výstup

BUFFER definuje oblast obsahující vyrovnávací paměť zpráv. Pro většinu dat je nutné vyrovnávací paměť zarovnat na 4bajtovou hranici.

Pokud *BUFFER* obsahuje znaková nebo číselná data, nastavte pole *CodedCharSetId* a *Encoding* v parametru **MSGDSC** na hodnoty odpovídající datům; to v případě potřeby umožní převod dat.

Pokud jsou ve vyrovnávací paměti zpráv nalezeny vlastnosti, jsou volitelně odebrány; později budou k dispozici v popisovači zprávy při návratu z volání.

V programovacím jazyku C je parametr deklarován jako ukazatel na prázdno, což znamená, že jako parametr lze zadat adresu libovolného typu dat.

Je-li parametr **BUFLEN** nulový, na *BUFFER* se neodkazuje. V tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo v sestavovacím modulu System/390 hodnotu null.

DATLEN (10místné celé číslo se znaménkem)-výstup

DATLEN je délka vrácených vlastností ve vyrovnávací paměti v bajtech. Pokud je hodnota nula, žádné vlastnosti neodpovídají hodnotě uvedené v souboru *PRNAME* a volání selže s kódem příčiny RC2471.

Pokud je hodnota *BUFLEN* menší než délka požadovaná pro uložení vlastností ve vyrovnávací paměti, volání MQMHBUF selže s hodnotou RC2469, ale hodnota je stále zadána do *DATLEN*. To umožňuje aplikaci určit velikost vyrovnávací paměti potřebnou pro přizpůsobení vlastností a poté znovu zadat volání s požadovaným *BUFLEN*.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

RC2501
(2501, X'095C') Struktura voleb popisovače zprávy do vyrovnávací paměti je neplatná.

RC2004
(2004, X'07D4') Parametr vyrovnávací paměti není platný.

RC2005
(2005, X'07D5') Parametr délky vyrovnávací paměti není platný.

RC2219
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

RC2009
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

RC2010
(2010, X'07DA') Parametr délky dat není platný.

RC2460
(2460, X'099C') popisovač zprávy není platný.

RC2026
(2026, X'07EA') Deskriptor zpráv není platný.

RC2499
(2499, X'09C3') Obsluha zprávy je již používána.

RC2046
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2442
(2442, X'098A') Název vlastnosti je neplatný.

RC2471
(2471, X'09A7') Vlastnost není k dispozici.

RC2469
(2469, X'09A5') Hodnota BufferLength je příliš malá na to, aby obsahovala zadané vlastnosti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                   CALLP     MQMHBUF(HCONN : HMSG : MHBOPT :
                                      PRNAME : MSGDSC : BUFLEN :
                                      BUFFER : DATLEN :
                                      CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQMHBUF          PR                  EXTPROC('MQMHBUF')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Message handle
D HMSG                          20I 0 VALUE
D* Options that control the action of MQMHBUF
D MHBOPT                        12A
D* Property name
D PRNAME                        32A
D* Message descriptor
D MSGDSC                       364A
D* Length in bytes of the Buffer area
D BUFLEN                        10I 0 VALUE
D* Area to contain the properties
D BUFFER                          *   VALUE
D* Length of the properties
D DATLEN                        10I 0
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CompCode
D REASON                        10I 0
```

IBM i MQOPEN (Otevřít objekt) na IBM i

Volání MQOPEN zavádí přístup k objektu.

Následující typy objektů jsou platné:

- Fronta (včetně distribučních seznamů)
- Seznam názvů
- Definice procesu

- Správce front
- Téma

Index

Syntaxe

MQOPEN *(HCONN, OBJDSC, OPTS, HOBJ, CMPCOD, REASON)*

Poznámky k použití

1. Otevřený objekt je jeden z následujících:
   - Fronta za účelem:
     - Získat nebo procházet zprávy (pomocí volání MQGET)
     - Vložit zprávy (pomocí volání MQPUT)
     - Dotaz na atributy fronty (pomocí volání MQINQ)
     - Nastavit atributy fronty (pomocí volání MQSET)

     Pokud je pojmenovaná fronta modelovou frontou, vytvoří se dynamická lokální fronta.

     Distribuční seznam je speciální typ objektu fronty, který obsahuje seznam front. Může být otevřen pro vložení zpráv, ale ne pro získání nebo procházení zpráv, nebo pro dotazování nebo nastavení atributů. Další podrobnosti viz poznámka k použití 8.

     Fronta, která má typ `QSGDISP(GROUP)` , je speciálním typem definice fronty, který nelze použít s voláními MQOPEN nebo MQPUT1 .
   - Seznam názvů, aby bylo možné:
     - Informujte se o názvech front v seznamu (pomocí volání MQINQ).
   - Definice procesu za účelem:
     - Informujte se o atributech procesu (pomocí volání MQINQ).
   - Správce front za účelem:
     - Informujte se o atributech lokálního správce front (pomocí volání MQINQ).
2. Je platné, aby aplikace otevřela stejný objekt více než jednou. Pro každé otevření se vrátí jiný popisovač objektu. Každý vrácený popisovač lze použít pro funkce, pro které bylo provedeno odpovídající otevření.
3. Pokud je otevíraný objekt frontou, ale nikoli frontou klastru, veškeré rozlišení názvů v lokálním správci front se uskuteční v době volání MQOPEN. Může se jednat o jednu nebo více následujících položek pro konkrétní volání MQOPEN:
   - Rozlišení aliasu na název základní fronty
   - Rozlišení názvu lokální definice vzdálené fronty na název vzdáleného správce front a názvu, pod kterým je fronta ve vzdáleném správci front známa.
   - Rozlišení názvu vzdáleného správce front na název lokální přenosové fronty

   Mějte však na paměti, že následná volání MQINQ nebo MQSET pro manipulátor se vztahují pouze k názvu, který byl otevřen, a nikoli k objektu, který je výsledkem po rozpoznání názvu. Pokud je například otevřený objekt alias, atributy vrácené voláním MQINQ jsou atributy aliasu, nikoli atributy

základní fronty, do které se alias převádí. Kontrola rozlišování názvů se však stále provádí bez ohledu na to, co je určeno pro parametr **OPTS** na odpovídajícím MQOPEN.

Je-li otevíraný objekt frontou klastru, může k rozpoznání názvu dojít v době volání MQOPEN nebo může být odloženo na pozdější dobu. Bod, ve kterém dochází k rozpoznání, je řízen volbami OOBND* určenými ve volání MQOPEN:

- OOBNDO
- OOBNDN
- OOBNDQ

Další informace o rozlišování názvů pro fronty klastru naleznete v tématu Rozlišení názvů .

4. Atributy objektu se mohou měnit, zatímco aplikace má objekt otevřený. V mnoha případech si to aplikace nevšimne, ale pro určité atributy označí správce front popisovač jako neplatný. Patří mezi ně:
   - Jakýkoli atribut, který ovlivňuje rozlišení názvu objektu. To platí bez ohledu na použité volby otevření a zahrnuje následující:
     - Změna atributu **BaseQName** alias fronty, která je otevřená.
     - Změna atributů fronty **RemoteQName** nebo **RemoteQMgrName** pro jakýkoli manipulátor otevřený pro tuto frontu nebo pro frontu, která se prostřednictvím této definice interpretuje jako alias správce front.
     - Jakákoli změna, která způsobí, že se aktuálně otevřený popisovač pro vzdálenou frontu interpretuje jako jiná *přenosová* fronta, nebo se vůbec nepodaří vyřešit jako jedna. Může například zahrnovat:
       - Změna atributu **XmitQName** lokální definice vzdálené fronty, bez ohledu na to, zda je definice používána pro frontu nebo pro alias správce front.

       Existuje jedna výjimka, a to vytvoření nové přenosové fronty. Popisovač, který by byl vyřešen pro tuto frontu, kdyby byl přítomen při otevření popisovače, ale místo něj by byl převeden na výchozí přenosovou frontu, není neplatný.
     - Změna atributu správce front **DefXmitQName** . V tomto případě jsou všechny otevřené popisovače, které se převedly na dříve pojmenovanou frontu (která se na ni převedla pouze proto, že se jednalo o výchozí přenosovou frontu), označeny jako neplatné. Popisovače, které byly vyřešeny pro tuto frontu z jiných důvodů, nejsou ovlivněny.
   - Atribut fronty **Shareability** , pokud existují dva nebo více manipulátorů, které momentálně poskytují přístup OOINPS pro tuto frontu, nebo pro frontu, která se interpretuje pro tuto frontu. Pokud ano, *všechny* popisovače, které jsou otevřené pro tuto frontu nebo pro frontu, která se interpretuje pro tuto frontu, jsou označeny jako neplatné bez ohledu na volby otevření.
   - Atribut fronty **Usage** pro všechny manipulátory, které jsou otevřené pro tuto frontu, nebo pro frontu, která se interpretuje pro tuto frontu, bez ohledu na volby otevření.

   Je-li manipulátor označen jako neplatný, všechna následná volání (jiná než MQCLOSE) používající tento manipulátor selžou s kódem příčiny RC2041; aplikace by měla zadat volání MQCLOSE (s použitím původního manipulátoru) a znovu otevřít frontu. Všechny nepotvrzené aktualizace oproti starému popisovači z předchozích úspěšných volání mohou být i nadále potvrzeny nebo vráceny zpět, jak vyžaduje logika aplikace.

   Pokud změna atributu způsobí, že k tomu dojde, musí být použita speciální verze příkazu "force" .
5. Správce front provádí kontroly zabezpečení při zadání volání MQOPEN, aby ověřil, zda má identifikátor uživatele, pod kterým je aplikace spuštěna, před povolením přístupu odpovídající úroveň oprávnění. Kontrola oprávnění se provádí na názvu objektu, který se otevírá, a ne na názvu nebo názvech, které jsou výsledkem vyřešení názvu.

   Je-li otevíraný objekt modelovou frontou, provede správce front úplnou kontrolu zabezpečení pro název modelové fronty i pro název vytvořené dynamické fronty. Je-li výsledná dynamická fronta otevřena explicitně, provede se další kontrola zabezpečení prostředků podle názvu dynamické fronty.

6. Vzdálenou frontu lze zadat jedním ze dvou způsobů v parametru **OBJDSC** tohoto volání (viz pole *ODON* a *ODMN* popsaná v tématu “MQOD (Object descriptor) na systému IBM i” na stránce 1154):

   - Zadáním názvu lokální definice vzdálené fronty pro parametr *ODON* . V tomto případě produkt *ODMN* odkazuje na lokálního správce front a lze jej zadat jako mezery.

     Ověření zabezpečení provedené lokálním správcem front ověřuje, zda má uživatel oprávnění k otevření lokální definice vzdálené fronty.
   - Zadáním názvu vzdálené fronty tak, jak je známa vzdálenému správci front, do parametru *ODON* . V tomto případě je *ODMN* název vzdáleného správce front.

     Ověření zabezpečení provedené lokálním správcem front ověřuje, zda má uživatel oprávnění k odesílání zpráv do přenosové fronty v důsledku procesu rozpoznávání názvů.

   V obou případech:

   - Lokální správce front neodesílá do vzdáleného správce front žádné zprávy, aby zkontroloval, zda je uživatel oprávněn vkládat zprávy do fronty.
   - Když zpráva dorazí do vzdáleného správce front, může ji vzdálený správce front odmítnout, protože uživatel, který zprávu spustil, nemá příslušnou autorizaci.
7. Volání MQOPEN s volbou OOBRW vytvoří kurzor procházení pro použití s voláními MQGET, která určují popisovač objektu a jednu z voleb procházení. To umožňuje skenovat frontu bez změny jejího obsahu. Zprávu, která byla nalezena procházením, lze později odebrat z fronty pomocí volby GMMUC.

   Pro jednu aplikaci může být aktivní více kurzorů procházení zadáním několika požadavků MQOPEN pro stejnou frontu.
8. Pro použití distribučních seznamů platí následující poznámky.

   - Pole ve struktuře MQOD musí být při otevírání distribučního seznamu nastavena takto:
     - *ODVER* musí být ODVER2 nebo vyšší.
     - *ODOT* musí být OTQ.
     - *ODON* musí být prázdný nebo prázdný řetězec.
     - *ODMN* musí být prázdný nebo prázdný řetězec.
     - *ODREC* musí být větší než nula.
     - Jedna z hodnot *ODORO* a *ODORP* musí být nula a druhá nenulová.
     - Nenulová hodnota může být nejvýše jedna z hodnot *ODRRO* a *ODRRP* .
     - Musí existovat záznamy objektů *ODREC* , adresované buď *ODORO* , nebo *ODORP*. Záznamy objektů musí být nastaveny na názvy cílových front, které se mají otevřít.
     - Je-li jedna z hodnot *ODRRO* a *ODRRP* nenulová, musí existovat *ODREC* záznamů odpovědí. Jsou nastaveny správcem front, pokud je volání dokončeno s kódem příčiny RC2136.

     Produkt version-2 MQOD lze také použít k otevření jedné fronty, která není v distribučním seznamu, a to tak, že zajistíte, že *ODREC* bude nula.
   - V parametru **OPTS** jsou platné pouze následující volby otevření:
     - OOOUT
     - OOPAS*
     - OOSET*
     - OOALTU-OOALTU
     - OOFIQ
   - Cílové fronty v rozdělovníku mohou být lokální, alias nebo vzdálené fronty, ale nemohou být modelovými frontami. Pokud je uvedena modelová fronta, tato fronta se neotevře, s kódem příčiny RC2057. To však nebrání úspěšnému otevření jiných front v seznamu.
   - Parametry kódu dokončení a kódu příčiny jsou nastaveny takto:

– Pokud jsou všechny otevřené operace pro fronty v rozdělovníku úspěšné nebo selhávají stejným způsobem, jsou parametry kódu dokončení a kódu příčiny nastaveny tak, aby popisovaly společný výsledek. V tomto případě nejsou nastaveny záznamy odpovědí MQRR (jsou-li poskytovány aplikací).

  Například, pokud je každé otevření úspěšné, kód dokončení je nastaven na CCOK a kód příčiny je RCNONE; pokud každé otevření selže, protože neexistuje žádná z front, parametry jsou nastaveny na CCFAIL a RC2085.

– Pokud operace otevření pro fronty v rozdělovníku nejsou všechny úspěšné nebo selhávají stejným způsobem:

  - Parametr kódu dokončení je nastaven na hodnotu CCWARN, pokud bylo alespoň jedno otevření úspěšné, a na hodnotu CCFAIL, pokud se vše nezdařilo.
  - Parametr kódu příčiny je nastaven na hodnotu RC2136.
  - Záznamy odezvy (jsou-li poskytovány aplikací) jsou nastaveny na jednotlivé kódy dokončení a kódy příčiny pro fronty v rozdělovníku.

- Po úspěšném otevření distribučního seznamu lze manipulátor *HOBJ* vrácený voláním použít v následných voláních MQPUT pro vložení zpráv do front v distribučním seznamu a ve volání MQCLOSE pro vzdání se přístupu k distribučnímu seznamu. Jediná platná volba zavření pro distribuční seznam je CONONE.

  Volání MQPUT1 lze také použít k vložení zprávy do distribučního seznamu. Struktura MQOD definující fronty v seznamu je určena jako parametr pro toto volání.

- Každé úspěšně otevřené místo určení v distribučním seznamu se při kontrole, zda aplikace překročila povolený maximální počet manipulátorů, počítá jako *samostatný* manipulátor (viz atribut správce front **MaxHandles**). To platí i v případě, že dvě nebo více míst určení v distribučním seznamu se skutečně vyřeší do stejné fyzické fronty. Pokud by volání MQOPEN nebo MQPUT1 pro distribuční seznam způsobilo překročení počtu popisovačů používaných aplikací *MaxHandles*, volání se nezdaří s kódem příčiny RC2017.
- Každé místo určení, které je úspěšně otevřeno, má hodnotu svého atributu **OpenOutputCount** zvýšenou o jednu. Pokud se dva nebo více cílů v rozdělovníku skutečně vyřeší na stejnou fyzickou frontu, má tato fronta svůj atribut **OpenOutputCount** zvýšený o počet míst určení v rozdělovníku, která se vyřeší na tuto frontu.
- Jakákoli změna definic front, která by způsobila, že by se manipulátor stal neplatným, kdyby byly fronty otevřeny jednotlivě (například změna v cestě řešení), nezpůsobí, že se manipulátor rozdělovníku stane neplatným. Při použití manipulátoru distribučního seznamu v následném volání MQPUT však dojde k selhání pro tuto konkrétní frontu.
- Je platné, aby rozdělovník obsahoval pouze jeden cíl.

9. Následující poznámky platí pro použití front klastru.

- Když je fronta klastru poprvé otevřena a lokální správce front není správcem front úplného úložiště, získá lokální správce front informace o frontě klastru ze správce front úplného úložiště. Je-li síť zaneprázdněna, může lokální správce front přijímat potřebné informace od správce front úložiště trvat několik sekund. V důsledku toho může aplikace vydávající volání MQOPEN čekat až 10 sekund před návratem řízení z volání MQOPEN. Pokud lokální správce front neobdrží potřebné informace o frontě klastru v této době, volání selže s kódem příčiny RC2189.
- Je-li fronta klastru otevřena a v klastru existuje více instancí fronty, závisí skutečně otevřená instance na volbách zadaných ve volání MQOPEN:

  – Pokud uvedené volby zahrnují některou z následujících možností:

    - OOBRW
    - OOINPQ
    - OOINPX
    - OOINPS
    - OOSET

Instance otevřené fronty klastru musí být lokální instance. Pokud neexistuje žádná lokální instance fronty, volání MQOPEN se nezdaří.

– Pokud uvedené volby nezahrnují nic z výše uvedeného, ale zahrnují jednu nebo obě z následujících možností:

  - OOINQ
  - OOOUT

  Otevřená instance je lokální instance, pokud existuje, a jinak vzdálená instance. Instance zvolená správcem front však může být změněna uživatelskou procedurou pracovní zátěže klastru (pokud existuje).

Další informace o frontách klastru naleznete v tématu Fronty klastru.

10. Aplikacím spuštěným monitorem spouštěčů je předán název fronty, která je přidružena k aplikaci při jejím spuštění. Tento název fronty lze zadat v parametru **OBJDSC** pro otevření fronty. Další podrobnosti naleznete v popisu struktury MQTMC.
11. Při použití volby OORLOQ je lokální fronta již vrácena, když je otevřena lokální, alias nebo modelová fronta, ale nejedná se o případ, kdy je například otevřena vzdálená fronta nebo jiná než lokální fronta klastru; hodnoty ResolvedQName a ResolvedQMgrjsou zadány spolu s názvy RemoteQName a RemoteQMgr, které jsou nalezeny v definici vzdálené fronty, nebo podobně s vybranou vzdálenou frontou klastru. Je-li při otevírání například vzdálené fronty zadána hodnota OORLOQ, bude jako přenosová fronta, do které budou vkládány zprávy, nyní zadána hodnota ResolvedQName . Název ResolvedQMgrbude zadán s názvem lokálního správce front, který je hostitelem přenosové fronty. Pokud má uživatel oprávnění k procházení, vstupu nebo výstupu ve frontě, má potřebné oprávnění k zadání tohoto příznaku ve volání MQOPEN. Není potřeba žádné speciální oprávnění.

Parametry

Volání MQOPEN má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

OBJDSC (MQOD)-vstupní/výstupní

Deskriptor objektu.

Toto je struktura, která identifikuje objekt, který se má otevřít; podrobnosti viz “MQOD (Object descriptor) na systému IBM i” na stránce 1154 .

Pokud je pole *ODON* v parametru **OBJDSC** název modelové fronty, dynamická lokální fronta. je vytvořen s atributy modelové fronty; k tomu dochází bez ohledu na otevřené volby určené parametrem **OPTS** . Následné operace používající funkci *HOBJ* vrácenou voláním MQOPEN jsou prováděny v nové dynamické frontě, nikoli v modelové frontě. To platí i pro volání MQINQ a MQSET. Název modelové fronty v parametru **OBJDSC** je nahrazen názvem vytvořené dynamické fronty. Typ dynamické fronty je určen hodnotou atributu **DefinitionType** modelové fronty (viz “Atributy pro fronty” na stránce 1360). Informace o volbách zavření použitelných pro dynamické fronty naleznete v popisu volání MQCLOSE.

OPTS (10místné celé číslo se znaménkem)-vstup

Volby, které řídí akci MQOPEN.

Musí být zadána alespoň jedna z následujících voleb:

- OOBRW
- OOINP* (pouze jeden z nich)
- OOINQ

- OOOUT
- OOSET
- OORLQ

Další volby lze zadat podle potřeby. Je-li požadována více než jedna volba, hodnoty lze přidat (nepřidávejte stejnou konstantu více než jednou). Kombinace, které nejsou platné, jsou uvedeny; všechny ostatní kombinace jsou platné. Povoleny jsou pouze volby použitelné pro typ objektu určený parametrem *OBJDSC* (viz Platné volby MQOPEN pro každý typ fronty).

Volby přístupu: Následující volby řídí typ operací, které lze na objektu provést:

OOINPQ
Otevřít frontu pro získání zpráv pomocí výchozího nastavení definovaného frontou.

Fronta je otevřena pro použití s následnými voláními MQGET. Typ přístupu je buď sdílený, nebo výlučný, v závislosti na hodnotě atributu fronty **DefInputOpenOption** ; podrobnosti viz “Atributy pro fronty” na stránce 1360 .

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami.

OOINPS
Chcete-li získat zprávy se sdíleným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání může být úspěšné, pokud je fronta momentálně otevřena touto nebo jinou aplikací s OOINPS, ale selže s kódem příčiny RC2042 , pokud je fronta momentálně otevřená s OOOINPX.

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami.

OOINPX
Chcete-li získat zprávy s výlučným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání selže s kódem příčiny RC2042 , pokud je fronta momentálně otevřena touto nebo jinou aplikací pro vstup libovolného typu (OOOINPS nebo OOINPX).

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami.

Pro tyto volby platí následující poznámky:

- Lze zadat pouze jednu z těchto voleb.
- Volání MQOPEN s jednou z těchto voleb může být úspěšné i v případě, že je atribut fronty **InhibitGet** nastaven na hodnotu QAGETI (ačkoli následná volání MQGET selžou, když je atribut nastaven na tuto hodnotu).
- Je-li fronta definována jako nesdílená (tj. atribut fronty **Shareability** má hodnotu QANSHR), pokusy o otevření fronty pro sdílený přístup se považují za pokusy o otevření fronty s výlučným přístupem.
- Pokud je fronta aliasů otevřena s jednou z těchto voleb, test pro výlučné použití (nebo pro to, zda má jiná aplikace výlučné použití) je proti základní frontě, do které se alias převádí.
- Tyto volby nejsou platné, pokud *ODMN* je název aliasu správce front. To platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro aliasy správce front je název lokálního správce front.

OOBRW
Chcete-li procházet zprávy, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET s jednou z následujících voleb:

- GMBRWF
- GMBRWN

- GMBRWC

To je povoleno, i když je fronta momentálně otevřená pro OOINPX. Volání MQOPEN s volbou OOBRW vytvoří kurzor procházení a umístí jej logicky před první zprávu ve frontě; další informace viz pole *GMOPT* popsané v části "MQGMO (Volby Get-message) na systému IBM i" na stránce 1073 .

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami. Není také platný, pokud *ODMN* je název aliasu správce front. To platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro aliasy správce front je název lokálního správce front.

OOOUT
Otevřená fronta pro vložení zpráv nebo téma či řetězec tématu pro publikování zpráv.

Fronta je otevřena pro použití s následnými voláními MQPUT.

Volání MQOPEN s touto volbou může být úspěšné i v případě, že je atribut fronty **InhibitPut** nastaven na hodnotu QAPUTI (i když následná volání MQPUT selžou, když je atribut nastaven na tuto hodnotu).

Tato volba je platná pro všechny typy front, včetně distribučních seznamů a témat.

OOINQ
Otevřete objekt pro dotazování atributů.

Fronta, seznam názvů, definice procesu nebo správce front jsou otevřeny pro použití s následnými voláními MQINQ.

Tato volba je platná pro všechny typy objektů kromě rozdělovníků. Je-li *ODMN* název aliasu správce front, není platný. To platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro aliasy správce front je název lokálního správce front.

OOSET
Chcete-li nastavit atributy, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQSET.

Tato volba je platná pro všechny typy front kromě rozdělovníků. Není platné, pokud *ODMN* je název lokální definice vzdálené fronty; to platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro aliasy správce front je název lokálního správce front.

Volby vazby: Následující volby platí, když je otevíraný objekt frontou klastru; tyto volby řídí vazbu popisovače fronty na instanci fronty klastru:

OOBNDO
Svázat manipulátor s cílem při otevření fronty.

To způsobí, že lokální správce front při otevření fronty sváže manipulátor fronty s instancí cílové fronty. V důsledku toho jsou všechny zprávy vkládané pomocí tohoto popisovače odesílány do stejné instance cílové fronty a stejnou přenosovou cestou.

Tato volba je platná pouze pro fronty a ovlivňuje pouze fronty klastru. Je-li zadána pro frontu, která není frontou klastru, volba se ignoruje.

OOBNDN
Nepřipojujte se ke specifickému cíli.

Tímto se zastaví lokální správce front, který váže manipulátor fronty k instanci cílové fronty. V důsledku toho mohou následná volání MQPUT používající tento popisovač vést k odeslání zpráv do *různých* instancí cílové fronty nebo k jejich odeslání do stejné instance, ale různými trasami. Umožňuje také pozdější změnu vybrané instance lokálním správcem front, vzdáleným správcem front nebo agentem kanálu zpráv (MCA) v souladu s podmínkami sítě.

Poznámka: Klientské a serverové aplikace, které potřebují vyměnit *řadu* zpráv za účelem dokončení transakce, by neměly používat OOBNDN (nebo OOBNDQ, když *DefBind* má hodnotu

BNDNOT), protože následné zprávy v řadě mohou být odeslány různým instancím serverové aplikace.

Je-li pro frontu klastru zadána volba OOBRW nebo jedna z voleb OOOINP*, je správce front nucen vybrat lokální instanci fronty klastru. V důsledku toho je vazba popisovače fronty pevná, a to i v případě, že je zadán OOBNDN.

Pokud je OOINQ uveden s OOBNDN, následná volání MQINQ, která používají tento popisovač, mohou dotazovat různé instance fronty klastru, ačkoli obvykle mají všechny instance stejné hodnoty atributů.

OOBNDN je platné pouze pro fronty a ovlivňuje pouze fronty klastru. Je-li zadána pro frontu, která není frontou klastru, volba se ignoruje.

OOBNDQ
Použít výchozí vazbu pro frontu.

To způsobí, že lokální správce front sváže manipulátor fronty způsobem definovaným atributem fronty **DefBind** . Hodnota tohoto atributu je buď BNDOPN, nebo BNDNOT.

OOBNDQ je předvolba, pokud nejsou uvedeny OOBNDO a OOBNDN.

OOBNDQ je definován jako pomůcka pro dokumentaci programu. Není zamýšleno, aby tato volba byla použita s některou z dalších dvou voleb vazby, ale protože její hodnota je nula, nelze takové použití zjistit.

Volby kontextu: Následující volby řídí zpracování kontextu zprávy:

OOSAVA
Uložit kontext při načtení zprávy.

Informace o kontextu jsou přidruženy k tomuto popisovači fronty. Tato informace je nastavena z kontextu libovolné zprávy načtené pomocí tohoto popisovače. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Tyto kontextové informace lze předat do zprávy, která je později vložena do fronty pomocí volání MQPUT nebo MQPUT1 . Viz volby PMPASI a PMPASA popsané v části “MQPMO (Vložit-volby zpráv) na IBM i” na stránce 1168.

Dokud nebude zpráva úspěšně načtena, nelze kontext předat do zprávy vkládané do fronty.

Zpráva načtená pomocí jedné z voleb procházení GMBRW* nemá uložené informace o kontextu (ačkoli jsou pole kontextu v parametru **MSGDSC** nastavena po procházení).

Tato volba je platná pouze pro lokální, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou frontami. Musí být zadána jedna z voleb OOINP*.

OOPASI
Povolit předání kontextu identity.

To umožňuje, aby byla volba PMPASI uvedena v parametru **PMO** , když je zpráva vložena do fronty; to poskytuje zprávě informace o kontextu identity ze vstupní fronty, která byla otevřena s volbou OOSAVA. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Musí být zadána volba OOOUT.

Tato volba je platná pro všechny typy front, včetně rozdělovníků.

OOPASA
Povolit předání všech kontextů.

To umožňuje, aby byla volba PMPASA uvedena v parametru **PMO** , když je zpráva vložena do fronty; to dává zprávě informace o identitě a původu kontextu ze vstupní fronty, která byla otevřena s volbou OOSAVA. Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Tato volba znamená OOPASI, které proto nemusí být specifikovány. Musí být zadána volba OOOUT.

Tato volba je platná pro všechny typy front, včetně rozdělovníků.

OOSETI
Povolit nastavení kontextu identity.

To umožňuje zadání volby PMSETI v parametru **PMO** při vložení zprávy do fronty. Zpráva tak získá informace o kontextu identity obsažené v parametru **MSGDSC** zadaném ve volání MQPUT nebo MQPUT1 . Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Tato volba znamená OOPASI, které proto nemusí být specifikovány. Musí být zadána volba OOOUT.

Tato volba je platná pro všechny typy front, včetně rozdělovníků.

OOSETA
Povolit nastavení všech kontextů.

To umožňuje zadat volbu PMSETA v parametru **PMO** , když je zpráva vložena do fronty; to poskytuje zprávě informace o identitě a původu kontextu obsažené v parametru **MSGDSC** určeném ve volání MQPUT nebo MQPUT1 . Další informace o kontextu zprávy viz Kontext zprávy a Řízení informací o kontextu.

Tato volba zahrnuje následující volby, které proto nemusí být uvedeny:

- OOPASI
- OOPASA
- OOSETI

Musí být zadána volba OOOUT.

Tato volba je platná pro všechny typy front, včetně rozdělovníků.

Další volby: Následující volby řídí kontrolu autorizace a to, co se stane, když je správce front uveden do klidového stavu:

OOALTU-OOALTU
Ověřit s uvedeným identifikátorem uživatele.

To znamená, že pole *ODAU* v parametru **OBJDSC** obsahuje identifikátor uživatele, který má být použit k ověření tohoto volání MQOPEN. Volání může být úspěšné pouze v případě, že je tento produkt *ODAU* autorizován k otevření objektu s uvedenými volbami přístupu, bez ohledu na to, zda je k tomu autorizován identifikátor uživatele, pod kterým je aplikace spuštěna. To se však nevztahuje na žádné zadané kontextové volby, které jsou vždy kontrolovány proti identifikátoru uživatele, pod kterým je aplikace spuštěna.

Tato volba je platná pro všechny typy objektů.

OOFIQ
Selhání při uvedení správce front do klidového stavu.

Tato volba vynutí selhání volání MQOPEN, pokud je správce front ve stavu uvedení do klidového stavu.

Tato volba je platná pro všechny typy objektů.

OORLQ
Zadejte název lokální fronty, která byla otevřena.

Tato volba určuje, že má být zadána hodnota ResolvedQName ve struktuře MQOD (je-li k dispozici) s názvem lokální fronty, která byla otevřena. Název ResolvedQMgrbude podobně zadán s názvem lokálního správce front, který je hostitelem lokální fronty.

Tabulka 750. Platné volby MQOPEN pro každý typ fronty

Volba	Alias (“1” na stránce 1324)	Lokální a model	Vzdálený	Nelokální klastr	Distribuční seznam	Téma
OOINPQ	✓	✓	-	-	-	-

Tabulka 750. Platné volby MQOPEN pro každý typ fronty (pokračování)						
Volba	**Alias ("1" na stránce 1324)**	**Lokální a model**	**Vzdálený**	**Nelokální klastr**	**Distribuční seznam**	**Téma**
OOINPS	✓	✓	-	-	-	-
OOINPX	✓	✓	-	-	-	-
OOBRW	✓	✓	-	-	-	-
OOOUT	✓	✓	✓	✓	✓	✓
OOINQ	✓	✓	"2" na stránce 1324	✓	-	-
OOSET	✓	✓	"2" na stránce 1324	-	-	-
OOBNDO ("3" na stránce 1324)	✓	✓	✓	✓	✓	-
OOBNDN ("3" na stránce 1324)	✓	✓	✓	✓	✓	-
OOBNDQ ("3" na stránce 1324)	✓	✓	✓	✓	✓	-
OOSAVA	✓	✓	-	-	-	-
OOPASI	✓	✓	✓	✓	✓	"5" na stránce 1325
OOPASA	✓	✓	✓	✓	✓	"5" na stránce 1325
OOSETI	✓	✓	✓	✓	✓	"5" na stránce 1325
OOSETA	✓	✓	✓	✓	✓	"5" na stránce 1325
OOALTU-OOALTU	✓	✓	✓	✓	✓	✓
OOFIQ	✓	✓	✓	✓	✓	✓
OORLQ	✓	✓	✓	✓	-	-

Notes:

1. Platnost voleb pro aliasy závisí na platnosti volby pro frontu, do které se alias převádí.
2. Tato volba je platná pouze pro lokální definici vzdálené fronty.
3. Tuto volbu lze zadat pro libovolný typ fronty, ale ignoruje se, pokud fronta není frontou klastru.
4. Tento atribut je pro téma ignorován.

5. Tyto atributy lze použít s tématem, ale ovlivňují pouze kontext nastavený pro zachovanou zprávu, nikoli pole kontextu odeslaná libovolnému odběrateli.

HOBJ (10místné celé číslo se znaménkem)-výstup

Popisovač objektu.

Tento popisovač představuje přístup, který byl ustanoven k objektu. Musí být zadán při následných voláních fronty zpráv, která pracují s objektem. Přestává být platný, když je vydáno volání MQCLOSE nebo když je ukončena jednotka zpracování, která definuje rozsah popisovače.

Rozsah rukojeti je omezen na nejmenší jednotku paralelní zpracování podporované platformou, na které je aplikace spuštěna; manipulátor není platný mimo jednotku paralelního zpracování, ze které bylo vydáno volání MQOPEN:

- V systému IBM ije rozsahem manipulátoru úloha vydávající volání.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

Prohlášení o RPG

```
     C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     C                     CALLP     MQOPEN(HCONN : OBJDSC : OPTS :
     C                                      HOBJ : CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQOPEN           PR                  EXTPROC('MQOPEN')
D* Connection handle
D HCONN                          10I 0 VALUE
D* Object descriptor
D OBJDSC                        468A
D* Options that control the action of MQOPEN
D OPTS                           10I 0 VALUE
D* Object handle
D HOBJ                           10I 0
D* Completion code
D CMPCOD                         10I 0
D* Reason code qualifying CMPCOD
D REASON                         10I 0
```

IBM i MQPUT (Vložit zprávu) na IBM i

Volání MQPUT vloží zprávu do fronty, distribučního seznamu nebo do tématu. Fronta, distribuční seznam nebo téma již musí být otevřené.

Syntaxe

MQPUT *(HCONN, HOBJ, MSGDSC, PMO, BUFLEN, BUFFER, CMPCOD, REASON)*

Poznámky k použití

Témata

Pro použití témat platí následující poznámky:

1. Při použití příkazu MQPUT k publikování zpráv v tématu, kde jednomu nebo více odběratelům daného tématu nelze danou publikaci poskytnout kvůli problému s frontou odběratele (například je plná), je kód příčiny vrácený volání MQPUT a chování při doručení závislé na nastavení atributů PMSGDLV nebo NPMSGDLV v tématu TOPIC. Všimněte si, že doručení publikace do fronty nedoručených zpráv, když je uvedeno RODLQ, nebo vyřazení zprávy, když je uvedeno RODISC, je považováno za úspěšné doručení zprávy. Není-li doručeno žádné publikování, příkaz MQPUT se vrátí s kódem RC2502. K tomu může dojít v následujících případech:
   - Zpráva je publikována do TOPIC s PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavenou na ALL a jakýkoli odběr (trvalý nebo ne) má frontu, která nemůže přijmout publikování.
   - Zpráva je publikována do TOPIC s parametrem PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastaveným na ALLDUR a trvalý odběr má frontu, která nemůže přijmout publikování.

   Příkaz MQPUT se může vrátit s RCNONE, i když publikování nebylo možné doručit některým odběratelům v následujících případech:
   - Zpráva je publikována do TOPIC s PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavenou na ALLAVAIL a jakýkoli odběr, trvalý nebo ne, má frontu, která nemůže přijmout publikování.
   - Zpráva je publikována do TOPIC s parametrem PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastaveným na ALLDUR a netrvalý odběr má frontu, která nemůže přijmout publikování.
2. Pokud pro používané téma nejsou žádní odběratelé, publikovaná zpráva nebude odeslána do žádné fronty a bude vyřazena. Nezáleží na tom, zda je tato zpráva trvalá nebo dočasná, nebo zda má neomezenou dobu platnosti nebo nějaký malý čas vypršení platnosti, je stále vyřazena, pokud nejsou žádní odběratelé. Výjimkou je situace, kdy má být zpráva uchována. V takovém případě, přestože není odeslána do žádné fronty odběratelů, je uložena pro téma, které má být doručeno novým odběrům nebo odběratelům, kteří žádají o zachovaná publikování pomocí MQSUBRQ.

MQPUT a MQPUT1

K vložení zpráv do fronty lze použít volání MQPUT i MQPUT1 . Volání, které má být použito, závisí na okolnostech.

- Volání MQPUT by mělo být použito v případě, že má být do *stejné* fronty umístěno více zpráv.

  Nejprve je zadáno volání MQOPEN určující volbu OOOUT a poté jeden či více požadavků MQPUT na přidání zpráv do fronty. Nakonec je fronta uzavřena voláním MQCLOSE. To poskytuje lepší výkon než opakované použití volání MQPUT1 .

- Volání MQPUT1 by mělo být použito v případě, že má být do fronty vložena pouze *jedna* zpráva.

  Toto volání zapouzdřuje volání MQOPEN, MQPUT a MQCLOSE do jednoho volání a minimalizuje počet volání, která je třeba zadat.

Cílové fronty

Pokud aplikace vloží posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, pokud jsou splněny následující podmínky. Některé podmínky platí pro lokální i vzdálené cílové fronty; jiné podmínky platí pouze pro vzdálené cílové fronty.

Podmínky pro lokální a vzdálené cílové fronty

- Všechna volání MQPUT jsou v rámci stejné pracovní jednotky, nebo žádná z nich není v rámci pracovní jednotky.

  Jsou-li zprávy vkládány do konkrétní fronty v rámci jedné pracovní jednotky, mohou být zprávy z jiných aplikací prokládány s posloupností zpráv ve frontě.
- Všechna volání MQPUT se provádějí pomocí stejného popisovače objektu *HOBJ*.

  V některých prostředích je posloupnost zpráv zachována i při použití různých manipulátorů objektů za předpokladu, že jsou volání provedena ze stejné aplikace. Význam "stejné aplikace" je určen prostředím:

  – V systému IBM ije úlohou aplikace.
- Všechny zprávy mají stejnou prioritu.

Další podmínky pro vzdálené cílové fronty

- Existuje pouze jedna cesta od odesílajícího správce front k cílovému správci front.

  Pokud existuje možnost, že některé zprávy v posloupnosti mohou jít jinou cestou (například kvůli opětovné konfiguraci, vyvažování provozu nebo výběru cesty na základě velikosti zprávy), pořadí zpráv v cílovém správci front nelze zaručit.
- Zprávy nejsou dočasně umisťovány do front nedoručených zpráv ve správcích front pro odesílání, zprostředkující nebo cílové fronty.

  Pokud je jedna nebo více zpráv dočasně vloženo do fronty nedoručených zpráv (například proto, že přenosová fronta nebo cílová fronta jsou dočasně zaplněné), mohou zprávy do cílové fronty dorazit mimo pořadí.
- Zprávy jsou buď všechny trvalé, nebo všechny dočasné.

  Pokud má kanál na trase mezi odesílajícími a cílovými správci front nastaven atribut **CDNPM** na hodnotu NPFAST, mohou přechodné zprávy skákat před trvalými zprávami, což má za následek, že pořadí trvalých zpráv vzhledem k dočasným zprávám nebude zachováno. Pořadí vzájemně relativních trvalých zpráv a přechodných zpráv vzájemně relativních je však zachováno.

Nejsou-li tyto podmínky splněny, lze použít skupiny zpráv k zachování pořadí zpráv, ale mějte na paměti, že to vyžaduje, aby odesílající i přijímající aplikace používaly podporu seskupení zpráv. Další informace o skupinách zpráv viz:

- Pole *MDMFL* v deskriptoru MQMD
- Volba PMLOGO v MQPMO
- Volba GMLOGO v MQGMO

Distribuční seznamy

Pro použití distribučních seznamů platí následující poznámky.

1. Zprávy lze umístit do distribučního seznamu pomocí MQPMO version-1 nebo version-2 . Pokud se použije version-1 MQPMO (nebo version-2 MQPMO s hodnotou *PMREC* rovnou nule), aplikace nemůže poskytnout žádné záznamy vložených zpráv ani záznamy odpovědí. To znamená, že nebude možné identifikovat fronty, u kterých se vyskytnou chyby, pokud je zpráva úspěšně odeslána do některých front v rozdělovníku a ne do jiných.

Pokud aplikace poskytuje záznamy vložených zpráv nebo záznamy odpovědí, pole *PMVER* musí být nastaveno na hodnotu PMVER2.

Produkt version-2 MQPMO lze také použít k odesílání zpráv do jedné fronty, která není v distribučním seznamu, a to tak, že zajistíte, že *PMREC* bude nula.

2. Parametry kódu dokončení a kódu příčiny jsou nastaveny takto:
   - Pokud jsou všechny operace vložení do front v rozdělovníku úspěšné nebo selhávají stejným způsobem, jsou parametry kódu dokončení a kódu příčiny nastaveny tak, aby popisovaly společný výsledek. V tomto případě nejsou nastaveny záznamy odpovědí MQRR (jsou-li poskytovány aplikací).

     Pokud je například každé vložení úspěšné, kód dokončení je nastaven na CCOK a kód příčiny je RCNONE; pokud každé vložení selže, protože všechny fronty jsou zablokovány pro vložení, parametry jsou nastaveny na CCFAIL a RC2051.
   - Pokud vložení do front v rozdělovníku nejsou všechna úspěšná nebo selhávají stejným způsobem:
     - Parametr kódu dokončení je nastaven na hodnotu CCWARN, pokud bylo alespoň jedno vložení úspěšné, a na hodnotu CCFAIL, pokud se vše nezdařilo.
     - Parametr kódu příčiny je nastaven na hodnotu RC2136.
     - Záznamy odezvy (jsou-li poskytovány aplikací) jsou nastaveny na jednotlivé kódy dokončení a kódy příčiny pro fronty v rozdělovníku.

     Pokud vložení do místa určení selže, protože otevření pro toto místo určení se nezdařilo, pole v záznamu odpovědi jsou nastavena na CCFAIL a RC2137; toto místo určení je zahrnuto v souboru *PMIDC*.
3. Pokud se cíl v rozdělovníku interpretuje jako lokální fronta, zpráva se umístí do této fronty v normální formě (tj. ne jako zpráva rozdělovníku). Pokud se více než jeden cíl interpretuje na stejnou lokální frontu, umístí se do fronty jedna zpráva pro každý takový cíl.

   Pokud se cíl v rozdělovníku interpretuje jako vzdálená fronta, zpráva se umístí do příslušné přenosové fronty. Pokud se několik míst určení vyřeší do stejné přenosové fronty, může být do přenosové fronty umístěna jedna zpráva distribučního seznamu obsahující tato místa určení, a to i v případě, že tato místa určení nebyla sousedící se seznamem míst určení poskytnutým aplikací. To však lze provést pouze v případě, že přenosová fronta podporuje zprávy distribučního seznamu (viz **DistLists** atribut fronty popsaný v části “Atributy pro fronty” na stránce 1360).

   Pokud přenosová fronta nepodporuje distribuční seznamy, jedna kopie zprávy v normálním formátu se umístí do přenosové fronty pro každé místo určení, které používá tuto přenosovou frontu.

   Pokud je rozdělovník s daty zprávy aplikace příliš velký pro přenosovou frontu, zpráva rozdělovníku se rozdělí na menší zprávy rozdělovníku, z nichž každá obsahuje méně míst určení. Pokud se data zpráv aplikace vejdou pouze do fronty, nelze zprávy distribučního seznamu vůbec použít a správce front vygeneruje jednu kopii zprávy v normální podobě pro každé místo určení, které používá danou přenosovou frontu.

   Pokud mají různá místa určení jinou prioritu zpráv nebo perzistenci zpráv (to se může vyskytnout, když aplikace uvádí PRQDEF nebo PEQDEF), zprávy nejsou zadrženy ve stejné zprávě rozdělovníku. Místo toho správce front generuje tolik zpráv distribučního seznamu, kolik je potřeba pro přizpůsobení různých hodnot priority a perzistence.
4. Vložení do rozdělovníku může mít za následek:
   - jedna zpráva rozdělovníku nebo
   - Počet menších zpráv distribučního seznamu, nebo
   - směs zpráv distribučního seznamu a běžných zpráv, nebo
   - Pouze normální zprávy.

   Který z předchozích výskytů závisí na tom, zda:
   - Místa určení v seznamu jsou lokální, vzdálená nebo smíšená.
   - Místa určení mají stejnou prioritu zpráv a perzistenci zpráv.

- Přenosové fronty mohou obsahovat zprávy distribučního seznamu.
- Maximální délka zpráv přenosových front je dostatečně velká, aby se vešel do zprávy ve formě rozdělovníku.

Avšak bez ohledu na to, která z výše uvedených zpráv se vyskytne, každá *fyzická* výsledná zpráva (tj. každá normální zpráva nebo zpráva distribučního seznamu vyplývající z vložení) se počítá jako *jedna* zpráva, když:

- Kontrola, zda aplikace překročila povolený maximální počet zpráv v transakci (viz atribut správce front **MaxUncommittedMsgs**).
- Kontrola, zda jsou splněny podmínky spouštěče.
- Zvýšení hloubky fronty a kontrola, zda by byla překročena maximální hloubka fronty fronty.

5. Jakákoli změna definic front, která by způsobila, že by se manipulátor stal neplatným, kdyby byly fronty otevřeny jednotlivě (například změna v cestě řešení), nezpůsobí, že se manipulátor rozdělovníku stane neplatným. Při použití manipulátoru distribučního seznamu v následném volání MQPUT však dojde k selhání pro tuto konkrétní frontu.

Záhlaví

Pokud je zpráva vložena s jednou nebo více strukturami záhlaví IBM MQ na začátek dat zprávy aplikace, provede správce front určité kontroly struktur záhlaví, aby ověřil, zda jsou platné. Pokud správce front zjistí chybu, volání selže s příslušným kódem příčiny. Provedené kontroly se liší v závislosti na konkrétních strukturách, které jsou přítomny. Kromě toho se kontroly provádějí pouze v případě, že je pro volání MQPUT nebo MQPUT1 použita verze version-2 nebo novější; kontroly se neprovádějí, pokud je použita hodnota version-1 MQMD, a to ani v případě, že je při spuštění dat zprávy aplikace přítomno prostředí MQMDE.

Následující struktury záhlaví IBM MQ jsou zcela ověřeny správcem front: MQDH, MQMDE.

V případě jiných struktur záhlaví IBM MQ provádí správce front určité ověření, ale nekontroluje každé pole. Struktury, které nejsou podporovány lokálním správcem front, a struktury, které následují po prvním MQDLH ve zprávě, nejsou ověřeny.

Kromě obecných kontrol polí ve strukturách IBM MQ musí být splněny následující podmínky:

- Struktura IBM MQ nesmí být rozdělena na dva nebo více segmentů-struktura musí být zcela obsažena v jednom segmentu.
- Součet délek struktur ve zprávě PCF se musí rovnat délce určené parametrem **BUFLEN** ve volání MQPUT nebo MQPUT1 . Zpráva PCF je zpráva, která má jeden z následujících názvů formátů:
  - FMADMN
  - FMEVNT
  - Prostředek FMPCF
- Struktury IBM MQ nesmí být zkráceny, s výjimkou následujících situací, kdy jsou zkrácené struktury povoleny:
  - Zprávy, které jsou zprávami sestavy.
  - Zprávy PCF.
  - Zprávy obsahující strukturu MQDLH. (Struktury *následující* první MQDLH lze zkrátit; struktury předcházející MQDLH nikoli.)

Vyrovnávací paměť

Parametr **BUFFER** zobrazený v programovacím příkladu RPG je deklarován jako řetězec; to omezuje maximální délku parametru na 256 bajtů. Je-li vyžadována větší vyrovnávací paměť, měl by být parametr deklarován jako struktura nebo jako pole ve fyzickém souboru. Tím se zvýší maximální možná délka na přibližně 32 kB.

Parametry

Volání MQPUT má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOBJ (10místné celé číslo se znaménkem)-vstup

Popisovač objektu.

Tento popisovač představuje frontu, do které je zpráva přidána, nebo téma, do kterého je zpráva publikována. Hodnota *HOBJ* byla vrácena předchozím voláním MQOPEN, které určilo volbu OOOOUT.

MSGDSC (MQMD)-vstupní/výstupní

Deskriptor zprávy.

Tato struktura popisuje atributy odesílané zprávy a přijímá informace o zprávě po dokončení požadavku na vložení. Podrobnosti viz "MQMD (deskriptor zprávy) na IBM i" na stránce 1105.

Pokud aplikace poskytuje databázi MQMD version-1 , data zprávy mohou mít předponu ve struktuře MQMDE, aby bylo možné zadat hodnoty pro pole, která existují v deskriptoru version-2 MQMD, ale nikoli version-1. Pole *MDFMT* v deskriptoru MQMD musí být nastaveno na hodnotu FMMDE, aby se označila přítomnost MQMDE. Další informace viz část "MQMDE (rozšíření deskriptoru zpráv) na IBM i" na stránce 1148.

PMO (MQPMO)-vstup/výstup

Volby, které řídí akci MQPUT.

Podrobnosti viz "MQPMO (Vložit-volby zpráv) na IBM i" na stránce 1168.

BUFLEN (10místné celé číslo se znaménkem)-vstup

Délka zprávy v souboru *BUFFER*.

Nula je platná a označuje, že zpráva neobsahuje žádná data aplikace. Horní limit pro *BUFLEN* závisí na různých faktorech:

- Pokud je cílovou frontou sdílená fronta, horní limit je 63 kB (64 512 bajtů).
- Pokud je cíl lokální frontou nebo je přeložen na lokální frontu (ale nejedná se o sdílenou frontu), závisí horní limit na tom, zda:
  - Lokální správce front podporuje segmentaci.
  - Odesílající aplikace určuje příznak, který umožňuje správci front segmentovat zprávu. Tento příznak je MFSEGA a lze jej zadat buď v produktu version-2 MQMD, nebo v prostředí MQMDE používaném s produktem version-1 MQMD.

  Jsou-li splněny obě tyto podmínky, nesmí hodnota *BUFLEN* překročit 999 999 999 999 minus hodnota pole *MDOFF* v deskriptoru MQMD. Nejdelší logická zpráva, kterou lze vložit, je proto 999 999 999 bajtů (když je *MDOFF* nula). Omezení prostředků stanovená operačním systémem nebo prostředím, ve kterém je aplikace spuštěna, však mohou mít za následek nižší limit.

  Pokud není splněna jedna nebo obě výše popsané podmínky, nesmí hodnota *BUFLEN* překročit menší hodnotu atributu **MaxMsgLength** fronty a atributu **MaxMsgLength** správce front.
- Je-li cílem vzdálená fronta nebo je-li přeložena do vzdálené fronty, platí podmínky pro lokální fronty, *ale v každém správci front, jehož prostřednictvím musí zpráva projít, aby se dostala do cílové fronty* ; zejména:
  1. Lokální přenosová fronta používaná k dočasnému uložení zprávy v lokálním správci front
  2. Intermediační přenosové fronty (pokud existují) používané k ukládání zpráv ve správcích front na trase mezi lokálními a cílovými správci front

3. Cílová fronta ve správci cílové fronty

Nejdelší zpráva, kterou lze vložit, je proto řízena nejrestriktivnějším z těchto front a správců front.

Když je zpráva v přenosové frontě, další informace jsou uloženy spolu s daty zprávy, což snižuje množství dat aplikace, která lze přenášet. V této situaci se doporučuje při určování limitu pro *BUFLEN*odečíst bajty LNMHD od hodnot *MaxMsgLength* přenosových front.

Poznámka: Při vložení zprávy lze diagnostikovat synchronně (s kódem příčiny RC2030 nebo RC2031) pouze selhání splnění podmínky 1. Nejsou-li splněny podmínky 2 nebo 3, je zpráva přesměrována do fronty nedoručených zpráv (nedoručených zpráv), a to buď do intermediačního správce front, nebo do cílového správce front. Pokud k tomu dojde, vygeneruje se zpráva sestavy, pokud ji odesilatel požadoval.

BUFFER (1bajtový bitový řetězec x BUFLEN)-vstup

Data zprávy.

Jedná se o vyrovnávací paměť obsahující data aplikace, která mají být odeslána. Vyrovnávací paměť by měla být zarovnána na hranici odpovídající povaze dat ve zprávě. 4bajtové zarovnání by mělo být vhodné pro většinu zpráv (včetně zpráv obsahujících struktury záhlaví MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8bajtové zarovnání.

Pokud *BUFFER* obsahuje znaková data, číselná data nebo obojí, pole *MDCSI* a *MDENC* v parametru **MSGDSC** by měla být nastavena na hodnoty odpovídající datům; to umožní příjemci zprávy převést data (v případě potřeby) na znakovou sadu a kódování používané příjemcem.

Poznámka: Všechny ostatní parametry volání MQPUT musí být ve znakové sadě zadané atributem správce front **CodedCharSetId** a v kódování lokálního správce front dané tabulkou ENNAT.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCWARN:

RC2104
(2104, X'838 ') Volba sestavy v deskriptoru zprávy nebyla rozpoznána.

RC2136
(2136, X'858 ') Bylo vráceno více kódů příčiny.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2004
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

RC2005
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2013
(2013, X'7DD') Doba vypršení platnosti není platná.

RC2014
(2014, X'7DE') Kód zpětné vazby není platný.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2019
(2019, X'7E3') popisovač objektu není platný.

RC2024
(2024, X'7E8') V rámci aktuální pracovní jednotky nelze zpracovat žádné další zprávy.

RC2026
(2026, X'7EA') Deskriptor zpráv není platný.

RC2027
(2027, X'7EB') Chybí fronta pro odpověď.

RC2029
(2029, X'7ED') Typ zprávy v deskriptoru zprávy není platný.

RC2030
(2030, X'7EE') Délka zprávy je větší než maximum pro frontu.

RC2031
(2031, X'7EF') Délka zprávy pro správce front je větší než maximum.

RC2039
(2039, X'7F7') Fronta není otevřena pro výstup.

RC2041
(2041, X'7F9') Definice objektu se od doby otevření změnila.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2047
(2047, X'7FF') Perzistence není platná.

RC2048
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

RC2050
(2050, X'802 ') Priorita zprávy není platná.

RC2051
(2051, X'803 ') Volání Put bylo pro frontu zablokováno.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2053
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

RC2056
(2056, X'808 ') Není k dispozici místo na disku pro frontu.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2061
(2061, X'80D') Volby sestavy v deskriptoru zprávy jsou neplatné.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2072
(2072, X'818 ') Podpora synchronizační funkce není k dispozici.

RC2093
(2093, X'82D') Fronta není otevřena pro průchod celého kontextu.

RC2094
(2094, X'82E') Fronta není otevřena pro průchozí kontext identity.

RC2095
(2095, X'82F') Fronta není otevřena pro nastavení celého kontextu.

RC2096
(2096, X'830 ') Fronta není otevřena pro nastavení kontextu identity.

RC2097
(2097, X'831 ') popisovač fronty, na který se odkazuje, neuloží kontext.

RC2098
(2098, X'832 ') Kontext není k dispozici pro popisovač fronty, na který se odkazuje.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2135
(2135, X'857 ') Struktura záhlaví distribuce není platná.

RC2136
(2136, X'858 ') Bylo vráceno více kódů příčiny.

RC2137
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

RC2149
(2149, X'865 ') Struktury PCF nejsou platné.

RC2154
(2154, X'86A') Počet přítomných záznamů není platný.

RC2156
(2156, X'86C') Záznamy odpovědí nejsou platné.

RC2158
(2158, X'86E') Příznaky záznamu vložení zprávy jsou neplatné.

RC2159
(2159, X'86F') Záznamy vkládající zprávy jsou neplatné.

RC2161
(2161, X'871 ') Správce front je uveden do klidového stavu.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2173
(2173, X'87D') Struktura voleb vkládání zpráv je neplatná.

RC2185
(2185, X'889 ') Nekonzistentní specifikace perzistence.

RC2188
(2188, X'88C') Volání bylo odmítnuto uživatelskou procedurou pracovní zátěže klastru.

RC2189
(2189, X'88D') Nezdařilo se rozpoznání názvu klastru.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2241
(2241, X'8C1') Skupina zpráv není dokončena.

RC2242
(2242, X'8C2') Logická zpráva není dokončena.

RC2245
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

RC2248
(2248, X'8C8') Rozšíření deskriptoru zprávy není platné.

RC2249
(2249, X'8C9') Příznaky zprávy nejsou platné.

RC2250
(2250, X'8CA') Pořadové číslo zprávy není platné.

RC2251
(2251, X'8CB') Offset segmentu zprávy není platný.

RC2252
(2252, X'8CC') Původní délka není platná.

RC2253
(2253, X'8CD') Délka dat v segmentu zprávy je nula.

RC2255
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

RC2257
(2257, X'8D1') Dodána nesprávná verze MQMD.

RC2258
(2258, X'8D2') Identifikátor skupiny není platný.

RC2266
(2266, X'8DA') Selhala uživatelská procedura pracovní zátěže klastru.

RC2269
(2269, X'8DD') Chyba prostředku klastru.

RC2270
(2270, X'8DE') Nejsou k dispozici žádné cílové fronty.

RC2420
(2420) Bylo vydáno volání MQPUT, ale data zprávy obsahují neplatnou strukturu MQEPH.

RC2479
(2479, X'9AF') Publikace nebyla uchována.

RC2480
(2480, X'9B0') Typ cíle se změnil: alias fronty odkazovalo na frontu, ale nyní odkazuje na téma.

RC2502
(2502, X'9C6') Publikování se nezdařilo a publikování nebylo doručeno žádnému odběrateli.

RC2551
(2551, X'9F7') Určený řetězec výběru není k dispozici.

RC2554
(2554, X'9FA') Obsah zprávy nelze analyzovat, aby se zjistilo, zda má být zpráva doručena odběrateli s rozšířeným selektorem zpráv.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQPUT(HCONN : HOBJ : MSGDSC : PMO :
C                                     BUFLEN : BUFFER : CMPCOD :
C                                     REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQPUT            PR                  EXTPROC('MQPUT')
D* Connection handle
D HCONN                          10I 0 VALUE
D* Object handle
D HOBJ                           10I 0 VALUE
D* Message descriptor
D MSGDSC                        364A
D* Options that control the action of MQPUT
D PMO                           200A
D* Length of the message in Buffer
D BUFLEN                         10I 0 VALUE
D* Message data
D BUFFER                           *   VALUE
D* Completion code
D CMPCOD                         10I 0
D* Reason code qualifying CMPCOD
D REASON                         10I 0
```

IBM i MQPUT1 (Vložit jednu zprávu) na IBM i

Volání MQPUT1 vloží jednu zprávu do fronty nebo distribučního seznamu nebo do tématu. Fronta, distribuční seznam nebo téma nemusí být otevřené.

Syntaxe

MQPUT1 *(HCONN, OBJDSC, MSGDSC, PMO, BUFLEN, BUFFER, CMPCOD, REASON)*

Poznámky k použití

1. K vložení zpráv do fronty lze použít volání MQPUT i MQPUT1 . Volání, které má být použito, závisí na okolnostech:
   - Volání MQPUT by mělo být použito v případě, že má být do *stejné* fronty umístěno více zpráv.

     Nejprve je zadáno volání MQOPEN určující volbu OOOUT a poté jeden či více požadavků MQPUT na přidání zpráv do fronty. Nakonec je fronta uzavřena voláním MQCLOSE. To poskytuje lepší výkon než opakované použití volání MQPUT1 .
   - Volání MQPUT1 by mělo být použito v případě, že má být do fronty vložena pouze *jedna* zpráva.

     Toto volání zapouzdřuje volání MQOPEN, MQPUT a MQCLOSE do jednoho volání a minimalizuje počet volání, která je třeba zadat.
2. Pokud aplikace vloží posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, pokud jsou splněny určité podmínky. Ve většině prostředí však volání MQPUT1 tyto podmínky nesplňuje, a proto nezachovává pořadí zpráv. V těchto prostředích musí být místo toho použito volání MQPUT. Podrobnosti naleznete v poznámkách k použití v popisu volání MQPUT.
3. Volání MQPUT1 lze použít k vložení zpráv do distribučních seznamů. Obecné informace o tomto tématu naleznete v poznámkách k použití pro volání MQOPEN a MQPUT.

   Při použití volání MQPUT1 platí následující rozdíly:

   a. Pokud jsou záznamy odpovědí MQRR poskytovány aplikací, musí být poskytnuty pomocí struktury MQOD; nemohou být poskytnuty pomocí struktury MQPMO.

b. Kód příčiny RC2137 není nikdy vrácen produktem MQPUT1 v záznamech odezvy; pokud se fronta neotevře, záznam odezvy pro tuto frontu obsahuje skutečný kód příčiny, který je výsledkem operace otevření.

   Pokud operace otevření pro frontu uspěje s kódem dokončení CCWARN, kód dokončení a kód příčiny v záznamu odezvy pro tuto frontu jsou nahrazeny kódy dokončení a příčiny, které jsou výsledkem operace vložení.

   Stejně jako u volání MQOPEN a MQPUT nastavuje správce front záznamy odpovědí (jsou-li poskytnuty) pouze v případě, že výsledek volání není stejný pro všechny fronty v distribučním seznamu; to je označeno dokončením volání s kódem příčiny RC2136.

4. Pokud se k vložení zprávy do fronty klastru používá volání MQPUT1 , volání se chová, jako by bylo ve volání MQOPEN zadáno OOBNDN.
5. Pokud je zpráva vložena s jednou nebo více strukturami záhlaví IBM MQ na začátek dat zprávy aplikace, provede správce front určité kontroly struktur záhlaví, aby ověřil, zda jsou platné. Další informace naleznete v poznámkách k použití pro volání MQPUT.
6. Pokud se vyskytne více než jedna z varovných situací (viz parametr **CMPCOD**), vrácený kód příčiny je *první* v následujícím seznamu, který platí:
   a. RC2136
   b. RC2242
   c. RC2241
   d. RC2049 nebo RC2104
7. Parametr **BUFFER** zobrazený v programovacím příkladu RPG je deklarován jako řetězec; to omezuje maximální délku parametru na 256 bajtů. Je-li vyžadována větší vyrovnávací paměť, měl by být parametr deklarován jako struktura nebo jako pole ve fyzickém souboru. Tím se zvýší maximální možná délka na přibližně 32 kB.

Parametry

Volání MQPUT1 má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

OBJDSC (MQOD)-vstupní/výstupní

Deskriptor objektu.

Jedná se o strukturu, která identifikuje frontu, do které je zpráva přidána. Podrobnosti viz “MQOD (Object descriptor) na systému IBM i” na stránce 1154.

Uživatel musí mít oprávnění k otevření fronty pro výstup. Fronta **nesmí** být modelovou frontou.

MSGDSC (MQMD)-vstupní/výstupní

Deskriptor zprávy.

Tato struktura popisuje atributy odesílané zprávy a po dokončení požadavku na vložení přijímá informace o zpětné vazbě. Podrobnosti viz “MQMD (deskriptor zprávy) na IBM i” na stránce 1105.

Pokud aplikace poskytuje databázi MQMD version-1 , data zprávy mohou mít předponu ve struktuře MQMDE, aby bylo možné zadat hodnoty pro pole, která existují v deskriptoru version-2 MQMD, ale nikoli version-1. Pole *MDFMT* v deskriptoru MQMD musí být nastaveno na hodnotu FMMDE, aby se označila přítomnost MQMDE. Další informace viz část “MQMDE (rozšíření deskriptoru zpráv) na IBM i” na stránce 1148.

PMO (MQPMO)-vstup/výstup

Volby, které řídí akci MQPUT1.

Podrobnosti viz “MQPMO (Vložit-volby zpráv) na IBM i” na stránce 1168.

BUFLEN (10místné celé číslo se znaménkem)-vstup

Délka zprávy v souboru *BUFFER*.

Nula je platná a označuje, že zpráva neobsahuje žádná data aplikace. Horní limit závisí na různých faktorech. Další podrobnosti naleznete v popisu parametru **BUFLEN** volání MQPUT.

BUFFER (1bajtový bitový řetězec x BUFLEN)-vstup

Data zprávy.

Jedná se o vyrovnávací paměť obsahující data zprávy aplikace, která mají být odeslána. Vyrovnávací paměť by měla být zarovnána na hranici odpovídající povaze dat ve zprávě. 4bajtové zarovnání by mělo být vhodné pro většinu zpráv (včetně zpráv obsahujících struktury záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8bajtové zarovnání.

Pokud *BUFFER* obsahuje znaková data, číselná data nebo obojí, pole *MDCSI* a *MDENC* v parametru **MSGDSC** by měla být nastavena na hodnoty odpovídající datům; to umožní příjemci zprávy převést data (v případě potřeby) na znakovou sadu a kódování používané příjemcem.

Poznámka: Všechny ostatní parametry volání MQPUT1 musí být ve znakové sadě zadané atributem správce front **CodedCharSetId** a v kódování lokálního správce front daném ENNAT.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCWARN:

RC2104
(2104, X'838 ') Volba sestavy v deskriptoru zprávy nebyla rozpoznána.

RC2136
(2136, X'858 ') Bylo vráceno více kódů příčiny.

RC2049
(2049, X'801 ') Priorita zprávy překračuje maximální podporovanou hodnotu.

RC2241
(2241, X'8C1') Skupina zpráv není dokončena.

RC2242
(2242, X'8C2') Logická zpráva není dokončena.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2001
(2001, X'7D1') Základní fronta aliasů není platný typ.

RC2004
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

RC2005
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2013
(2013, X'7DD') Doba vypršení platnosti není platná.

RC2014
(2014, X'7DE') Kód zpětné vazby není platný.

RC2017
(2017, X'7E1') Nejsou k dispozici žádné další popisovače.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2024
(2024, X'7E8') V rámci aktuální pracovní jednotky nelze zpracovat žádné další zprávy.

RC2026
(2026, X'7EA') Deskriptor zpráv není platný.

RC2027
(2027, X'7EB') Chybí fronta pro odpověď.

RC2029
(2029, X'7ED') Typ zprávy v deskriptoru zprávy není platný.

RC2030
(2030, X'7EE') Délka zprávy je větší než maximum pro frontu.

RC2031
(2031, X'7EF') Délka zprávy pro správce front je větší než maximum.

RC2035
(2035, X'7F3') Není autorizováno pro přístup.

RC2042
(2042, X'7FA') Objekt je již otevřen s konfliktními volbami.

RC2043
(2043, X'7FB') Typ objektu není platný.

RC2044
(2044, X'7FC') Struktura deskriptoru objektu není platná.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2047
(2047, X'7FF') Perzistence není platná.

RC2048
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

RC2050
(2050, X'802 ') Priorita zprávy není platná.

RC2051
(2051, X'803 ') Volání Put bylo pro frontu zablokováno.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2053
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

RC2056
(2056, X'808 ') Není k dispozici místo na disku pro frontu.

RC2057
(2057, X'809 ') Typ fronty není platný.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2061
(2061, X'80D') Volby sestavy v deskriptoru zprávy jsou neplatné.

RC2063
(2063, X'80F') Došlo k chybě zabezpečení.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2072
(2072, X'818 ') Podpora synchronizační funkce není k dispozici.

RC2082
(2082, X'822 ') Neznámý alias základní fronty.

RC2085
(2085, X'825 ') Neznámý název objektu.

RC2086
(2086, X'826 ') Neznámý správce front objektů.

RC2087
(2087, X'827 ') Neznámý vzdálený správce front.

RC2091
(2091, X'82B') Přenosová fronta není lokální.

RC2092
(2092, X'82C') Přenosová fronta s chybným použitím.

RC2097
(2097, X'831 ') popisovač fronty, na který se odkazuje, neuloží kontext.

RC2098
(2098, X'832 ') Kontext není k dispozici pro popisovač fronty, na který se odkazuje.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2135
(2135, X'857 ') Struktura záhlaví distribuce není platná.

RC2136
(2136, X'858 ') Bylo vráceno více kódů příčiny.

RC2149
(2149, X'865 ') Struktury PCF nejsou platné.

RC2154
(2154, X'86A') Počet přítomných záznamů není platný.

RC2155
(2155, X'86B') Záznamy objektů nejsou platné.

RC2156
(2156, X'86C') Záznamy odpovědí nejsou platné.

RC2158
(2158, X'86E') Příznaky záznamu vložení zprávy jsou neplatné.

RC2159
(2159, X'86F') Záznamy vkládající zprávy jsou neplatné.

RC2161
(2161, X'871 ') Správce front je uveden do klidového stavu.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2173
(2173, X'87D') Struktura voleb vkládání zpráv je neplatná.

RC2184
(2184, X'888 ') Název vzdálené fronty není platný.

RC2188
(2188, X'88C') Volání bylo odmítnuto uživatelskou procedurou pracovní zátěže klastru.

RC2189
(2189, X'88D') Nezdařilo se rozpoznání názvu klastru.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2196
(2196, X'894 ') Neznámá přenosová fronta.

RC2197
(2197, X'895 ') Neznámá výchozí přenosová fronta.

RC2198
(2198, X'896 ') Výchozí přenosová fronta není lokální.

RC2199
(2199, X'897 ') Výchozí chyba použití přenosové fronty.

RC2258
(2258, X'8D2') Identifikátor skupiny není platný.

RC2248
(2248, X'8C8') Rozšíření deskriptoru zprávy není platné.

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2249
(2249, X'8C9') Příznaky zprávy nejsou platné.

RC2250
(2250, X'8CA') Pořadové číslo zprávy není platné.

RC2251
(2251, X'8CB') Offset segmentu zprávy není platný.

RC2252
(2252, X'8CC') Původní délka není platná.

RC2253
(2253, X'8CD') Délka dat v segmentu zprávy je nula.

RC2255
(2255, X'8CF') Pracovní jednotka není k dispozici pro použití správcem front.

RC2257
(2257, X'8D1') Dodána nesprávná verze MQMD.

RC2266
(2266, X'8DA') Selhala uživatelská procedura pracovní zátěže klastru.

RC2269
(2269, X'8DD') Chyba prostředku klastru.

RC2270
(2270, X'8DE') Nejsou k dispozici žádné cílové fronty.

RC2420
(2420) Bylo vydáno volání MQPUT1 , ale data zprávy obsahují neplatnou strukturu MQEPH.

RC2551
(2551, X'9F7') Určený řetězec výběru není k dispozici.

RC2554
(2554, X'9FA') Obsah zprávy nelze analyzovat, aby se zjistilo, zda má být zpráva doručena odběrateli s rozšířeným selektorem zpráv.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQPUT1(HCONN : OBJDSC : MSGDSC :
C                                      PMO : BUFLEN : BUFFER :
C                                      CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQPUT1           PR                  EXTPROC('MQPUT1')
D* Connection handle
D HCONN                          10I 0 VALUE
D* Object descriptor
D OBJDSC                        468A
D* Message descriptor
D MSGDSC                        364A
D* Options that control the action of MQPUT1
D PMO                           200A
D* Length of the message in BUFFER
D BUFLEN                         10I 0 VALUE
D* Message data
D BUFFER                           *   VALUE
D* Completion code
D CMPCOD                         10I 0
D* Reason code qualifying CMPCOD
D REASON                         10I 0
```

IBM i MQSET (Nastavení atributů objektu) na IBM i

Volání MQSET se používá ke změně atributů objektu reprezentovaného popisovačem. Objekt musí být frontou.

Syntaxe

MQSET *(HCONN, HOBJ, SELCNT, SELS, IACNT, INTATR, CALEN, CHRATR, CMPCOD, REASON)*

Poznámky k použití

1. Pomocí tohoto volání může aplikace určit pole celočíselných atributů, kolekci řetězců znakových atributů nebo obojí. Pokud se nevyskytují žádné chyby, uvedené atributy jsou nastaveny současně. Dojde-li k chybě (například pokud selektor není platný nebo dojde-li k pokusu o nastavení atributu na neplatnou hodnotu), volání selže a nejsou nastaveny žádné atributy.
2. Hodnoty atributů lze určit pomocí volání MQINQ; Podrobnosti viz "MQINQ (dotazovat se na atributy objektu) na IBM i" na stránce 1298 .

Poznámka: Ne všechny atributy s hodnotami, které lze požadovat při použití volání MQINQ, mohou mít své hodnoty změněny pomocí volání MQSET. S tímto voláním nelze například nastavit žádné atributy objektu procesu nebo správce front.

3. Změny atributů jsou zachovány v rámci restartů správce front (kromě změn dočasných dynamických front, které nepřežívají restartování správce front).
4. Atributy modelové fronty nelze změnit pomocí volání MQSET. Pokud však otevřete modelovou frontu pomocí volání MQOPEN s volbou MQOO_SET, můžete pomocí volání MQSET nastavit atributy dynamické lokální fronty vytvořené voláním MQOPEN.
5. Pokud je nastavovaný objekt frontou klastru, musí existovat lokální instance fronty klastru, aby bylo otevření úspěšné.

Další informace o atributech objektu viz:

- “Atributy pro fronty” na stránce 1360
- “Atributy pro seznamy názvů” na stránce 1388
- “Atributy pro definice procesů v systému IBM i” na stránce 1389
- “Atributy pro správce front v systému IBM i” na stránce 1391

Parametry

Volání MQSET má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota HCONN byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOBJ (10místné celé číslo se znaménkem)-vstup

Popisovač objektu.

Tento popisovač představuje objekt fronty s atributy, které mají být nastaveny. Manipulátor byl vrácen předchozím voláním MQOPEN, které určilo volbu OOSET.

SELCNT (10místné celé číslo se znaménkem)-vstup

Počet selektorů.

Jedná se o počet selektorů, které jsou dodány v poli SELS . Jedná se o počet atributů, které mají být nastaveny. Nula je platná hodnota. Maximální povolený počet je 256.

SELS (10místné celé číslo se znaménkem x SELCNT)-vstup

Pole selektorů atributů.

Toto je pole selektorů atributů **SELCNT** ; každý selektor identifikuje atribut (celé číslo nebo znak) s hodnotou, která má být nastavena.

Každý selektor musí být platný pro typ fronty, který HOBJ představuje. Povoleny jsou pouze určité hodnoty IA* a CA*; tyto hodnoty jsou uvedeny dále v této části.

Selektory lze zadat v libovolném pořadí. Hodnoty atributů, které odpovídají selektorům celočíselných atributů (selektory IA*), musí být uvedeny v souboru INTATR ve stejném pořadí, v jakém se tyto selektory vyskytují v souboru SELS. Hodnoty atributů, které odpovídají selektorům znakových atributů (selektory CA*), musí být uvedeny v souboru CHRATR ve stejném pořadí, v jakém se tyto selektory vyskytují. Selektory IA* mohou být prokládány se selektory CA*; důležité je pouze relativní pořadí v rámci každého typu.

Není chybou zadat stejný selektor více než jednou; pokud se tak stane, poslední hodnota zadaná pro konkrétní selektor je ta, která se projeví.

Poznámka:

1. Selektory celočíselných a znakových atributů jsou přiděleny ve dvou různých rozsazích; selektory IA* jsou umístěny v rozsahu IAFRST až IALAST a selektory CA* v rozsahu CAFRST až CALAST.

   Pro každý rozsah konstanty IALSTU a CALSTU definují nejvyšší hodnotu, kterou bude správce front přijímat.
2. Pokud se nejprve vyskytnou všechny selektory IA*, lze k adresování odpovídajících prvků v polích SELS a INTATR použít stejná čísla prvků.

Atributy, které lze nastavit, jsou uvedeny v následující tabulce. Pomocí tohoto volání nelze nastavit žádné další atributy. Pro selektory atributů CA* je v závorkách uvedena konstanta, která definuje délku řetězce, který je požadován v souboru CHRATR , v bajtech.

Tabulka 751. Selektory atributů MQSET pro fronty

Selektor	Popis	Poznámka
CATRGD	Data spouštěče (LNTRGD).	“2” na stránce 1343
IADIST	Podpora distribučního seznamu.	“1” na stránce 1343
IAIGET	Zda jsou povoleny operace získání.	
IAIPUT	Zda jsou povoleny operace vložení.	
IATRGC-změna pořadí	Ovládací prvek spouštěče.	“2” na stránce 1343
IATRGD	Hloubka spouątěče.	“2” na stránce 1343
IATRGP-řízení	Priorita zprávy prahové hodnoty pro spouątěče.	“2” na stránce 1343
IATRGT-další	Typ spouątěče.	“2” na stránce 1343

Notes:

1. Podporováno pouze na následujících platformách:
   - AIX
   - IBM i
   - Windows

   a pro klienty IBM MQ připojené k těmto systémům.
2. Není podporováno na systému VSE/ESA.

IACNT (celé číslo se znaménkem 10 číslic)-vstup

Počet celočíselných atributů.

Jedná se o počet prvků v poli INTATR a musí se jednat alespoň o počet selektorů IA* v parametru **SELS** . Nula je platná hodnota, pokud žádná není.

INTATR (10místné znaménko integ x rxIACNT)-vstup

Pole celočíselných atributů.

Toto je pole celočíselných hodnot atributu IACNT . Tyto hodnoty atributů musí být ve stejném pořadí jako selektory IA* v poli SELS .

CALEN (10místné celé číslo se znaménkem)-vstup

Délka vyrovnávací paměti atributů znaků.

Jedná se o délku parametru **CHRATR** v bajtech, která musí být alespoň součtem délek znakových atributů uvedených v poli SELS . Nula je platná hodnota, pokud v souboru SELSnejsou žádné selektory CA*.

CHRATR (1bajtový znakový řetězec x CALEN)-vstup

Znakové atributy.

Jedná se o vyrovnávací paměť obsahující hodnoty znakového atributu, zřetězenou dohromady. Délka vyrovnávací paměti je dána parametrem **CALEN** .

Atributy znaků musí být zadány ve stejném pořadí jako selektory CA* v poli SELS . Délka každého znakového atributu je pevná (viz SELS). Pokud hodnota, která má být nastavena pro atribut, obsahuje méně neprázdných znaků, než je definovaná délka atributu, hodnota v souboru CHRATR musí být vyplněna vpravo mezerami, aby se hodnota atributu shodovala s definovanou délkou atributu.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující CMPCOD.

Pokud je CMPCOD CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr CMPCOD hodnotu CCFAIL:

RC2219
(2219, X'8AB') Volání MQI bylo znovu zadáno před dokončením předchozího volání.

RC2006
(2006, X'7D6') Délka znakových atributů není platná.

RC2007
(2007, X'7D7') Řetězec znakových atributů není platný.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2019
(2019, X'7E3') popisovač objektu není platný.

RC2020
(2020, X'7E4') Hodnota pro atribut fronty inhibit-get nebo inhibit-put není platná.

RC2021
(2021, X'7E5') Počet celočíselných atributů není platný.

RC2023
(2023, X'7E7') Pole atributů Integer je neplatné.

RC2040
(2040, X'7F8') Fronta není otevřena pro nastavení.

RC2041
(2041, X'7F9') Definice objektu se od doby otevření změnila.

RC2101
(2101, X'835 ') Objekt poškozen.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2058
(2058, X'80A') Název správce front je neplatný nebo neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2162
(2162, X'872 ') Probíhá ukončování činnosti správce front.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2065
(2065, X'811 ') Počet selektorů není platný.

RC2067
(2067, X'813 ') Selektor atributů není platný.

RC2066
(2066, X'812 ') Počet selektorů je příliš velký.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2075
(2075, X'81B') Hodnota pro atribut řízení spouštěče není platná.

RC2076
(2076, X'81C') Hodnota pro atribut hloubky spouštěče není platná.

RC2077
(2077, X'81D') Hodnota pro atribut trigger-message-priority není platná.

RC2078
(2078, X'81E') Hodnota pro atribut typu spouštěče není platná.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                    CALLP     MQSET(HCONN : HOBJ : SELCNT :
C                                     SELS(1) : IACNT : INTATR(1) :
C                                     CALEN : CHRATR : CMPCOD :
C                                     REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQSET            PR                  EXTPROC('MQSET')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Object handle
D HOBJ                          10I 0 VALUE
D* Count of selectors
D SELCNT                        10I 0 VALUE
D* Array of attribute selectors
D SELS                          10I 0
D* Count of integer attributes
D IACNT                         10I 0 VALUE
D* Array of integer attributes
D INTATR                        10I 0
D* Length of character attributes buffer
D CALEN                         10I 0 VALUE
D* Character attributes
D CHRATR                          *   VALUE
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CMPCOD
D REASON                        10I 0
```

IBM i MQSETMP (Nastavení vlastnosti manipulátoru zprávy) na systému IBM i

Volání MQSETMP nastavuje nebo upravuje vlastnost manipulátoru zprávy.

Syntaxe

MQSETMP *(Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength, Value, CompCode, Reason)*

Poznámky k použití

- Toto volání lze použít pouze v případě, že správce front sám koordinuje jednotku práce. Může se jednat o:
  - Lokální pracovní jednotka, kde změny ovlivňují pouze prostředky IBM MQ .
  - Globální jednotka práce, kde změny mohou ovlivnit prostředky patřící jiným správcům prostředků a prostředky IBM MQ .

  Další podrobnosti o lokálních a globálních jednotkách práce viz "MQBEGIN (začátek transakce) na IBM i" na stránce 1247.
- V prostředích, kde správce front nekoordinuje pracovní jednotku, použijte místo MQBACK příslušné zpětné volání. Prostředí může také podporovat implicitní vrácení zpět způsobené nestandardním ukončením aplikace.
  - V systému z/OSpoužijte následující volání:
    - Dávkové programy (včetně IMS dávkových programů DL/I) mohou používat volání MQBACK, pokud pracovní jednotka ovlivňuje pouze prostředky IBM MQ . Pokud však pracovní jednotka ovlivňuje prostředky IBM MQ i prostředky patřící jiným správcům prostředků (například Db2), použijte volání SRRBACK poskytované službou RRS (z/OS Obnovitelný Resource Service). Volání SRRBACK vrací změny prostředků náležejících správcům prostředků, kteří mají povolenou koordinaci RRS.
    - Aplikace systému CICS musí použít příkaz `EXEC CICS SYNCPOINT ROLLBACK` k vrácení pracovní jednotky zpět. Nepoužívejte volání MQBACK pro aplikace CICS .

- Aplikace IMS (jiné než dávkové programy DL/I) musí používat IMS volání, jako např. ROLB , aby se vrátila jednotka práce. Nepoužívejte volání MQBACK pro aplikace IMS (jiné než dávkové programy DL/I).

– V systému IBM ipoužijte toto volání pro lokální pracovní jednotky koordinované správcem front. To znamená, že na úrovni úlohy nesmí existovat definice vázaného zpracování, to znamená, že pro úlohu nesmí být vydán příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** .

- Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, odebrání těchto změn závisí na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v části "MQDISC (Odpojit správce front) na systému IBM i" na stránce 1283 .
- Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, správce front uchovává informace týkající se skupiny zpráv a logické zprávy pro poslední úspěšná volání MQPUT a MQGET. Tyto informace jsou přidruženy k popisovači fronty a zahrnují například:

  – Hodnoty polí *GroupId*, *MsgSeqNumber*, *Offset*a *MsgFlags* v deskriptoru MQMD.

  – Zda je zpráva součástí jednotky práce.

  – Pro volání MQPUT: zda je zpráva trvalá nebo dočasná.

  Správce front uchovává tři sady informací o skupinách a segmentech, jednu sadu pro každou z následujících položek:

  – Poslední úspěšné volání MQPUT (může být součástí pracovní jednotky).

  – Poslední úspěšné volání MQGET, které odebrala zprávu z fronty (může být součástí pracovní jednotky).

  – Poslední úspěšné volání MQGET, které procházelo zprávu ve frontě (nemůže být součástí pracovní jednotky).

  Pokud aplikace vloží nebo získá zprávy jako součást jednotky práce a pak se rozhodne vrátit zpět jednotku práce, informace o skupině a segmentu se obnoví na hodnotu, kterou měla dříve:

  – Informace přidružené k volání MQPUT jsou obnoveny na hodnotu, kterou měly před prvním úspěšným voláním MQPUT pro daný popisovač fronty v aktuální transakci.

  – Informace přidružené k volání MQGET jsou obnoveny na hodnotu, kterou měly před prvním úspěšným voláním MQGET pro daný manipulátor fronty v aktuální transakci.

  Fronty, které byly aktualizovány aplikací po spuštění jednotky práce, ale mimo rozsah jednotky práce, nemají obnoveny informace o skupinách a segmentech, pokud je jednotka práce odvolána.

  Obnova informací o skupině a segmentu na předchozí hodnotu, když je jednotka práce odvolána, umožňuje aplikaci rozložit velkou skupinu zpráv nebo velkou logickou zprávu sestávající z mnoha segmentů do několika jednotek práce a restartovat ji ve správném bodě ve skupině zpráv nebo logické zprávě, pokud jedna z jednotek práce selže.

  Použití několika pracovních jednotek může být výhodné, pokud má lokální správce front pouze omezené úložiště front. Aplikace však musí udržovat dostatečné informace, aby mohla znovu spustit vkládání nebo získávání zpráv ve správném bodě, dojde-li k selhání systému.

  Podrobnosti o tom, jak provést restart ve správném bodě po selhání systému, viz volba PMLOGO popsaná v části PMOPT (10místné celé číslo se znaménkem)a volba GMLOGO popsaná v části GMOPT (10místné celé číslo se znaménkem).

  Zbývající poznámky k použití platí pouze v případě, že správce front koordinuje pracovní jednotky:

- Jednotka práce má stejný rozsah jako manipulátor připojení. Všechna volání IBM MQ , která ovlivňují konkrétní jednotku práce, musí být provedena pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného manipulátoru připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu manipulátorů připojení viz HCONN (10místné celé číslo se znaménkem)-výstup .
- Toto volání ovlivní pouze zprávy, které byly vloženy nebo načteny jako součást aktuální pracovní jednotky.

- Přerušitelná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale nikdy nevydá volání potvrzení nebo odvolání, může zaplnit fronty zprávami, které nejsou k dispozici pro jiné aplikace. Chcete-li se vyhnout této možnosti, administrátor musí nastavit atribut správce front **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila tomu, že by běžné aplikace zaplňovaly fronty, ale dostatečně vysoká, aby umožňovala správné fungování očekávaných aplikací systému zpráv.

Parametry

Volání MQSETMP má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota musí odpovídat manipulátoru připojení, který byl použit k vytvoření manipulátoru zprávy uvedeného v parametru **HMSG** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podprocesu nastavující vlastnost popisovače zprávy, jinak volání selže s kódem příčiny RC2009 .

HMSG (20místné celé číslo se znaménkem)-vstup

Jedná se o popisovač zprávy, který má být upraven. Hodnota byla vrácena předchozím voláním MQCRTMH.

SETOPT (MQSMPO)-vstup

Řízení způsobu nastavení vlastností zprávy.

Tato struktura umožňuje aplikacím určit volby, které řídí způsob nastavení vlastností zprávy. Struktura je vstupní parametr pro volání MQSETMP. Další informace viz MQSMPO .

PRNAME (MQCHARV)-vstup

Jedná se o název vlastnosti, která má být nastavena.

Další informace o použití názvů vlastností viz Názvy vlastností a Omezení názvů vlastností .

PRPDSC (MQPD)-vstupní/výstupní

Tato struktura se používá k definování atributů vlastnosti, včetně:

- co se stane, když vlastnost není podporována
- jaký kontext zprávy, do kterého vlastnost patří,
- do jakých zpráv je vlastnost kopírována v rámci toku

Další informace o této struktuře viz MQPD .

TYPE (10místné celé číslo se znaménkem)-vstup

Datový typ nastavované vlastnosti. Může se jednat o jednu z následujících možností:

Počet TYPOBOLŮ
Logická hodnota. *ValueLength* musí být 4.

TYPBST
Bajtový řetězec. *ValueLength* musí být nula nebo větší.

TYPI8
8bitové celé číslo se znaménkem. *ValueLength* musí být 1.

TYPI16
16bitové celé číslo se znaménkem. *ValueLength* musí být 2.

TYPI32
32bitové celé číslo se znaménkem. *ValueLength* musí být 4.

TYPI64
64bitové celé číslo se znaménkem. *ValueLength* musí být 8.

TYPF32
32bitové číslo s pohyblivou řádovou čárkou. *ValueLength* musí být 4.

TYPF64
64bitové číslo s pohyblivou řádovou čárkou. *ValueLength* musí být 8.

TYPSTR
Znakový řetězec. *ValueLength* musí být nula nebo větší nebo speciální hodnota VLNULL.

TYPNUL
Vlastnost existuje, ale má hodnotu null. *ValueLength* musí být nula.

VALLEN (10místné celé číslo se znaménkem)-vstup

Délka hodnoty vlastnosti v parametru *Value* v bajtech.

Nula je platná pouze pro hodnoty null nebo pro řetězce nebo bajtové řetězce. Nula označuje, že vlastnost existuje, ale že hodnota neobsahuje žádné znaky nebo bajty.

Hodnota musí být větší nebo rovna nule, nebo následující speciální hodnota, pokud má parametr *Type* nastaven parametr TYPSTR:

VLNULL
Hodnota je oddělena první hodnotou null zjištěnou v řetězci. Hodnota null není zahrnuta jako součást řetězce. Tato hodnota je neplatná, pokud není nastavena také volba TYPSTR.

Poznámka: Znak null použitý k ukončení řetězce, pokud je nastavena hodnota VLNULL, je hodnotou null ze znakové sady hodnoty.

VALUE (1bajtový bitový řetězec x VALLEN)-vstup

Hodnota vlastnosti, která má být nastavena. Vyrovnávací paměť musí být zarovnána na hranici odpovídající povaze dat v hodnotě.

V programovacím jazyku C je parametr deklarován jako ukazatel-na-void; jako parametr lze zadat adresu libovolného typu dat.

Je-li *ValueLength* nula, *Hodnota* se neodkazuje. V tomto případě může mít adresa parametru předaná programy napsané v jazyku C nebo v sestavovacím modulu System/390 hodnotu null.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Je-li *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Je-li hodnota *CMPCOD* CCWARN:

RC2421
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nelze analyzovat.

Je-li hodnota *CMPCOD* CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'86D') Primární a domovský identifikátor ASID se liší.

RC2004
(2004, X'07D4') Parametr hodnoty není platný.

RC2005
(2005, X'07D5') Parametr délky hodnoty není platný.

RC2219
(2219, X'08AB') Volání MQI zadané před dokončením předchozího volání.

RC2460
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

RC2499
(2499, X'09C3') Obsluha zprávy je již používána.

RC2046
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2482
(2482, X'09B2') Struktura deskriptoru vlastnosti není platná.

RC2442
(2442, X'098A') Neplatný název vlastnosti.

RC2473
(2473, X'09A9') Neplatný datový typ vlastnosti.

RC2472
(2472, X'09A8') Došlo k chybě formátu čísla v datech hodnoty.

RC2463
(2463, X'099F') Nastavení struktury vlastností zprávy není platné.

RC2111
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Další informace viz část "Návratové kódy pro IBM i (ILE RPG)" na stránce 1418.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQSETMP(HCONN : HMSG : SETOPT :
                                        PRNAME : PRPDSC :
                                        TYPE : VALLEN : VALUE :
                                        CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
DMQSETMP          PR                  EXTPROC('MQSETMP')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Message handle
D HMSG                          10I 0 VALUE
D* Options that control the action of MQSETMP
D SETOPT                        20A
D* Property name
D PRNAME                        32A
D* Property descriptor
D PRPDSC                        24A
D* Property data type
D TYPE                          10I 0 VALUE
D* Length of the Value area
D VALLEN                        10I 0 VALUE
```

```
     D* Property value
     D VALUE                          *   VALUE
     D* Completion code
     D CMPCOD                       10I 0
     D* Reason code qualifying CompCode
     D REASON                       10I 0
```

IBM i MQSTAT (Načtení informací o stavu) na systému IBM i

Pomocí volání MQSTAT načtěte informace o stavu. Typ vrácených informací o stavu je určen hodnotou STYPE uvedenou ve volání.

Syntaxe

MQSTAT *(HCONN, STYPE, STAT, CMPCOD, REASON)*

Poznámky k použití

1. Volání MQSTAT s určením typu STATAPT vrátí informace o předchozích asynchronních operacích MQPUT a MQPUT1 . Struktura MQSTAT předaná na volání je dokončena s prvními zaznamenanými asynchronními varovnými nebo chybovými informacemi pro dané připojení. Pokud za prvními následují další chyby nebo varování, obvykle tyto hodnoty nezmění. Dojde-li však k chybě s kódem dokončení CCWARN, vrátí se místo toho následné selhání s kódem dokončení CCFAIL.
2. Pokud se od vytvoření připojení nebo od posledního volání MQSTAT nevyskytly žádné chyby, vrátí se CMPCOD CCOK a REASON z RCNONE.
3. Počet asynchronních volání, která byla zpracována pod manipulátor připojení, je vrácen pomocí tří čítačů: STSPSC, STSPWC a STSPFC. Tyto čítače jsou správcem front zvyšovány pokaždé, když je asynchronní operace úspěšně zpracována, mají varování nebo selhávají (všimněte si, že pro účely evidence se počet vložení do distribučního seznamu počítá jednou pro cílovou frontu, nikoli jednou pro distribuční seznam).
4. Úspěšné volání MQSTAT má za následek resetování všech předchozích chybových informací nebo počtů.

Parametry

Volání MQSTAT má následující parametry:

Hconn (MQHCONN)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

STYPE (10místné celé číslo se znaménkem)-vstup

Typ požadovaných informací o stavu. Jediná platná hodnota je:

STATAPT
Vrátit informace o předchozích asynchronních operacích vložení.

STS (MQSTS)-vstupní/výstupní

Struktura informací o stavu. Podrobnosti viz "MQSTS (vykazovací struktura stavu) na systému IBM i" na stránce 1224.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení.

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2374
(2374, X' 946 ') Selhala uživatelská procedura rozhraní API

RC2183
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

RC2219
(2219, X'8AB') Volání MQI zadané před dokončením předchozího volání.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2203
(2203, X'89B') Probíhá ukončování připojení.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2162
(2162, X'872 ') Zastavování správce front

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2430
(2430, X'97E') Chyba typu MQSTAT.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2424
(2424, X' 978 ') Chyba se strukturou MQSTS

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2298
(2298, X'8FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

Podrobné informace o těchto kódech viz:

- Zprávy a kódy příčiny

Prohlášení o RPG

```
C*.. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7
C                     CALLP     MQSTAT(HCONN : ETYPE : ERR :
C                                      CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D.. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7
DMQSTAT           PR                  EXTPROC('MQSTAT')
```

```
     D* Connection handle
     D HCONN                         10I 0 VALUE
     D* Status information type
     D STYPE                         10I 0 VALUE
     D* Status information
     D STATUS                       296A
     D* Completion code
     D CMPCOD                        10I 0
     D* Reason code qualifying CompCode
     D REASON                        10I 0
```

IBM i MQSUB (Registrace odběru) na systému IBM i

Volání MQSUB registruje odběr aplikací pro konkrétní téma.

Syntaxe

MQSUB *(HCONN, SUBDSC, HOBJ, HSUB, CMPCOD, REASON)*

Poznámky k použití

- Odběr je proveden pro téma s názvem buď pomocí krátkého názvu předdefinovaného objektu tématu, úplného názvu řetězce tématu, nebo je vytvořen zřetězením dvou částí, jak je popsáno v tématu Kombinace řetězců tématu.
- Správce front provádí kontroly zabezpečení při zadání volání MQSUB, aby ověřil, zda má identifikátor uživatele, pod kterým je aplikace spuštěna, před povolením přístupu odpovídající úroveň oprávnění. Příslušný objekt tématu je umístěn buď podle krátkého názvu zadaného ve volání, nebo podle nejbližšího objektu krátkého názvu v hierarchii témat, který je nalezen, pokud je zadán dlouhý název. Na tomto objektu tématu se provádí kontrola oprávnění, aby se zajistilo, že je nastaveno oprávnění k odběru, a na cílové frontě, aby se zajistilo, že je nastaveno oprávnění k výstupu. Je-li použita volba SDMAN, znamená to, že je provedena kontrola oprávnění pro název spravované fronty přidružené k tomuto objektu tématu, a je-li poskytnuta nespravovaná fronta, znamená to, že je ve frontě reprezentované parametrem **HOBJ** provedena kontrola oprávnění.
- Při použití volby SOMAN lze zjistit hodnotu *HOBJ* vrácenou při volání MQSUB, aby bylo možné zjistit atributy, jako např. prahovou hodnotu vrácení a název nadměrného vrácení. Můžete také zjistit název spravované fronty, ale neměli byste se pokoušet tuto frontu přímo otevřít.
- Odběry lze seskupit tak, aby bylo možné skupině odběrů doručit pouze jedno publikování, a to i v případě, že více než jedna ze skupin odpovídá publikování. Odběry jsou seskupeny pomocí volby SOGRP a aby bylo možné seskupit odběry, musí:
  - používá stejnou pojmenovanou frontu (která nepoužívá volbu SOMAN) ve stejném správci front- reprezentovaném parametrem **HOBJ** ve volání MQSUB
  - sdílení stejného *SDCID*
  - být stejné *SDSL*

  Tyto atributy definují sadu odběrů považovaných za členy skupiny a jsou také atributy, které nelze změnit, pokud je odběr seskupen. Změna *SDSL* má za následek RC2512a změna všech ostatních (které lze změnit, pokud není odběr seskupen) má za následek RC2515.
- Pole v tabulce MQSD jsou vyplněna při návratu z volání MQSUB, které používá volbu SORES. Vrácený MQSD lze předat přímo do volání MQSUB, které používá volbu SOALT se všemi změnami, které je třeba provést v odběru použitém na MQSD. Některá pole mají zvláštní aspekty, jak je uvedeno v tabulce.

Tabulka 752. Výstup MQSD z MQSUB	
Název pole v MQSD	**Speciální aspekty.**
Volby přístupu nebo vytvoření	Žádná z těchto voleb není nastavena při návratu z volání MQSUB. Pokud později znovu použijete MQSD ve volání MQSUB, musí být volba, kterou požadujete, explicitně nastavena.
Volby trvanlivosti, Volby místa určení, Volby registrace a možnosti zástupného znaku	Tyto volby budou nastaveny podle potřeby.
Volby publikování	Tyto volby budou nastaveny podle potřeby, s výjimkou SONEWP, který se vztahuje pouze na SOCRE.
Další volby	Tyto volby se při návratu z volání MQSUB nezměnily. Řídí, jak je volání rozhraní API vydáváno a není uloženo s odběrem. Musí být nastaveny podle potřeby pro každé následné volání MQSUB, které znovu použije MQSD.
ObjectName	Toto pole pouze pro vstup je při návratu z volání MQSUB nezměněno.
ObjectString	Toto pole pouze pro vstup je při návratu z volání MQSUB nezměněno. Použitý úplný název tématu je vrácen v poli *SDRO* , pokud je poskytnuta vyrovnávací paměť.
AlternateUserID a AlternateSecurityID	Tato vstupní pole jsou při návratu z volání MQSUB beze změny. Řídí, jak je volání rozhraní API vydáváno a není uloženo s odběrem. Musí být nastaveny podle potřeby pro každé následné volání MQSUB, které znovu použije MQSD.
SubExpiry	Při návratu z volání MQSUB pomocí volby SORES bude toto pole nastaveno na původní vypršení platnosti odběru, nikoli na zbývající dobu vypršení platnosti. Pokud poté znovu použijete MQSD ve volání MQSUB s použitím volby SOALT, resetujete vypršení platnosti odběru a začnete znovu počítat.
SubName	Toto pole je vstupní pole pro volání MQSUB a ve výstupu není změněno.
SubUserData a SelectionString	Tato pole s proměnnou délkou budou vrácena při výstupu volání MQSUB s použitím volby SORES, pokud je zadána vyrovnávací paměť, a také s kladnou délkou vyrovnávací paměti v souboru *VCHRP*. Není-li poskytnuta žádná vyrovnávací paměť, bude vrácena pouze délka v poli *VCHRL* atributu MQCHARV.If poskytnutá vyrovnávací paměť menší než prostor požadovaný pro vrácení pole, vrátí se v poskytnuté vyrovnávací paměti pouze *VCHRP* bajtů. Pokud později znovu použijete MQSD ve volání MQSUB s použitím volby SOALT a není poskytnuta vyrovnávací paměť, ale je zadána nenulová hodnota *VCHRL* , pokud tato délka odpovídá existující délce pole, nebude v poli provedena žádná změna.
SubCorrelID a token PubAccounting	Pokud nepoužíváte identifikátor SOSCID, bude správce front generovat soubor *SDCID* . Pokud nepoužijete SOSETI, bude správce front generovat *SDACC* . Tato pole budou vrácena v modulu MQSD z volání MQSUB pomocí volby SORES. Pokud jsou generovány správcem front, bude vygenerovaná hodnota vrácena při volání MQSUB pomocí volby SOCRE nebo SOALT.
PubPriority, SubLevel & PubApplIdentityData	Tato pole budou vrácena v modulu MQSD.
Řetězec ResObject	Toto pole pouze pro výstup bude vráceno v MQSD, pokud je poskytnuta vyrovnávací paměť.

Parametry

Volání MQSUB má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

SUBDSC (MQSD)-vstup/výstup

Jedná se o strukturu, která identifikuje objekt s použitím, který je registrován aplikací. Další informace viz “MQSD (deskriptor odběru) na systému IBM i” na stránce 1207.

HOBJ (10místné celé číslo se znaménkem)-vstup/výstup

Tento popisovač představuje přístup, který byl vytvořen pro získání zpráv odeslaných tomuto odběru. Tyto zprávy mohou být uloženy ve specifické frontě nebo může být správce front požádán o správu úložiště bez potřeby specifické fronty.

Popisovač objektu.

Má-li být použita specifická fronta, musí být přidružena k odběru v době vytvoření. To lze provést dvěma způsoby:

- Poskytnutím tohoto popisovače při volání MQSUB s volbou SDCRT. Je-li tento manipulátor zadán jako vstupní parametr volání, musí se jednat o platný manipulátor objektu vrácený z předchozího volání MQOPEN fronty s použitím alespoň jedné z voleb OOOINP*, OOOUT (například pro vzdálenou frontu) nebo OOBRW. Pokud se nejedná o tento případ, volání se nezdaří s volbou RC2019. Nemůže se jednat o popisovač objektu pro frontu aliasů, která se interpretuje jako objekt tématu. Pokud ano, volání selže s RC2019
- Použitím příkazu DEFINE SUB MQSC a zadáním tohoto příkazu s názvem objektu fronty.

Má-li správce front spravovat úložiště zpráv odesílaných do tohoto odběru, měli byste tuto skutečnost označit při vytvoření odběru pomocí volby SOMAN a nastavením parametru na hodnotu HONONE. Správce front vrací manipulátor jako výstupní parametr volání a vrácený manipulátor je označován jako spravovaný manipulátor. Je-li uveden HONONE a není-li také uveden SOMAN, volání selže s RC2019.

Spravovaný popisovač vrácený správcem front lze použít pro volání MQGET nebo MQCB, s volbami procházení nebo bez nich, pro volání MQINQ nebo MQCLOSE. Nelze jej použít na MQPUT, MQSET nebo na následném MQSUB; pokus o to se nezdaří s RC2039 pro MQPUT, RC2040 pro MQSET nebo RC2038 pro MQSUB.

Je-li k obnovení tohoto odběru použita volba SORES v poli *OPTS* ve struktuře MQSD, lze manipulátor vrátit aplikaci v tomto parametru, je-li zadán parametr HONONE. Tuto možnost můžete použít bez ohledu na to, zda odběr používá spravovaný manipulátor či nikoli. Může být užitečné pro odběry vytvořené pomocí příkazu DEFINE SUB, pokud chcete manipulovat s frontou odběrů definovanou v příkazu DEFINE SUB. V případě obnovení administrativně vytvořeného odběru se fronta otevře s OOINPQ a OOBRW. Jsou-li zapotřebí další volby, musí aplikace explicitně otevřít frontu odběru a poskytnout manipulátor objektu pro volání. Pokud se vyskytl problém s otevřením fronty, volání selže s RC2522. Je-li zadán parametr *HOBJ* , musí být ekvivalentem parametru *HOBJ* v původním volání MQSUB. To znamená, že pokud je poskytován popisovač objektu vrácený z volání MQOPEN, musí být tento popisovač ve stejné frontě, jako byl použit dříve, jinak volání selže s RC2019.

Pokud je tento odběr měněn pomocí volby SOALT v poli *OPTS* ve struktuře MQSD, lze zadat jiný parametr *HOBJ* . Veškerá publikování, která byla doručena do fronty dříve identifikované prostřednictvím tohoto parametru, zůstávají v této frontě a je zodpovědností aplikace tyto zprávy načíst, pokud parametr **HOBJ** nyní představuje jinou frontu.

Použití tohoto parametru s různými volbami odběru je shrnuto v následující tabulce:

Tabulka 753. Použití Hobj s různými možnostmi předplatného

Volby	HOBJ	Popis
SOCRT + SOMAN	Ignorováno na vstupu	Vytvoří odběr se spravovaným úložištěm zpráv správce front.

Tabulka 753. Použití Hobj s různými možnostmi předplatného (pokračování)		
Volby	**HOBJ**	**Popis**
SOCRT	Platný popisovač objektu	Vytvoří odběr poskytující specifickou frontu jako cíl pro zprávy.
SORES	HONONE	Obnoví dříve vytvořený odběr (spravovaný či nikoli) a správce front vrátí manipulátor objektu pro použití aplikací.
SORES	Platný, odpovídající, popisovač objektu	Obnoví dříve vytvořený odběr, který používá specifickou frontu jako cíl pro zprávy a používá popisovač objektu se specifickými otevřenými volbami.
SOALT + SOMAN v České a České	HONONE	Změní existující odběr, který dříve používal specifickou frontu, na nyní spravovaný.
ALT	Platný popisovač objektu	Změní existující odběr tak, aby používal specifickou frontu (buď ze spravované, nebo z jiné specifické fronty).

Bez ohledu na to, zda byl zadán nebo vrácen, musí být v následných voláních MQGET zadán parametr *HOBJ* , který potřebujete pro příjem publikování.

Popisovač *HOBJ* přestane být platný, když je na něm vydáno volání MQCLOSE nebo když je ukončena jednotka zpracování, která definuje rozsah popisovače. Rozsah vráceného manipulátoru objektu je stejný jako rozsah manipulátoru připojení určeného ve volání. Informace o rozsahu popisovače viz HCONN . Objekt MQCLOSE manipulátoru *HOBJ* nemá žádný vliv na manipulátor *HSUB* .

HSUB (10místné celé číslo se znaménkem)-výstup

Tento popisovač představuje odběr, který byl proveden. Může být použit pro další dvě operace:

- Lze jej použít při následném volání MQSUBRQ a požádat o odeslání publikování v případě, že byla při vytváření odběru použita volba SOPUBR.
- Lze jej použít při následném volání MQCLOSE k odebrání provedeného odběru. Manipulátor *HSUB* přestane být platný při zadání volání MQCLOSE nebo při ukončení jednotky zpracování, která definuje rozsah manipulátoru. Rozsah vráceného manipulátoru objektu je stejný jako rozsah manipulátoru připojení určeného ve volání. Objekt MQCLOSE manipulátoru *HSUB* nemá žádný vliv na manipulátor *HOBJ* .

Tento manipulátor nelze předat volání MQGET nebo MQCB. Musíte použít parametr **HOBJ** . Předání tohoto popisovače jakémukoli jinému volání IBM MQ má za následek RC2019.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení

CCWARN (varování)
Varování (částečné dokončení)

CCFAIL
Volání selhalo

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Pokud je *CMPCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CMPCOD* hodnotu CCFAIL:

RC2019
(2019 X'07E3') popisovač objektu není platný

RC2046
(2046 X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2085
(2085 X'0825 ') Nelze nalézt identifikovaný objekt

RC2161
(2161 X'0871 ') uvedení správce front do klidového stavu

RC2298
(2298 X'08FA') Funkce není podporována.

RC2424
(2424 X'0978 ') Deskriptor odběru (MQSD) není platný

RC2425
(2441 X' 979 ') Řetězec tématu není platný

RC2428
(2428 X'097C') Zadaný název odběru neodpovídá existujícím odběrům.

RC2429
(2429 X'097D') Název odběru existuje a je používán jinou aplikací.

RC2431
(2431 X'097F') SubUsernení platné

RC2432
(2432 X'0980 ') Odběr existuje

RC2434
(2434 X'0982 ') Název odběru odpovídá existujícímu odběru.

RC2440
(2440 X'0988 ') Pole SubName není platné.

RC2441
(2441 X'0989 ') Pole Objectstring není platné

RC2435
(2435 X'0983 ') Atribut nelze změnit pomocí SDALT nebo byl vytvořen odběr pomocí SDIMM.

RC2436
(2436 X'0984 ') Volba SODUR není platná

RC2459
(2459, X'99B') Chyba syntaxe řetězce výběru.

RC2503
(2503 X'09C7') Volání MQSUB jsou v současné době blokována pro odebíraná témata.

RC2519
(2519, X'9D7') Řetězec výběru není uveden v popisu použití struktury MQCHARV.

RC2551
(2551, X'9F7') Určený řetězec výběru není k dispozici.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                   CALLP     MQSUB(HCONN : SUBDSC : HOBJ :
C                                   HSUB : CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQSUB            PR                EXTPROC('MQSUB')
```

```
     D* Connection handle
     D HCONN                         10I 0 VALUE
     D* Subscription descriptor
     D SUBDSC                       400A
     D* Object handle for queue
     D HOBJ                          10I 0
     D* Subscription object handle
     D HSUB                          10I 0
     D* Completion code
     D CMPCOD                        10I 0
     D* Reason code qualifying CompCode
     D REASON                        10I 0
```

IBM i MQSUBRQ (požadavek na odběr) na systému IBM i

Volání MQSUBRQ vytváří požadavek na odběr.

Syntaxe

MQSUBRQ *(HCONN, HSUB, ACTION, SUBROPT, CMPCOD, REASON)*

Poznámky k použití

Pro použití SRAPUB platí následující poznámky k použití:

1. Pokud je toto příkazové slovo úspěšně dokončeno, zachovaná publikování odpovídající zadanému odběru byla odeslána do odběru a lze je přijmout pomocí příkazu MQGET nebo MQCB s použitím příkazu HOBJ vráceného v původním příkazu MQSUB, který vytvořil odběr.
2. Pokud téma odebírané původním příkazovým slovem MQSUB, které vytvořilo odběr, obsahovalo zástupný znak, může být odesláno více zachovaných publikování. Počet publikování odeslaných jako výsledek tohoto volání je zaznamenán v poli *SRNMP* ve struktuře SBROPT.
3. Pokud je toto příkazové slovo dokončeno s kódem příčiny RC2437 , pak pro uvedené téma nebyla momentálně zachována žádná publikování.
4. Pokud je toto příkazové slovo dokončeno s kódem příčiny RC2525 nebo RC2526 , pak jsou momentálně uchována publikování pro uvedené téma, ale vyskytla se chyba, která znamenala, že nebylo možné je doručit.
5. Aplikace musí mít aktuální odběr tématu, aby mohla toto volání provést. Pokud byl odběr proveden v předchozí instanci aplikace a není k dispozici platný manipulátor odběru, musí aplikace nejprve volat MQSUB s volbou SORES, aby získala manipulátor pro použití v tomto volání.
6. Publikování jsou odesílána do místa určení, které je registrováno pro použití s aktuálním odběrem této aplikace. Má-li být publikování odesláno někam jinam, je třeba nejprve změnit odběr pomocí volání MQSUB s volbou SOALT.

Parametry

Volání MQSUBRQ má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V aplikacích z/OS for CICS lze volání MQCONN vynechat a pro *HCONN*zadat následující hodnotu:

HCDEFH
Výchozí manipulátor připojení.

HSUB (10místné celé číslo se znaménkem)-vstup

Tento popisovač představuje odběr, pro který má být požadována aktualizace. Hodnota *HSUB* byla vrácena z předchozího volání MQSUB.

ACTION (10místné celé číslo se znaménkem)-vstup

Tento parametr řídí konkrétní akci, která je požadována pro odběr. Musí být uveden jeden (a pouze jeden) z následujících:

SRAPUB

Tato akce vyžaduje odeslání publikace aktualizace pro uvedené téma. Tato volba se obvykle používá v případě, že odběratel zadal volbu SOPUBR ve volání MQSUB při provádění odběru. Pokud má správce front zachované publikování pro dané téma, bude odesláno odběrateli. Pokud ne, volání selže. Pokud je aplikaci odesláno publikování, které bylo zachováno, je to označeno vlastností zprávy MQIsRetained tohoto publikování.

Vzhledem k tomu, že téma v existujícím odběru reprezentovaném parametrem **HSUB** může obsahovat zástupné znaky, může odběratel obdržet více zachovaných publikování.

SBROPT (MQSRO)-vstupní/výstupní

Tyto volby řídí akci MQSUBRQ, podrobnosti viz “MQSRO-Volby požadavku na odběr” na stránce 591 .

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení; jedná se o jeden z následujících:

CCOK
Úspěšné dokončení

CCWARN (varování)
Varování (částečné dokončení)

CCFAIL
Volání selhalo

Příčina (10místné celé číslo se znaménkem)-výstup

Kód příčiny, který kvalifikuje *CMPCOD*.

Pokud je *CPMCOD* CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CPMCOD* hodnotu CCFAIL:

RC2298
2298 (X'08FA') Požadovaná funkce není v aktuálním prostředí k dispozici.

RC2437
2437 (X'0985 ') Pro toto téma nejsou aktuálně uložena žádná zachovaná publikování.

RC2046
2046 (X'07FE') Parametr nebo pole voleb obsahuje neplatné volby nebo kombinaci neplatných voleb.

RC2161
2161 (X'0871 ') uvedení správce front do klidového stavu

RC2438
2438 (X'0986 ') Při volání MQSUBRQ nejsou volby požadavku na odběr MQSRO platné.

Prohlášení o RPG

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQSUBRQ(HCONN : HSUB : ACTION :
C                                       SBROPT : CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQSUBRQ          PR                  EXTPROC('MQSUBRQ')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Subscription handle
D HSUB                          10I 0 VALUE
D* Action requested on the subscription
D ACTION                        10I 0 VALUE
D* Subscription Request Options
D SBROPT                        16A
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CompCode
D REASON                        10I 0
```

IBM i Atributy objektů v systému IBM i

V této kolekci témat jsou uvedeny pouze ty objekty IBM MQ , které mohou být předmětem volání funkce MQINQ, a jsou uvedeny podrobnosti o atributech, které lze dotazovat, a o selektorech, které mají být použity.

Atributy pro fronty

Pomocí těchto informací se dozvíte o různých typech definic front a atributech podporovaných jednotlivými definicemi.

Typy front: Správce front podporuje následující typy definic front:

Lokální fronta
Jedná se o fyzickou frontu, která ukládá zprávy. Fronta existuje v lokálním správci front.

Aplikace připojené k lokálnímu správci front mohou umisťovat zprávy do front tohoto typu a odebírat je z nich. Hodnota atributu fronty **QType** je QTLOC.

Sdílená fronta
Jedná se o fyzickou frontu, která ukládá zprávy. Fronta existuje ve sdíleném úložišti, které je přístupné pro všechny správce front, kteří patří do skupiny sdílení front, jež je vlastníkem sdíleného úložiště.

Aplikace připojené k libovolnému správci front ve skupině sdílení front mohou umísťovat zprávy do front tohoto typu a odebírat zprávy z front tohoto typu. Tyto fronty jsou ve skutečnosti stejné jako lokální fronty. Hodnota atributu fronty **QType** je QTLOC.

- Sdílené fronty jsou podporovány pouze v systému z/OS.

Fronta klastru
Jedná se o fyzickou frontu, která ukládá zprávy. Fronta existuje buď v lokálním správci front, nebo v jednom či více správcích front, kteří patří do stejného klastru jako lokální správce front.

Aplikace připojené k lokálnímu správci front mohou umisťovat zprávy do front tohoto typu bez ohledu na umístění fronty. Pokud instance fronty existuje v lokálním správci front, fronta se chová stejně jako lokální fronta a aplikace připojené k lokálnímu správci front mohou odebírat zprávy z fronty. Hodnota atributu fronty **QType** je QTCLUS.

Fronta aliasů
Nejedná se o fyzickou frontu-jedná se o alternativní název pro lokální frontu. Název lokální fronty, na kterou se alias interpretuje, je součástí definice alias fronty.

Aplikace připojené k lokálnímu správci front mohou umísťovat zprávy do front aliasů a odebírat je z front aliasů-zprávy jsou umisťovány do lokální fronty, do níž se alias převádí, a odebírány z ní. Hodnota atributu fronty **QType** je QTALS.

Vzdálená fronta
Nejedná se o fyzickou frontu-jedná se o lokální definici fronty, která existuje ve vzdáleném správci front. Lokální definice vzdálené fronty obsahuje informace, které lokálnímu správci front sdělují, jak má směrovat zprávy do vzdáleného správce front.

Aplikace připojené k lokálnímu správci front mohou umisťovat zprávy do vzdálených front-zprávy jsou umístěny do lokální přenosové fronty používané ke směrování zpráv do vzdáleného správce front. Aplikace nemohou odebírat zprávy ze vzdálených front. Hodnota atributu fronty **QType** je QTREM.

Definici vzdálené fronty lze také použít pro:

- Alias fronty odpovědí

  V tomto případě je název definice názvem fronty pro odpověď. Další informace naleznete v tématu Definice aliasu fronty pro odpověď.
- Aliasy správce front

  V tomto případě je název definice aliasem pro správce front, nikoli názvem fronty. Další informace naleznete v tématu Definice aliasů správce front.

Modelová fronta
Nejedná se o fyzickou frontu-jedná se o sadu atributů fronty, ze které lze vytvořit lokální frontu.

Zprávy nelze ukládat do front tohoto typu.

Některé atributy fronty platí pro všechny typy front; jiné atributy fronty platí pouze pro určité typy front. Typy front, pro které se atribut používá, jsou označeny znakem "X" v tabulce Tabulka 754 na stránce 1361 a následných tabulkách.

Tabulka 754 na stránce 1361 shrnuje atributy, které jsou specifické pro fronty. Atributy jsou popsány v abecedním pořadí.

Názvy atributů zobrazené v tabulce jsou názvy použité s voláními MQINQ a MQSET. Při použití příkazů MQSC k definování, změně nebo zobrazení atributů jsou použity alternativní krátké názvy. Podrobnosti naleznete v části Příkazy MQSC .

V následující tabulce platí následující sloupce:

- Sloupec pro lokální fronty platí také pro sdílené fronty.
- Sloupec pro modelové fronty označuje, které atributy jsou zděděny lokální frontou vytvořenou z modelové fronty.
- Sloupec pro fronty klastru označuje atributy, které lze zjistit, když je fronta klastru otevřena pouze pro dotazování nebo pro dotazování a výstup. Je-li fronta klastru otevřena pro dotazování plus jeden nebo více vstupů, procházení nebo nastavení, použije se místo toho sloupec pro lokální fronty.

Tabulka 754. Atributy pro fronty

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
AlterationDate	Datum, kdy byla definice naposledy změněna	X		X	X	
AlterationTime	Čas, kdy byla definice naposledy změněna	X		X	X	
BackoutRequeueQName	Nadměrný název fronty vrácení	X	X			
BackoutThreshold	Práh vrácení	X	X			
BaseQName	Název fronty, na kterou se alias interpretuje			X		

Tabulka 754. Atributy pro fronty (pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
ClusterChannel	Název odesílacího kanálu klastru	✓	✓			
ClusterName	Název klastru, ke kterému fronta patří	X		X	X	
ClusterNamelist	Název objektu seznamu názvů obsahujícího názvy klastrů, ke kterým fronta náleží	X		X	X	
CreationDate	Datum, kdy byla fronta vytvořena	X				
CreationTime	Čas vytvoření fronty	X				
CurrentQDepth	Aktuální hloubka fronty	X				
DefBind	Výchozí vazba	X		X	X	X
DefinitionType	Typ definice fronty	X	X			
DefInputOpenOption	Výchozí volba otevření pro vstup	X	X			
DefPersistence	Výchozí trvalost zpráv	X	X	X	X	X
DefPriority	Výchozí priorita zpráv	X	X	X	X	X
DistLists	Podpora seznamu distribuce	X	X			
HardenGetBackout	Zda se má udržovat přesný počet vrácení	X	X			
InhibitGet	Řídí, zda jsou povoleny operace získání pro frontu	X	X	X		
InhibitPut	Řídí, zda jsou povoleny operace vložení pro frontu	X	X	X	X	X
InitiationQName	Název inicializační fronty	X	X			
MaxMsgLength	Maximální délka zprávy v bajtech	X	X			
MaxQDepth	Maximální hloubka fronty	X	X			

Tabulka 754. Atributy pro fronty (pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
MediaLog	Identita nejstaršího rozsahu protokolu (nebo nejstaršího žurnálového zásobníku na systému IBM i) potřebné pro obnovu médií uvedené fronty	✓	✓			
MsgDeliverySequence	Pořadí doručení zpráv	X	X			
OpenInputCount	Počet otevření pro vstup	X				
OpenOutputCount	Počet otevření pro výstup	X				
ProcessName	Název procesu	X	X			
QDepthHighEvent	Řídí, zda jsou generovány události Vysoká hloubka fronty	X	X			
QDepthHighLimit	Horní limit pro délku fronty	X	X			
QDepthLowEvent	Řídí, zda jsou generovány události dolní hloubky fronty	X	X			
QDepthLowLimit	Dolní limit pro hloubku fronty	X	X			
QDepthMaxEvent	Řídí, zda jsou generovány události zaplnění fronty	X	X			
QDesc	Popis fronty	X	X	X	X	X
QName	Název fronty	X		X	X	X
QServiceInterval	Cíl pro interval služby fronty	X	X			
QServiceIntervalUdálost	Řídí, zda jsou generovány události Service Interval High nebo Service Interval OK	X	X			
QTYPE	Typ fronty	X		X	X	X
RemoteQmgrName	Název vzdáleného správce front				X	
RemoteQName	Název vzdálené fronty				X	
RetentionInterval	Interval uchování	X	X			

Tabulka 754. Atributy pro fronty (pokračování)						
Atribut	**Popis**	**Lokální**	**Model**	**Alias**	**Vzdálený**	**Klastr**
Obor	Určuje, zda položka pro frontu existuje také v adresáři buňky.	X		X	X	
Možnost sdílení	Možnost sdílení fronty	X	X			
TriggerControl	Řízení spouštěče	X	X			
TriggerData	Data spouątěče	X	X			
TriggerDepth	Hloubka spouątěče	X	X			
TriggerMsgPriority	Prahová hodnota priority zpráv pro spouštěče	X	X			
TriggerType	Typ spouątěče	X	X			
Pouľití	Použití fronty	X	X			
XmitQName	Jméno přenosové fronty				X	

IBM i *AlterationDate (12bajtový znakový řetězec) na IBM i*

Datum, kdy byla definice naposledy změněna.

Tabulka 755. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	

Toto je datum, kdy byla definice naposledy změněna. Formát data je `YYYY-MM-DD`, vyplněný dvěma koncovými mezerami, aby byla délka 12 bajtů (například `1992-09-23¬¬`, kde ¬¬ představuje dva prázdné znaky).

Hodnoty určitých atributů (například *`CurrentQDepth`*) se mění při práci správce front. Změny těchto atributů neovlivňují *`AlterationDate`*.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTD s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNDATE.

IBM i *AlterationTime (8bajtový znakový řetězec) na IBM i*

Čas, kdy byla definice naposledy změněna.

Tabulka 756. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	

Toto je čas, kdy byla definice naposledy změněna. Formát času je `HH.MM.SS` používající 24hodinové hodiny, s úvodní nulou, pokud je hodina menší než 10 (například `09.10.20`). Čas je místní čas.

Hodnoty určitých atributů (například *`CurrentQDepth`*) se mění při práci správce front. Změny těchto atributů neovlivňují *`AlterationTime`*.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTT s voláním MQINQ. Délka tohoto atributu je dána LNTIME.

IBM i *BackoutRequeueQName (48bajtový znakový řetězec) na IBM i*

Nadměrný počet vrácený název fronty pro vrácení.

Tabulka 757. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Aplikace spuštěné v produktu WebSphere Application Server a ty, které používají zařízení serveru IBM MQ Application Server, používají tento atribut k určení, kam by měly zprávy, které byly vráceny zpět, jít. Pro všechny ostatní aplikace, kromě povolení dotazování na její hodnotu, neprovede správce front žádnou akci založenou na hodnotě atributu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CABRQN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

IBM i *BackoutThreshold (10místné celé číslo se znaménkem) na IBM i*

Prahová hodnota vyřazených zpráv.

Tabulka 758. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Aplikace spuštěné v produktu WebSphere Application Server a ty, které používají zařízení IBM MQ Application Server Facilities, používají tento atribut k určení, zda má být zpráva vrácena zpět. Pro všechny ostatní aplikace, kromě povolení dotazování na její hodnotu, neprovede správce front žádnou akci založenou na hodnotě atributu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IABTHR s voláním MQINQ.

IBM i *BaseQName (48bajtový znakový řetězec) na IBM i*

Název fronty, na kterou se alias interpretuje.

Tabulka 759. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
		X		

Jedná se o název fronty, která je definována pro lokálního správce front. (Další informace o názvech front naleznete v popisu pole *ODON* v produktu MQOD. Fronta je jednoho z následujících typů:

QTLOC
Lokální fronta.

QTREM
Lokální definice vzdálené fronty.

QTCLUS
Fronta klastru.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CABASQ s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

IBM i *BaseType (struktura celočíselných parametrů) na IBM i*

Typ objektu, na který se alias interpretuje.

Tabulka 760. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
		X		

Tento atribut může mít některou z následujících hodnot:

OTQ (dotazů)
Základní typ objektu je fronta

OTTOP
Základní typ objektu je téma

IBM i *CFStrucName (12bajtový znakový řetězec) na IBM i*

Název struktury prostředku Coupling Facility.

Tabulka 761. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Jedná se o název struktury prostředku Coupling Facility, v níž jsou uloženy zprávy ve frontě. První znak názvu je v rozsahu A až Z a zbývající znaky jsou v rozsahu A až Z, 0 až 9 nebo prázdné.

Úplný název struktury v prostředku Coupling Facility je získán doplněním hodnoty atributu správce front **QSGName** o hodnotu atributu fronty **CFStrucName** .

Tento atribut platí pouze pro sdílené fronty; je ignorován, pokud *QSGDisp* nemá hodnotu QSGDSH.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACFSN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNCFSN.

z/OS Tento atribut je podporován pouze na systému z/OS.

ClusterChannelNázev (20bajtový znakový řetězec)

`ClusterChannel` je generický název odesílacích kanálů klastru, které používají tuto frontu jako přenosovou frontu. Atribut uvádí, které odesílací kanály klastru budou z této přenosové fronty klastru posílat zprávy do přijímacího kanálu klastru.

Tabulka 762. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Výchozí konfigurace správce front je určena pro všechny odesílací kanály klastru k odesílání zpráv z jedné přenosové fronty `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. Výchozí konfiguraci lze změnit úpravou atributu správce front, **DefClusterXmitQueueType**. Výchozí hodnota tohoto atributu je SCTQ. Tuto hodnotu můžete změnit na `CHANNEL`. Nastavíte-li atribut **DefClusterXmitQueueType** na hodnotu `CHANNEL`, bude každý odesílací kanál klastru standardně používat specifickou přenosovou frontu klastru, `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`*.

Atribut přenosové fronty `ClusterChannelName` můžete také nastavit na odesílací kanál klastru ručně. Zprávy, které jsou určeny pro správce front připojeného prostřednictvím odesílacího kanálu klastru, jsou uloženy do přenosové fronty, která identifikuje odesílací kanál klastru. Tyto zprávy se nebudou ukládat do výchozí přenosové fronty klastru. Pokud nastavíte atribut `ClusterChannelName` na prázdné znaky, přepne se kanál na výchozí přenosovou frontu klastru, jakmile se kanál restartuje. Výchozí fronta je buď `SYSTEM.CLUSTER.TRANSMIT.`*`ChannelName`*, nebo `SYSTEM.CLUSTER.TRANSMIT.QUEUE`, v závislosti na hodnotě atributu správce front `DefClusterXmitQueueType`.

Zadáním hvězdiček, "`*`", do pole **`ClusterChannelName`** můžete přidružit přenosovou frontu k sadě odesílacích kanálů klastru. Hvězdička může být na začátku, na konci nebo kdekoli ve středu řetězce názvu klastru. Pole **`ClusterChannelName`** je omezeno na délku 20 znaků: `MQ_CHANNEL_NAME_LENGTH`.

IBM i *ClusterName (48bajtový znakový řetězec) na IBM i*

Název klastru, ke kterému fronta patří.

Tabulka 763. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Jedná se o název klastru, ke kterému fronta patří. Pokud fronta patří do více než jednoho klastru, *`ClusterNamelist`* uvádí název objektu seznamu názvů, který identifikuje klastry, a *`ClusterName`* je prázdné. Alespoň jeden z *`ClusterName`* a *`ClusterNamelist`* musí být prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACLN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNCLUN.

IBM i *ClusterNamelist (48bajtový znakový řetězec) na IBM i*

Název objektu seznamu názvů obsahujícího názvy klastrů, ke kterým fronta náleží.

Tabulka 764. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Jedná se o název objektu seznamu názvů, který obsahuje názvy klastrů, do kterých tato fronta patří. Pokud fronta náleží pouze jednomu klastru, obsahuje objekt seznamu názvů pouze jeden název. Alternativně lze *`ClusterName`* použít k určení názvu klastru, v takovém případě je *`ClusterNamelist`* prázdný. Alespoň jeden z *`ClusterName`* a *`ClusterNamelist`* musí být prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACLNL s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNNLN.

IBM i *CreationDate (12bajtový znakový řetězec) na IBM i*

Datum, kdy byla fronta vytvořena.

Tabulka 765. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Toto je datum vytvoření fronty. Formát data je `YYYY-MM-DD`, vyplněný dvěma koncovými mezerami, aby byla délka 12 bajtů (například `1992-09-23¬¬`, kde ¬¬ představuje dva prázdné znaky).

- V systému IBM ise může datum vytvoření fronty lišit od data vytvoření základní entity operačního systému (souboru nebo uživatelského prostoru), která představuje frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACRTD s voláním MQINQ. Délka tohoto atributu je dána LNCRTD.

IBM i *CreationTime (8bajtový znakový řetězec) na IBM i*

Čas, kdy byla fronta vytvořena.

Tabulka 766. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Toto je čas, kdy byla fronta vytvořena. Formát času je `HH.MM.SS` používající 24hodinové hodiny, s úvodní nulou, pokud je hodina menší než 10 (například `09.10.20`). Čas je místní čas.

- V systému IBM ise může čas vytvoření fronty lišit od času vytvoření základní entity operačního systému (souboru nebo uživatelského prostoru), která představuje frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACRTT s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNCRTT.

IBM i *CurrentQDepth (10místné celé číslo se znaménkem) na IBM i*

Aktuální hloubka fronty.

Tabulka 767. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Jedná se o počet zpráv aktuálně uložených ve frontě. Zvyšuje se během volání MQPUT a během vrácení volání MQGET. Je dekrementován během volání MQGET bez procházení a během odvolání volání MQPUT. Výsledkem je, že tento počet zahrnuje zprávy, které byly vloženy do fronty v rámci pracovní jednotky, ale dosud nebyly potvrzeny, i když nejsou vhodné k načtení voláním MQGET. Podobně vyloučí zprávy, které byly načteny v rámci pracovní jednotky pomocí volání MQGET, ale dosud nebyly potvrzeny.

Tento počet také zahrnuje zprávy, které prošly časem vypršení platnosti, ale dosud nebyly vyřazeny, ačkoli tyto zprávy nejsou vhodné k načtení. Viz pole *MDEXP* popsané v části "MQMD (deskriptor zprávy) na IBM i" na stránce 1105.

Zpracování jednotky práce a segmentace zpráv mohou způsobit, že *CurrentQDepth* překročí *MaxQDepth*. To však nemá vliv na načítatelnost zpráv- *všechny* zprávy ve frontě lze načíst běžným způsobem pomocí volání MQGET.

Hodnota tohoto atributu kolísá v průběhu činnosti správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACDEP s voláním MQINQ.

IBM i *DefBind (10místné celé číslo se znaménkem) na IBM i*

Výchozí vazba.

Tabulka 768. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	X

Tento atribut je výchozí vazbou, která se používá, když je ve volání MQOPEN zadána hodnota OOBNDQ a fronta je fronta klastru. DefBind může mít jednu z následujících hodnot:

BNDOPN
Vazba byla opravena voláním MQOPEN.

BNDNOT
Vazba není opravena.

BNDGRP
Vazba není opravena voláním MQOPEN, ale je opravena na MQPUT pro všechny zprávy v logické skupině.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADBND s voláním MQINQ.

IBM i *DefinitionType (10místné celé číslo se znaménkem) na IBM i*

Typ definice fronty.

Tabulka 769. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Označuje, jak byla fronta definována. Hodnota je jedna z následujících:

QDPRE
Předdefinovaná trvalá fronta.

Fronta je trvalá fronta vytvořená administrátorem systému; odstranit ji může pouze administrátor systému.

Předdefinované fronty jsou vytvořeny pomocí příkazu DEFINE MQSC a lze je odstranit pouze pomocí příkazu DELETE MQSC. Předdefinované fronty nelze vytvořit z modelových front.

Příkazy mohou být vydány buď operátorem, nebo autorizovaným uživatelem odesílajícím zprávu příkazu do vstupní fronty příkazů (viz atribut **CommandInputQName** popsaný v části “Atributy pro správce front v systému IBM i” na stránce 1391).

QDPERM
Dynamicky definovaná trvalá fronta.

Fronta je trvalá fronta, která byla vytvořena aplikací vydávající volání MQOPEN s názvem modelové fronty určené v deskriptoru objektu MQOD. Definice modelové fronty měla hodnotu QDPERM pro atribut **DefinitionType** .

Tento typ fronty lze odstranit pomocí volání MQCLOSE. Další informace viz část “MQCLOSE (Zavřít objekt) na IBM i” na stránce 1261.

Hodnota atributu **QSGDisp** pro trvalou dynamickou frontu je QSGDQM.

QDTEMP
Dynamicky definovaná dočasná fronta.

Fronta je dočasná fronta, která byla vytvořena aplikací vydávající volání MQOPEN s názvem modelové fronty určené v deskriptoru objektu MQOD. Definice modelové fronty měla hodnotu QDTEMP pro atribut **DefinitionType** .

Tento typ fronty je automaticky odstraněn voláním MQCLOSE, když je zavřen aplikací, která jej vytvořila.

Hodnota atributu **QSGDisp** pro dočasnou dynamickou frontu je QSGDQM.

QDSHAR
Dynamicky definovaná sdílená fronta.

Fronta je sdílená trvalá fronta, která byla vytvořena aplikací vydávající volání MQOPEN s názvem modelové fronty určené v deskriptoru objektu MQOD. Definice modelové fronty měla hodnotu QDSHAR pro atribut **DefinitionType** .

Tento typ fronty lze odstranit pomocí volání MQCLOSE. Další informace viz část “MQCLOSE (Zavřít objekt) na IBM i” na stránce 1261.

Hodnota atributu **QSGDisp** pro sdílenou dynamickou frontu je QSGDSH.

Tento atribut v definici modelové fronty neoznačuje, jak byla modelová fronta definována, protože modelové fronty jsou vždy předdefinované. Místo toho se hodnota tohoto atributu v modelové frontě používá k určení *DefinitionType* každé dynamické fronty vytvořené z definice modelové fronty pomocí volání MQOPEN.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADEFT s voláním MQINQ.

IBM i *DefInputOpenOption (10místné celé číslo se znaménkem) na IBM i*

Výchozí volba otevření vstupu.

Tabulka 770. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Jedná se o výchozí způsob, jakým by měla být fronta otevřena pro vstup. Použije se, pokud je volba OOINPQ zadána ve volání MQOPEN při otevření fronty. Může mít jednu z následujících hodnot:

OOINPX
Chcete-li získat zprávy s výlučným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání selže s kódem příčiny RC2042 , pokud je fronta momentálně otevřena touto nebo jinou aplikací pro vstup libovolného typu (OOOINPS nebo OOINPX).

OOINPS
Chcete-li získat zprávy se sdíleným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání může být úspěšné, pokud je fronta momentálně otevřena touto nebo jinou aplikací s OOINPS, ale selže s kódem příčiny RC2042 , pokud je fronta momentálně otevřená s OOOINPX.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADINP s voláním MQINQ.

IBM i *DefPersistence (10místné celé číslo se znaménkem) na IBM i*

Výchozí perzistence zpráv.

Tabulka 771. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Jedná se o výchozí perzistenci zpráv ve frontě. Použije se, pokud je PEQDEF uveden v deskriptoru zprávy, když je zpráva vložena.

Pokud je v cestě rozpoznání názvu fronty více než jedna definice, je výchozí perzistence převzata z hodnoty tohoto atributu v *první* definici v cestě v době volání MQPUT nebo MQPUT1 . To může být:

- Alias fronty
- Lokální fronta
- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta *DefXmitQName*).

Může mít jednu z následujících hodnot:

PEPER
Zpráva je trvalá.

To znamená, že zpráva přežije selhání systému a restartuje správce front. Trvalé zprávy nelze umístit na:

- Dočasné dynamické fronty
- Sdílené fronty

Trvalé zprávy lze umístit do trvalých dynamických front a do předdefinovaných front.

PENPER
Zpráva není trvalá.

To znamená, že zpráva obvykle nepřežije selhání systému nebo restartování správce front. To platí i v případě, že je během restartování správce front v pomocné paměti nalezena neporušená kopie zprávy.

Ve zvláštním případě sdílených front dočasné zprávy *přežijí* restarty správců front ve skupině sdílení front, ale nepřežijí selhání prostředku Coupling Facility používaného pro ukládání zpráv ve sdílených frontách.

Ve stejné frontě mohou existovat trvalé i přechodné zprávy.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADPER s voláním MQINQ.

IBM i *DefPriority (10místné celé číslo se znaménkem) na IBM i*

Výchozí priorita zprávy.

Tabulka 772. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Toto je výchozí priorita pro zprávy ve frontě. To platí v případě, že je PRQDEF uveden v deskriptoru zprávy, když je zpráva vložena do fronty.

Pokud je v cestě k rozlišení názvu fronty více než jedna definice, je výchozí priorita pro zprávu převzata z hodnoty tohoto atributu v *první* definici v cestě v době operace vložení. To může být:

- Alias fronty
- Lokální fronta
- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta *DefXmitQName*).

Způsob umístění zprávy do fronty závisí na hodnotě atributu **MsgDeliverySequence** fronty:

- Má-li atribut **MsgDeliverySequence** hodnotu MSPRIO, závisí logická pozice, na které je zpráva umístěna do fronty, na hodnotě pole *MDPRI* v deskriptoru zprávy.
- Je-li atribut **MsgDeliverySequence** MSFIFO, zprávy se umístí do fronty, jako by měly prioritu rovnající se *DefPriority* vyřešené fronty, bez ohledu na hodnotu pole *MDPRI* v deskriptoru zprávy. V poli *MDPRI* je však zachována hodnota určená aplikací, která zprávu vložila. Další informace viz atribut **MsgDeliverySequence** popsaný v části "Atributy pro fronty" na stránce 1360 .

Priority jsou v rozsahu nula (nejnižší) až *MaxPriority* (nejvyšší); viz atribut **MaxPriority** popsaný v části "Atributy pro správce front v systému IBM i" na stránce 1391.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADPRI s voláním MQINQ.

IBM i *DefReadpředčíslí (10místné celé číslo se znaménkem) na IBM i*

Určuje výchozí chování dopředného čtení pro dočasné zprávy doručené klientovi.

Tabulka 773. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X		

DefReadAhead lze nastavit na jednu z následujících hodnot:

RAHNO
Přechodné zprávy se klientovi neodesílají dopředu, dokud je aplikace nepožádá. Pokud klient skončí abnormálně, dojde ke ztrátě maximálně jedné netrvalé zprávy.

RAHYES
Přechodné zprávy jsou klientovi odesílány dopředu před tím, než je aplikace požádá. Dočasné zprávy mohou být ztraceny, pokud klient skončí nestandardně nebo pokud klient nespotřebuje všechny zprávy, které odeslal.

RAHDIS-informační systém
Čtení před dočasnou zprávou není pro tuto frontu povoleno. Zprávy se klientovi neodesílají dopředu bez ohledu na to, zda aplikace klienta požaduje dopředné čtení.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADRAH s voláním MQINQ.

IBM i *DefPResp (10místné celé číslo se znaménkem) na IBM i*

Atribut výchozího typu odezvy vložení (DEFPRESP) definuje hodnotu používanou aplikacemi, když byl typ PutResponsev rámci MQPMO nastaven na PMRASQ. Tento atribut je platný pro všechny typy front.

Tabulka 774. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Může mít jednu z následujících hodnot:

SYNC
Operace vložení je vydána synchronně a vrací odezvu.

ASYNC
Operace vložení je vydána asynchronně a vrací podmnožinu polí MQMD.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADPRT s voláním MQINQ.

IBM i *DistLists (10místné celé číslo se znaménkem) na IBM i*

Podpora distribučního seznamu.

Tabulka 775. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Označuje, zda lze zprávy rozdělovníku umístit do fronty. Atribut je nastaven agentem kanálu zpráv (MCA), aby informoval lokálního správce front, zda správce front na druhém konci kanálu podporuje distribuční seznamy. Tento druhý správce front (nazývaný "partnerský správce front") je ten, který poté, co byl odebrán z lokální přenosové fronty odesílajícím agentem MCA, obdrží zprávu.

Atribut je nastaven odesílajícím agentem MCA při každém vytvoření připojení k přijímajícímu adaptéru MCA v partnerském správci front. Tímto způsobem může odesílající agent MCA způsobit, že lokální správce front umístí do přenosové fronty pouze zprávy, které může správně zpracovat partnerský správce front.

Tento atribut je primárně určen pro použití s přenosovými frontami, ale popsané zpracování se provádí bez ohledu na použití definované pro frontu (viz atribut **Usage**).

Může mít jednu z následujících hodnot:

DLSUPP
Distribuční seznamy jsou podporovány.

To znamená, že zprávy distribučního seznamu mohou být uloženy ve frontě a přeneseny do partnerských správců front v tomto formátu. Tím se sníží množství zpracování potřebné k odeslání zprávy do více míst určení.

DLNSUP
Distribuční seznamy nejsou podporovány.

To znamená, že zprávy distribučního seznamu nelze uložit do fronty, protože partnerský správce front nepodporuje distribuční seznamy. Pokud aplikace vloží zprávu distribučního seznamu a tato zpráva má být umístěna do této fronty, správce front místo toho rozdělí zprávu distribučního seznamu a umístí jednotlivé zprávy do fronty. Tím se zvýší množství zpracování potřebné k odeslání zprávy do více míst určení, ale zajistí se, že budou zprávy správně zpracovány partnerským správcem front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADIST s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *HardenGetVrátit zpět (10místné celé číslo se znaménkem) na IBM i*

Zda se má udržovat přesný počet vrácení.

Tabulka 776. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Pro každou zprávu je zachován počet případů, kdy je zpráva načtena voláním MQGET v rámci pracovní jednotky, a tato jednotka práce byla později odvolána. Tento počet je k dispozici v poli *MDBOC* v deskriptoru zprávy po dokončení volání MQGET.

Počet vrácení zpráv přežije po restartování správce front. Chcete-li se však ujistit, že je počet přesný, musí být informace "napevno" (zaznamenány na disku nebo jiném zařízení trvalého úložiště) pokaždé, když je zpráva načtena voláním MQGET v rámci pracovní jednotky pro tuto frontu. Pokud se tak nestane a dojde-li k selhání správce front společně se zpětnou operací volání MQGET, nemusí se počet zvýšit.

Upřesňující informace pro každé volání MQGET v rámci pracovní jednotky však vyžadují náklady na výkon a atribut **HardenGetBackout** by měl být nastaven na hodnotu QABH pouze v případě, že má být počet přesný.

- V systému IBM ije počet vrácení zprávy vždy zpřesněn, bez ohledu na nastavení tohoto atributu.

Možné jsou následující hodnoty:

QABH
Počet vrácení byl zapamatován.

K zajištění přesnosti počtu vrácení pro zprávy v této frontě se používá zpřesnění.

QABNH
Počet vrácení nemusí být zapamatován.

Upřesnění se nepoužívá k zajištění přesnosti počtu vrácení pro zprávy v této frontě. Počet proto může být nižší, než by měl být.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAHGB s voláním MQINQ.

IBM i *InhibitGet (10místné celé číslo se znaménkem) na IBM i*

Určuje, zda jsou povoleny operace získání pro tuto frontu.

Tabulka 777. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X		

Pokud se jedná o alias fronty, operace get musí být povoleny pro alias i pro základní frontu v době operace get, aby bylo volání MQGET úspěšné. Hodnota je jedna z následujících:

QAGETI
Operace získání jsou zablokovány.

Volání MQGET se nezdaří s kódem příčiny RC2016. To zahrnuje volání MQGET, která uvádějí GMBRWF nebo GMBRWN.

Poznámka: Pokud se volání MQGET pracující v rámci pracovní jednotky úspěšně dokončí, změna hodnoty atributu **InhibitGet** po QAGETI nezabrání v potvrzení transakce.

QAGETA
Operace získání jsou povoleny.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAIGET s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *InhibitPut (10místné celé číslo se znaménkem) na IBM i*

Určuje, zda jsou povoleny operace vložení pro tuto frontu.

Tabulka 778. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Pokud je v cestě rozpoznání názvu fronty více než jedna definice, musí být operace vložení povoleny pro *každou* definici v cestě (včetně všech definic aliasů správce front) v době operace vložení, aby bylo volání MQPUT nebo MQPUT1 úspěšné. Může mít jednu z následujících hodnot:

QAPUTI
Operace vložení jsou zablokovány.

Volání MQPUT a MQPUT1 se nezdaří s kódem příčiny RC2051.

Poznámka: Pokud se volání MQPUT operující v rámci pracovní jednotky úspěšně dokončí, změna hodnoty atributu **InhibitPut** později na QAPUTI nezabrání potvrzení transakce.

QAPUTA
Operace vložení jsou povoleny.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAIPUT s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *InitiationQName (48bajtový znakový řetězec) na IBM i*

Název inicializační fronty.

Tabulka 779. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o název fronty definované v lokálním správci front. Fronta musí být typu QTLOC. Správce front odešle zprávu spouštěče do inicializační fronty v případě, že je vyžadováno spuštění aplikace v důsledku doručení zprávy do fronty, do které tento atribut patří. Inicializační fronta musí být monitorována aplikací monitoru spouštěčů, která spustí příslušnou aplikaci po přijetí zprávy spouštěče.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAINIQ s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

IBM i *MaxMsgDélka (10místné celé číslo se znaménkem) na IBM i*

Maximální délka zprávy v bajtech.

Tabulka 780. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o horní limit délky nejdelší *fyzické* zprávy, kterou lze umístit do fronty. Protože však lze atribut fronty **MaxMsgLength** nastavit nezávisle na atributu správce front **MaxMsgLength** , je skutečným horním limitem délky nejdelší fyzické zprávy, kterou lze umístit do fronty, nižší z těchto dvou hodnot.

Pokud správce front podporuje segmentaci, je možné, aby aplikace vložila *logickou* zprávu, která je delší než nižší z obou atributů **MaxMsgLength** , ale pouze v případě, že aplikace v deskriptoru MQMD určuje příznak MFSEGA. Je-li uveden tento příznak, horní limit pro délku logické zprávy je 999 999 999 bajtů, ale obvykle omezení prostředků zavedená operačním systémem nebo prostředím, ve kterém je aplikace spuštěna, má za následek dolní limit.

Pokus o umístění zprávy, která je příliš dlouhá, do fronty selže s kódem příčiny:

- RC2030 , pokud je zpráva pro frontu příliš velká
- RC2031 , pokud je zpráva příliš velká pro správce front, ale není příliš velká pro frontu

Dolní limit pro atribut **MaxMsgLength** je nula. Horní mez je určena prostředím:

- V systému IBM ije maximální délka zprávy 100 MB (104 857 600 bajtů).

Další informace viz parametr **BUFLEN** popsaný v části “MQPUT (Vložit zprávu) na IBM i” na stránce 1325.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMLEN s voláním MQINQ.

IBM i *MaxQDepth (10místné celé číslo se znaménkem) na IBM i*

Maximální hloubka fronty.

Tabulka 781. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Toto je definovaný horní limit pro počet fyzických zpráv, které mohou existovat ve frontě v libovolném okamžiku. Pokus o vložení zprávy do fronty, která již obsahuje zprávy *MaxQDepth* , selže s kódem příčiny RC2053.

Zpracování jednotky práce a segmentace zpráv mohou způsobit, že skutečný počet fyzických zpráv ve frontě překročí hodnotu *MaxQDepth*. To však nemá vliv na načítatelnost zpráv- *všechny* zprávy ve frontě lze načíst běžným způsobem pomocí volání MQGET.

Hodnota tohoto atributu je nula nebo větší. Horní mez je určena prostředím.

Poznámka: Je možné, aby byl úložný prostor dostupný pro frontu vyčerpán, i když je ve frontě méně než *MaxQDepth* zpráv.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMDEP s voláním MQINQ.

IBM i *MediaLog (10místné celé číslo se znaménkem) na IBM i*

Identita oblasti protokolu (nebo žurnálového zásobníku na systému IBM i) potřebné pro obnovu médií konkrétní fronty.

Tabulka 782. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

U správců front, kteří používají kruhové protokolování, je hodnota vrácena jako řetězec s hodnotou Null.

IBM i *MsgDeliveryPořadí (10místné celé číslo se znaménkem) na IBM i*

Sekvence doručení zpráv.

Tabulka 783. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Určuje pořadí, ve kterém jsou zprávy vraceny aplikaci pomocí volání MQGET:

MSFIFO
Zprávy jsou vráceny v pořadí FIFO (první dovnitř, první ven).

To znamená, že volání MQGET vrátí *první* zprávu, která splňuje kritéria výběru uvedená ve volání, bez ohledu na prioritu zprávy.

MSPRIO
Zprávy jsou vraceny v pořadí podle priority.

To znamená, že volání MQGET vrátí zprávu *s nejvyšší prioritou* , která splňuje kritéria výběru uvedená ve volání. V rámci každé úrovně priority jsou zprávy vraceny v pořadí FIFO (první dovnitř, první ven).

Pokud se příslušné atributy změní, když jsou ve frontě zprávy, pořadí doručení je následující:

- Pořadí, ve kterém jsou zprávy vraceny voláním MQGET, je určeno hodnotami atributů **MsgDeliverySequence** a **DefPriority** , které jsou platné pro frontu v době, kdy zpráva dorazí do fronty:

– Je-li *MsgDeliverySequence* MSFIFO při doručení zprávy, zpráva se umístí do fronty, jako by její priorita byla *DefPriority*. To nemá vliv na hodnotu pole *MDPRI* v deskriptoru zprávy; toto pole zachová hodnotu, kterou mělo při prvním vložení zprávy.
– Je-li *MsgDeliverySequence* MSPRIO, když zpráva dorazí, zpráva se umístí do fronty na místo odpovídající prioritě dané polem *MDPRI* v deskriptoru zprávy.

Pokud se hodnota atributu **MsgDeliverySequence** změní, když jsou ve frontě zprávy, pořadí zpráv ve frontě se nezmění.

Pokud se hodnota atributu **DefPriority** změní, když jsou ve frontě zprávy, zprávy nemusí být nutně doručeny v pořadí FIFO, i když je atribut **MsgDeliverySequence** nastaven na MSFIFO; ty, které byly umístěny do fronty s vyšší prioritou, jsou doručeny jako první.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMDS s voláním MQINQ.

IBM i *OpenInputPočet (10místné celé číslo se znaménkem) na IBM i*

Počet otevření pro vstup.

Tabulka 784. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Jedná se o počet popisovačů, které jsou aktuálně platné pro odebrání zpráv z fronty pomocí volání MQGET. Jedná se o celkový počet popisovačů známých *lokálnímu* správci front. Pokud se jedná o sdílenou frontu, počet nezahrnuje otevření pro vstup, který byl proveden pro frontu v jiných správcích front ve skupině sdílení front, do které lokální správce front patří.

Počet zahrnuje popisovače, kde byla pro vstup otevřena alias fronta, která se interpretuje na tuto frontu. Tento počet nezahrnuje popisovače, kde byla fronta otevřena pro akce, které nezahrnovaly vstup (například fronta otevřená pouze pro procházení).

Hodnota tohoto atributu kolísá v průběhu činnosti správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAOIC s voláním MQINQ.

IBM i *OpenOutputPočet (10místné celé číslo se znaménkem) na IBM i*

Počet otevření pro výstup.

Tabulka 785. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X				

Jedná se o počet popisovačů, které jsou aktuálně platné pro přidávání zpráv do fronty pomocí volání MQPUT. Jedná se o celkový počet popisovačů známých *lokálnímu* správci front; nezahrnuje otevření pro výstup, která byla provedena pro tuto frontu ve vzdálených správcích front. Pokud se jedná o sdílenou frontu, tento počet nezahrnuje otevření pro výstup, který byl proveden pro frontu v jiných správcích front ve skupině sdílení front, do které lokální správce front patří.

Tento počet zahrnuje popisovače, kde byla pro výstup otevřena alias fronta, která se interpretuje na tuto frontu. Počet nezahrnuje popisovače, kde byla fronta otevřena pro akce, které nezahrnovaly výstup (například fronta otevřená pouze pro dotazování).

Hodnota tohoto atributu kolísá v průběhu činnosti správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAOOC s voláním MQINQ.

IBM i *ProcessName (48bajtový znakový řetězec) na IBM i*

Název procesu.

Tabulka 786. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o název objektu procesu, který je definován v lokálním správci front. Objekt procesu identifikuje program, který může obsluhovat frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAPRON s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNPRON.

IBM i *QDepthHighUdálost (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda jsou generovány události Vysoká hloubka fronty.

Tabulka 787. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Událost Vysoká hloubka fronty označuje, že aplikace vložila zprávu do fronty, což způsobilo, že počet zpráv ve frontě byl větší nebo roven horní prahové hodnotě hloubky fronty (viz atribut **QDepthHighLimit**).

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

QDepthHighUdálost může mít jednu ze dvou hodnot:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDHE s voláním MQINQ.

IBM i *QDepthHighLimit (10místné celé číslo se znaménkem) na IBM i*

Horní limit pro hloubku fronty.

Tabulka 788. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o prahovou hodnotu, se kterou je porovnána hloubka fronty při generování události Vysoká hloubka fronty. Tato událost označuje, že aplikace vložila zprávu do fronty, což způsobilo, že počet zpráv ve frontě byl větší nebo roven horní prahové hodnotě hloubky fronty. Viz atribut **QDepthHighEvent** .

Hodnota je vyjádřena jako procentní část maximální hloubky fronty (atribut**MaxQDepth**) a je v rozsahu od 0 do 100. Výchozí hodnota je 80.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDHL s voláním MQINQ.

IBM i *QDepthLowUdálost (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda jsou generovány události dolní hloubky fronty.

Tabulka 789. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Událost Nízká hloubka fronty označuje, že aplikace načetla zprávu z fronty, což způsobilo, že počet zpráv ve frontě byl menší nebo roven dolní prahové hodnotě hloubky fronty (viz atribut **QDepthLowLimit**).

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Událost QDepthLowmůže mít jednu z následujících hodnot:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDLE s voláním MQINQ.

IBM i *QDepthLowLimit (10místné celé číslo se znaménkem) na IBM i*

Dolní limit pro hloubku fronty.

Tabulka 790. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Jedná se o prahovou hodnotu, se kterou je porovnána hloubka fronty při generování události dolní hloubky fronty. Tato událost označuje, že aplikace načetla zprávu z fronty, a to způsobilo, že počet zpráv ve frontě byl menší nebo roven dolní prahové hodnotě hloubky fronty. Viz atribut **QDepthLowEvent** .

Hodnota je vyjádřena jako procentní část maximální hloubky fronty (atribut**MaxQDepth**) a je v rozsahu od 0 do 100. Výchozí hodnota je 20.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDLL s voláním MQINQ.

IBM i *QDepthMaxUdálost (10místné celé číslo se znaménkem) na IBM i*

Určuje, zda jsou generovány události zaplnění fronty.

Tabulka 791. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Událost zaplnění fronty označuje, že vložení do fronty bylo odmítnuto, protože fronta je plná, to znamená, že hloubka fronty již dosáhla své maximální hodnoty.

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Může mít jednu z následujících hodnot:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDME s voláním MQINQ.

IBM i *QDesc (64bitový znakový řetězec) na systému IBM i*

Popis fronty.

Tabulka 792. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Toto je pole, které lze použít pro popisný komentář. Obsah pole nemá pro správce front žádný význam, ale správce front může vyžadovat, aby pole obsahovaly pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou obsaženy ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAQD s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQD.

IBM i *QName (48bajtový znakový řetězec) na IBM i*

Název fronty.

Tabulka 793. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Jedná se o název fronty definované v lokálním správci front. Další informace o názvech front naleznete v tématu Pravidla pro pojmenování IBM MQ objektů. Všechny fronty definované ve správci front sdílejí stejný obor názvů fronty. Proto fronta QTLOC a fronta QTALS nemohou mít stejný název.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAQN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

IBM i *QServiceInterval (10místné celé číslo se znaménkem) na IBM i*

Cíl pro interval služby fronty.

Tabulka 794. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Jedná se o interval služby použitý pro porovnání pro generování událostí servisního intervalu vysoké a servisního intervalu OK. Viz atribut **QServiceIntervalEvent** .

Hodnota je v jednotkách milisekund a je v rozsahu od 0 do 999 999 999 999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQSI s voláním MQINQ.

IBM i *QServiceIntervalUdálost (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda se generují události servisního intervalu vysokého nebo servisního intervalu OK.

Tabulka 795. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

- Událost vysokého servisního intervalu je generována, když kontrola označuje, že z fronty nebyly načteny žádné zprávy alespoň po dobu označenou atributem **`QServiceInterval`** .
- Událost OK servisního intervalu je generována, když kontrola označuje, že zprávy byly načteny z fronty v čase označeném atributem **`QServiceInterval`** .

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Tento atribut může mít některou z následujících hodnot:

QSIEHI
Události vysokého servisního intervalu fronty jsou povoleny.

- Události vysokého servisního intervalu fronty jsou **povoleny** a
- Události OK servisního intervalu fronty jsou **zakázány**.

QSIEOK
Události OK intervalu služby fronty jsou povoleny.

- Události vysokého servisního intervalu fronty jsou **zakázány** a
- Události OK intervalu služby fronty jsou **povoleny**.

QSIENO
Nejsou povoleny žádné události intervalu služby fronty.

- Události vysokého servisního intervalu fronty jsou **zakázány** a
- Události OK intervalu služby fronty jsou také **zakázány**.

Pro sdílené fronty se hodnota tohoto atributu ignoruje; předpokládá se hodnota QSIENO.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQSIE s voláním MQINQ.

IBM i *QSGDisp (10místné celé číslo se znaménkem) na IBM i*

Dispozice skupiny sdílení front.

Tabulka 796. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	

Tato volba určuje dispozici fronty. Hodnota je jedna z následujících:

QSGDQM
Dispozice správce front.

Objekt má dispozici správce front. To znamená, že definice objektu je známa pouze lokálnímu správci front; definice není známa ostatním správcům front ve skupině sdílení front.

Je možné, aby každý správce front ve skupině sdílení front měl objekt se stejným názvem a typem jako aktuální objekt, ale jedná se o samostatné objekty a neexistuje mezi nimi žádná korelace. Jejich atributy nejsou omezeny tak, aby byly stejné jako ostatní.

QSGDCP
Odebrání kopírovaného objektu.

Objekt je lokální kopií definice hlavního objektu, která existuje ve sdíleném úložišti. Každý správce front ve skupině sdílení front může mít svou vlastní kopii objektu. Na začátku mají všechny kopie stejné atributy, ale pomocí příkazů MQSC lze každou kopii změnit tak, aby se její atributy lišily od atributů ostatních kopií. Atributy kopií se znovu synchronizují, když se změní hlavní definice ve sdíleném úložišti.

QSGDSH
Sdílená dispozice.

Objekt má sdílenou dispozici. To znamená, že ve sdíleném úložišti existuje jediná instance objektu, která je známa všem správcům front ve skupině sdílení front. Když správce front ve skupině přistupuje k objektu, přistupuje k jediné sdílené instanci objektu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQSGD s voláním MQINQ.

z/OS Tento atribut je podporován pouze na systému z/OS.

IBM i *QType (10místné celé číslo se znaménkem) na IBM i*

Typ fronty.

Tabulka 797. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Tento atribut může mít některou z následujících hodnot:

QTALS
Definice alias fronty.

QTCLUS
Fronta klastru.

QTLOC
Lokální fronta.

QTREM
Lokální definice vzdálené fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQTYP s voláním MQINQ.

IBM i *RemoteQMgrNázev (48bajtový znakový řetězec) v systému IBM i*

Název vzdáleného správce front.

Tabulka 798. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
			X	

Jedná se o název vzdáleného správce front, ve kterém je fronta *RemoteQName* definována. Pokud má fronta *RemoteQName* hodnotu *QSGDisp* QSGDCP nebo QSGDSH, *RemoteQMgrName* může být název skupiny sdílení front, která vlastní *RemoteQName*.

Pokud aplikace otevře lokální definici vzdálené fronty, *RemoteQMgrName* nesmí být prázdné a nesmí být název lokálního správce front. Je-li hodnota *XmitQName* prázdná, použije se jako přenosová fronta lokální fronta se stejným názvem jako *RemoteQMgrName* . Pokud neexistuje žádná fronta s názvem *RemoteQMgrName*, použije se fronta určená atributem správce front **DefXmitQName** .

Pokud je tato definice použita pro alias správce front, *RemoteQMgrName* je název správce front, který je aliasem. Může se jednat o název lokálního správce front. Jinak, pokud je *XmitQName* při otevření prázdné, musí existovat lokální fronta se stejným názvem jako *RemoteQMgrName* ; tato fronta se používá jako přenosová fronta.

Je-li tato definice použita pro alias pro odpověď, jedná se o název správce front, kterým má být *MDRM*.

Poznámka: Při vytváření nebo úpravě definice fronty se na hodnotě uvedené pro tento atribut neprovede žádné ověření.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CARQMN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQMN.

IBM i *RemoteQName (48bajtový znakový řetězec) na IBM i*

Název vzdálené fronty.

Tabulka 799. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
			X	

Jedná se o název fronty tak, jak je znám ve vzdáleném správci front *RemoteQMgrName*.

Pokud aplikace otevře lokální definici vzdálené fronty, nesmí být hodnota *RemoteQName* při otevření prázdná.

Je-li tato definice použita pro definici aliasu správce front, musí být hodnota *RemoteQName* při otevření prázdná.

Pokud je definice použita pro alias odpovědi, tento název je název fronty, která má být *MDRQ*.

Poznámka: Při vytváření nebo úpravě definice fronty se na hodnotě uvedené pro tento atribut neprovede žádné ověření.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CARQN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

IBM i *RetentionInterval (10místné celé číslo se znaménkem) na IBM i*

Interval uchování.

Tabulka 800. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Jedná se o čas, kdy by měla být fronta zachována. Po uplynutí této doby je fronta vhodná pro odstranění.

Čas se měří v hodinách, počítáno od data a času, kdy byla fronta vytvořena. Datum vytvoření fronty je zaznamenáno v souboru *CreationDate* a čas vytvoření fronty je zaznamenán v atributu **CreationTime** .

Tyto informace jsou poskytnuty, aby umožnily úklidové aplikaci nebo operátorovi identifikovat a odstranit fronty, které již nejsou požadovány.

Poznámka: Správce front se nikdy nepokouší odstranit fronty založené na tomto atributu nebo zabránit odstranění front s intervalem uchování, jehož platnost dosud nevypršela. Je odpovědností uživatele, aby provedl jakoukoli požadovanou akci.

Chcete-li zabránit hromadění trvalých dynamických front, je třeba použít realistický interval uchování (viz *DefinitionType*). Tento atribut však lze také použít s předdefinovanými frontami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IARINT s voláním MQINQ.

IBM i *Rozsah (10místné celé číslo se znaménkem) na IBM i*

Určuje, zda položka pro tuto frontu existuje také v adresáři buňky.

Tabulka 801. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Adresář buňky je poskytován instalovatelnou službou názvů. Může mít jednu z následujících hodnot:

SCOQM-řízení kvality
Obor správce front.

Definice fronty má obor správce front. To znamená, že definice fronty nepřekračuje rámec správce front, který ji vlastní. Chcete-li otevřít frontu pro výstup z jiného správce front, musí být zadán buď název vlastnícího správce front, nebo jiný správce front musí mít lokální definici fronty.

SCOCEL
Rozsah buňky.

Definice fronty má rozsah buňky. To znamená, že definice fronty je také umístěna do adresáře buňky, který je k dispozici všem správcům front v buňce. Frontu lze otevřít pro výstup z libovolného správce front v buňce pouze zadáním názvu fronty. Není třeba zadávat název správce front, který frontu vlastní. Definice fronty však není k dispozici žádnému správci front v buňce, která má také lokální definici fronty s tímto názvem, protože lokální definice má přednost.

Adresář buňky je poskytován instalovatelnou službou názvů, například LDAP (Lightweight Directory Access Protocol). Všimněte si, že produkt IBM MQ již nepodporuje službu názvů DCE (Distributed Computing Environment), která byla dříve používána pro vkládání definic front do adresáře DCE (také již není podporována).

Modelové a dynamické fronty nemohou mít rozsah buňky.

Tato hodnota je platná pouze v případě, že byla konfigurována služba názvů podporující adresář buňky.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASCOP s voláním MQINQ.

Podpora tohoto atributu podléhá následujícím omezením:

- V systému IBM ije atribut podporován, ale pouze SCOQM je platný.

IBM i *Možnost sdílení (10místné celé číslo se znaménkem) na IBM i*

Zda lze frontu sdílet pro vstup.

Tabulka 802. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Označuje, zda lze frontu otevřít pro vstup vícekrát souběžně. Může mít jednu z následujících hodnot:

QASHR
Frontu lze sdílet.

Vícenásobná otevření s volbou OOINPS jsou povolena.

QANSHR
Frontu nelze sdílet.

Volání MQOPEN s volbou OOINPS je považováno za volání OOINPX.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASHAR s voláním MQINQ.

IBM i *TriggerControl (10místné celé číslo se znaménkem) na IBM i*

Ovládací prvek spouštěče.

Tabulka 803. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

To řídí, zda se zprávy spouštěče zapisují do inicializační fronty, aby se aplikace spustila pro obsluhu fronty. Jedná se o jednu z následujících možností:

TCOFF
Zprávy spouštěče nejsou požadovány.

Pro tuto frontu se nemají zapisovat žádné zprávy spouštěče. Hodnota *TriggerType* je v tomto případě irelevantní.

TCON
Jsou vyžadovány zprávy spouštěče.

Zprávy spouštěče se zapisují pro tuto frontu, když dojde k příslušným událostem spouštěče.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGC s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *TriggerData (64bitový znakový řetězec) na IBM i*

Data spouątěče.

Tabulka 804. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Jedná se o data ve volném formátu, která správce front vloží do zprávy spouštěče, když zpráva přicházející do této fronty způsobí zápis zprávy spouštěče do inicializační fronty.

Obsah těchto dat nemá pro správce front žádný význam. Má význam buď pro aplikaci monitoru spouštěčů, která zpracovává inicializační frontu, nebo pro aplikaci, která je spuštěna monitorem spouštěčů.

Znakový řetězec nesmí obsahovat žádné hodnoty null. V případě potřeby je vyplněna vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CATRGD s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET. Délka tohoto atributu je dána hodnotou LNTRGD.

IBM i *TriggerDepth (10místné celé číslo se znaménkem) na IBM i*

Hloubka spouątěče.

Tabulka 805. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o počet zpráv s prioritou *TriggerMsgPriority* nebo vyšší, které musí být ve frontě, než se zapíše zpráva spouštěče. Toto platí, když je parametr *TriggerType* nastaven na TTDPTH. Hodnota *TriggerDepth* je jedna nebo větší. Tento atribut není jinak použit.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGD s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *TriggerMsgPriorita (10místné celé číslo se znaménkem) na IBM i*

Prahová hodnota priority zpráv pro spouštěče na systému IBM MQ for IBM i.

Tabulka 806. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Toto je priorita zprávy, pod kterou zprávy nepřispívají ke generování zpráv spouštěče (to znamená, že správce front tyto zprávy ignoruje při určování, zda má být zpráva spouštěče generována). Hodnota *TriggerMsgPriority* může být v rozsahu nula (nejnižší) až *MaxPriority* (nejvyšší; viz “Atributy pro správce front v systému IBM i” na stránce 1391); hodnota nula způsobí, že všechny zprávy přispějí ke generování zpráv spouštěče.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGP s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *TriggerType (10místné celé číslo se znaménkem) na IBM i*

Typ spouątěče.

Tabulka 807. Typy front, na které se tento atribut vztahuje				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

To řídí podmínky, za kterých jsou zprávy spouštěče zapisovány jako výsledek zpráv přicházejících do této fronty. Hodnota je jedna z následujících:

TTNONE
Žádné zprávy spouštěče.

V důsledku zpráv v této frontě nejsou zapsány žádné zprávy spouštěče. To má stejný účinek jako nastavení parametru *TriggerControl* na hodnotu TCOFF.

TTFRST
Spustit zprávu, když je hloubka fronty od 0 do 1.

Zpráva spouštěče se zapíše vždy, když se počet zpráv s prioritou *TriggerMsgPriority* nebo vyšší ve frontě změní z 0 na 1.

TTEVRY
Spustit zprávu pro každou zprávu.

Zpráva spouštěče se zapíše vždy, když do fronty dorazí zpráva s prioritou *TriggerMsgPriority* nebo vyšší.

TTDPTH
Spustit zprávu, když je překročena prahová hodnota hloubky.

Zpráva spouštěče se zapíše vždy, když se počet zpráv s prioritou *TriggerMsgPriority* nebo vyšší ve frontě rovná nebo překročí hodnotu *TriggerDepth*. Po zapsání zprávy spouštěče je parametr *TriggerControl* nastaven na hodnotu TCOFF, aby se zabránilo dalšímu spouštění, dokud nebude znovu explicitně zapnut.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGT s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *Použití (10místné celé číslo se znaménkem) na IBM i*

Použití fronty.

Tabulka 808. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Označuje, k čemu se fronta používá. Hodnota je jedna z následujících:

USNORM (Spojené státy)
Normální použití.

Jedná se o frontu, kterou používají běžné aplikace při vkládání a získávání zpráv. Fronta není přenosovou frontou.

USTRAN-další
Přenosová fronta.

Jedná se o frontu používanou k ukládání zpráv určených pro vzdálené správce front. Když normální aplikace odešle zprávu do vzdálené fronty, lokální správce front uloží zprávu dočasně do příslušné přenosové fronty ve speciálním formátu. Agent kanálu zpráv poté přečte zprávu z přenosové fronty a přenese ji do vzdáleného správce front. Další informace o přenosových frontách naleznete v tématu Přenosová fronta.

Pouze privilegované aplikace mohou otevřít přenosovou frontu pro OOOUT, aby do ní mohly přímo vkládat zprávy. Obvykle se očekává, že to budou provádět pouze obslužné aplikace. Je třeba dbát na to, aby byl formát dat zprávy správný (viz “MQXQH (záhlaví přenosové fronty) na IBM i” na stránce 1239), jinak by se během procesu přenosu mohly vyskytnout chyby. Kontext není předán nebo nastaven, pokud není zadána jedna z voleb kontextu PM*.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAUSAG s voláním MQINQ.

IBM i *XmitQName (48bajtový znakový řetězec) na IBM i*

Název přenosové fronty.

Tabulka 809. Typy front, na které se tento atribut vztahuje

Lokální	Model	Alias	Vzdálený	Klastr
			X	

Je-li tento atribut neprázdný, když dojde k otevření, buď pro vzdálenou frontu, nebo pro definici aliasu správce front, uvádí název lokální přenosové fronty, která se má použít pro postoupení zprávy.

Je-li hodnota *XmitQName* prázdná, použije se jako přenosová fronta lokální fronta se stejným názvem jako *RemoteQMgrName* . Pokud neexistuje žádná fronta s názvem *RemoteQMgrName*, použije se fronta určená atributem správce front **DefXmitQName** .

Tento atribut je ignorován, pokud se definice používá jako alias správce front a *RemoteQMgrName* je název lokálního správce front. Také se ignoruje tehdy, jestliže se definice pouľívá jako definice alias odpovídací fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAXQN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

Atributy pro seznamy názvů

Toto téma shrnuje atributy, které jsou specifické pro seznamy názvů. Atributy jsou popsány v abecedním pořadí.

Poznámka: Zobrazené názvy atributů jsou názvy použité s voláními MQINQ a MQSET.

Popisy atributů

Objekt seznamu názvů má následující atributy:

AlterationDate (12bajtový znakový řetězec)

Datum, kdy byla definice naposledy změněna.

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby se délka 12 bajtů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTD s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNDATE.

AlterationTime (8bajtový znakový řetězec)

Čas, kdy byla definice naposledy změněna.

Toto je čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTT s voláním MQINQ. Délka tohoto atributu je dána LNTIME.

NameCount (10místné celé číslo se znaménkem)

Počet jmen v seznamu názvů.

Tato hodnota je větší nebo rovna nule. Je definována následující hodnota:

NCMXNL
Maximální počet názvů v seznamu názvů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IANAMC s voláním MQINQ.

NamelistDesc (64bitový znakový řetězec)

Popis seznamu názvů.

Jedná se o pole, které může být použito pro popisný komentář; jeho hodnota je stanovena definičním procesem. Obsah pole nemá pro správce front žádný význam, ale správce front může vyžadovat, aby pole obsahovaly pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může toto pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou obsaženy ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CALSTD s voláním MQINQ.

Délka tohoto atributu je dána LNNLD.

NamelistName (48bajtový znakový řetězec)

Název seznamu názvů.

Jedná se o název seznamu názvů, který je definován v lokálním správci front.

Každý seznam názvů má název, který se liší od názvů jiných seznamů názvů náležejících správci front, ale může duplikovat názvy jiných objektů správců front různých typů (například front).

Chcete-li určit hodnotu tohoto atributu, použijte selektor CALSTN s voláním MQINQ.

Délka tohoto atributu je dána hodnotou LNNLN.

Názvy (48bajtový znakový řetězec x NameCount)

Seznam názvů *NameCount* .

Každý název je název objektu, který je definován pro lokálního správce front. Další informace o názvech objektů naleznete v tématu Pojmenování IBM MQ objektů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CANAMS s voláním MQINQ.

Délka každého názvu v seznamu je dána hodnotou LNOBJN.

IBM i Atributy pro definice procesů v systému IBM i

Toto téma shrnuje atributy, které jsou specifické pro definice procesu. Atributy jsou popsány v abecedním pořadí.

Poznámka: Zobrazené názvy atributů jsou názvy použité s voláními MQINQ a MQSET. Při použití příkazů MQSC k definování, změně nebo zobrazení atributů jsou použity alternativní krátké názvy. Podrobnosti naleznete v části Příkazy MQSC .

Popisy atributů

Objekt definice procesu má následující atributy:

AlterationDate (12bajtový znakový řetězec)

Datum, kdy byla definice naposledy změněna.

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby se délka 12 bajtů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTD s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNDATE.

AlterationTime (8bajtový znakový řetězec)

Čas, kdy byla definice naposledy změněna.

Toto je čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTT s voláním MQINQ. Délka tohoto atributu je dána LNTIME.

ApplId (256bajtový znakový řetězec)

Identifikátor aplikace.

Jedná se o znakový řetězec, který identifikuje aplikaci, která má být spuštěna. Tyto informace jsou určeny pro použití aplikací monitoru spouštěčů, která zpracovává zprávy v inicializační frontě. Informace se odesílají do inicializační fronty jako součást zprávy spouštěče.

Význam parametru *ApplId* je určen aplikací pro monitorování spouštěčů. Monitor spouštěčů poskytovaný produktem IBM MQ vyžaduje, aby *ApplId* byl název spustitelného programu.

Znakový řetězec nesmí obsahovat žádné hodnoty null. V případě potřeby je vyplněna vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAAPPI s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNPROA.

ApplType (10místné celé číslo se znaménkem)

Typ aplikace.

Označuje povahu programu, který se má spustit jako odpověď na přijetí zprávy spouštěče. Tyto informace jsou určeny pro použití aplikací monitoru spouštěčů, která zpracovává zprávy v inicializační frontě. Informace se odesílají do inicializační fronty jako součást zprávy spouštěče.

ApplType může mít libovolnou hodnotu. Pro standardní typy můžete použít následující hodnoty; typy aplikací definované uživatelem jsou omezeny na hodnoty v rozsahu ATUFST až ATULST:

rovnoCICS
CICS transakce.

AT400
IBM i .

ATUFST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

ATULST
Nejvyšší hodnota pro typ aplikace definovaný uživatelem.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAAPPT s voláním MQINQ.

EnvData (128bajtový znakový řetězec)

Data prostředí.

Jedná se o znakový řetězec, který obsahuje informace související s prostředím týkající se aplikace, která má být spuštěna. Tyto informace jsou určeny pro použití aplikací monitoru spouštěčů, která zpracovává zprávy v inicializační frontě. Informace se odesílají do inicializační fronty jako součást zprávy spouštěče.

Význam parametru *EnvData* je určen aplikací pro monitorování spouštěčů. Monitor spouštěčů poskytovaný pomocí IBM MQ připojení *EnvData* k seznamu parametrů předaný spuštěné aplikaci. Seznam parametrů se skládá ze struktury MQTMC2 , následované jednou mezerou, následovanou *EnvData* s odebranými koncovými mezerami.

Znakový řetězec nesmí obsahovat žádné hodnoty null. V případě potřeby je vyplněna vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAENVD s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNPROE.

ProcessDesc (64bitový znakový řetězec)

Popis procesu.

Toto je pole, které lze použít pro popisný komentář. Obsah pole nemá pro správce front žádný význam, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou obsaženy ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAPROD s voláním MQINQ.

Délka tohoto atributu je dána LNPROD.

ProcessName (48bajtový znakový řetězec)

Název procesu.

Jedná se o název definice procesu, která je definována v lokálním správci front.

Každá definice procesu má název, který se liší od názvů ostatních definic procesů náležejících správci front. Název definice procesu však může být stejný jako názvy jiných objektů správce front různých typů (například front).

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAPRON s voláním MQINQ.

Délka tohoto atributu je dána hodnotou LNPRON.

UserData (128bajtový znakový řetězec)

Uživatelská data.

Jedná se o znakový řetězec, který obsahuje informace o uživateli týkající se aplikace, která má být spuštěna. Tyto informace jsou určeny pro použití aplikací monitoru spouštěčů, která zpracovává zprávy v inicializační frontě, nebo aplikací, která je spuštěna monitorem spouštěčů. Informace se odešlou do inicializační fronty jako součást zprávy spouštěče.

Význam parametru *UserData* je určen aplikací pro monitorování spouštěčů. Monitor spouštěčů poskytovaný produktem IBM MQ předává *UserData* spuštěné aplikaci jako součást seznamu parametrů. Seznam parametrů se skládá ze struktury MQTMC2 (obsahující *UserData*), následované jednou mezerou, následovanou *EnvData* s odebranými koncovými mezerami.

Znakový řetězec nesmí obsahovat žádné hodnoty null. V případě potřeby je vyplněna vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAUSRD s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNPROU.

IBM i Atributy pro správce front v systému IBM i

Souhrn atributů správce front.

Některé atributy správce front jsou opraveny pro konkrétní implementace, zatímco jiné lze změnit pomocí příkazu MQSC `ALTER QMGR`. Atributy lze také zobrazit pomocí příkazu `DISPLAY QMGR`. Většinu atributů správce front lze získat otevřením speciálního objektu OTQM a použitím volání MQINQ s vráceným popisovačem.

Následující tabulka shrnuje atributy, které jsou specifické pro správce front. Atributy jsou popsány v abecedním pořadí.

Poznámka: Názvy atributů zobrazené v této sekci jsou názvy použité s voláními MQINQ a MQSET. Při použití příkazů MQSC k definování, změně nebo zobrazení atributů jsou použity alternativní krátké názvy. Další informace naleznete v tématu Příkazy MQSC .

Tabulka 810. Atributy pro správce front

Atribut	Popis
AlterationDate	Datum, kdy byla definice naposledy změněna
AlterationTime	Čas, kdy byla definice naposledy změněna
AuthorityEvent	Řídí, zda jsou generovány události autorizace (bez autorizace)
BridgeEvent	Řídí, zda jsou generovány události mostu IMS
ChannelAutoDef	Určuje, zda je povolena automatická definice kanálu.
ChannelAutoDefEvent	Řídí, zda jsou generovány události automatické definice kanálu
ChannelAutoDefExit	Název uživatelské procedury pro automatickou definici kanálu
ChannelEvent	Řídí, zda jsou generovány události kanálu
ClusterCache	Určuje, zda má mezipaměť klastru pevnou velikost nebo zda má dynamickou velikost.
ClusterWorkloadData	Uživatelská data pro uživatelskou proceduru pracovní zátěže klastru

Tabulka 810. Atributy pro správce front (pokračování)	
Atribut	**Popis**
ClusterWorkloadExit	Název uživatelské procedury pro správu pracovní zátěže klastru
ClusterWorkloadLength	Maximální délka dat zprávy předaných uživatelské proceduře pracovní zátěže klastru
CodedCharSetId	Identifikátor znakové sady
CommandEvent	Řídí, zda jsou zprávy událostí příkazu zařazeny do fronty
CommandInputQName	Název fronty vstupu příkazů
CommandLevel	Úroveň příkazů
ConfigurationEvent	Událost konfigurace
DeadLetterQName	Název fronty nedoručených zpráv
DefClusterXmitQueue	Výchozí typ přenosové fronty klastru
DefXmitQName	Výchozí název přenosové fronty
DistLists	Podpora seznamu distribuce
InhibitEvent	Řídí, zda jsou generovány události blokování (Inhibit Get a Inhibit Put)
LocalEvent	Řídí, zda jsou generovány lokální chybové události
LoggerEvent	Řídí, zda jsou generovány události protokolu pro zotavení
MaxHandles	Maximální počet popisovačů
MaxMsgLength	Maximální délka zprávy v bajtech
MaxPriority	Maximální priorita
MaxUncommittedMsgs	Maximální počet nepotvrzených zpráv v rámci pracovní jednotky
PerformanceEvent	Řídí, zda jsou generovány události související s výkonem
Platforma	Platforma, na které je spuštěn správce front
PubSubMode	Zda je spuštěn stroj publikování/odběru a rozhraní publikování/odběru ve frontě
QMgrDesc	Popis správce front
QMgrIdentifier	Jedinečný interně generovaný identifikátor správce front
QMgrName	Název správce front
RemoteEvent	Řídí, zda jsou generovány vzdálené chybové události
RepositoryName	Název klastru, pro který tento správce front poskytuje služby úložiště
RepositoryNamelist	Název objektu seznamu názvů obsahujícího názvy klastrů, pro které tento správce front poskytuje služby úložiště
SSLCRLNamelist	Název objektu seznamu názvů obsahujícího názvy objektů ověřovacích informací (viz poznámka 1).
SSLEvent	Řídí, zda jsou generovány události TLS
SSLKeyRepository	Umístění úložiště klíčů TLS (viz poznámka 1)
SSLKeyResetPočet	Určuje počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS před opětným vyjednáním šifrovacího klíče

Tabulka 810. Atributy pro správce front (pokračování)	
Atribut	**Popis**
StartStopEvent	Řídí, zda jsou generovány události spuštění a zastavení
SyncPoint	Dostupnost synchronizačního bodu
TraceRouteRecording	Řídí záznam informací o trasovací trase pro zprávy
TreeLifeTime	Doba životnosti neadministrativních témat (v sekundách)
TriggerInterval	Interval zpráv spouštěče

Notes:

1. Tento atribut nelze zjišťovat pomocí volání MQINQ a není popsán v této sekci. Další informace o tomto atributu naleznete v tématu Změnit správce front.

IBM i *AlterationDate (12bajtový znakový řetězec) na IBM i*

Datum, kdy byla definice naposledy změněna.

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby se délka 12 bajtů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTD s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNDATE.

IBM i *AlterationTime (8bajtový znakový řetězec) na IBM i*

Čas, kdy byla definice naposledy změněna.

Toto je čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTT s voláním MQINQ. Délka tohoto atributu je dána LNTIME.

IBM i *AuthorityEvent (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda se generují události autorizace (bez autorizace).

Atribut AuthorityEvent musí být nastaven na jednu z následujících hodnot:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAAUTE s voláním MQINQ.

IBM i *BridgeEvent (znakový řetězec) na IBM i*

Tento atribut určuje, zda jsou zprávy událostí mostu IMS vloženy do SYSTEM.ADMIN.CHANNEL.EVENT . Je podporován pouze na systému z/OS.

IBM i *ChannelAutoDef (10místné celé číslo se znaménkem) na IBM i*

Určuje, zda je povolena automatická definice kanálu.

Tento atribut řídí automatickou definici kanálů typu CTRCVR a CTSVCN. Všimněte si, že automatická definice kanálů CTCLSD je vždy povolena. Může mít jednu z následujících hodnot:

CHADDI
Automatická definice kanálu je zakázána.

CHADEN
Automatická definice kanálu je povolena.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACAD s voláním MQINQ.

IBM i *ChannelAutoDefEvent (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda jsou generovány události automatické definice kanálu.

Toto platí pro kanály typu CTRCVR, CTSVCN a CTCLSD. Může mít jednu z následujících hodnot:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování a výkon.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACADE s voláním MQINQ.

IBM i *ChannelAutoDefExit (20bajtový znakový řetězec) na IBM i*

Název uživatelské procedury pro automatickou definici kanálu.

Je-li tento název neprázdný a parametr *ChannelAutoDef* má hodnotu CHADEN, bude uživatelská procedura volána pokaždé, když se správce front chystá vytvořit definici kanálu. Toto platí pro kanály typu CTRCVR, CTSVCN a CTCLSD. Ukončení pak může provést jednu z následujících možností:

- Chcete-li pokračovat beze změny, povolte vytvoření definice kanálu.
- Upravte atributy vytvořené definice kanálu.
- Zcela potlačit vytvoření kanálu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACADX s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNEXN.

IBM i *ChannelEvent (znakový řetězec) na IBM i*

Určuje, zda jsou generovány zprávy událostí kanálu.

Tento atribut určuje, zda jsou zprávy událostí kanálu vkládány do SYSTEM.ADMIN.CHANNEL.EVENT queue, a pokud ano, jaký typ zpráv je ve frontě (například 'channel started', 'channel stopped', 'channel not activated'). Před implementací tohoto atributu byl jediným způsobem, jak zabránit zařazení zpráv událostí kanálu do fronty, odstranění cílové fronty.

Tento atribut vám také umožňuje shromažďovat pouze události mostu IMS (protože nyní můžete vypnout události kanálu, nedostanou se do stejné fronty). Totéž platí pro události TLS, které mohou být také shromážděny bez nutnosti shromažďovat také události kanálu.

Tento atribut vám také umožňuje shromažďovat pouze významné události (například když mají kanály chyby, ne když se spouštějí a zastavují normálně).

Hodnota atributu ChannelEvent může být jedna z následujících:

- EVREXP (jsou generovány pouze následující události kanálu: RC2279, RC2283, RC2284, RC2295, RC2296).
- EVRENA (všechny události kanálu jsou generovány; to znamená, že kromě událostí generovaných systémem EVREXP jsou generovány také události RC2282a RC2283).
- EVRDIS (nejsou generovány žádné události kanálu; jedná se o počáteční výchozí hodnotu správce front).

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACHNE s voláním MQINQ.

IBM i *ClusterCacheTyp (32bajtový znakový řetězec) na IBM i*

Určuje, zda má mezipaměť klastru pevnou velikost, nebo zda má dynamickou velikost.

Jedná se o uživatelem definovaný 32bajtový znakový řetězec, který je při volání předán uživatelské proceduře pracovní zátěže klastru. Pokud neexistují žádná data, která by bylo možné předat uživatelské proceduře, řetězec je prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACLWD s voláním MQINQ.

IBM i *ClusterWorkloadData (32bajtový znakový řetězec) na IBM i*

Uživatelská data pro ukončení pracovní zátěže klastru.

Jedná se o uživatelem definovaný 32bajtový znakový řetězec, který je při volání předán uživatelské proceduře pracovní zátěže klastru. Pokud neexistují žádná data, která by bylo možné předat uživatelské proceduře, řetězec je prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACLWD s voláním MQINQ.

IBM i *ClusterWorkloadUkončit (20bajtový znakový řetězec) na IBM i*

Název uživatelské procedury pro správu pracovní zátěže klastru.

Není-li tento název prázdný, je uživatelská procedura volána pokaždé, když je zpráva vložena do fronty klastru nebo přesunuta z jedné odesílací fronty klastru do jiné. Uživatelská procedura pak může buď přijmout instanci fronty vybranou správcem front jako místo určení pro zprávu, nebo vybrat jinou instanci fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACLWX s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNEXN.

IBM i *ClusterWorkloadDélka (10místné celé číslo se znaménkem) na IBM i*

Maximální délka dat zprávy předaných uživatelské proceduře pracovní zátěže klastru.

Jedná se o maximální délku dat zprávy, která jsou předána uživatelské proceduře pracovní zátěže klastru. Skutečná délka dat předaných do ukončení je minimální z následujících hodnot:

- Délka zprávy.
- Atribut **MaxMsgLength** správce front.
- Atribut **ClusterWorkloadLength** .

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACLWL s voláním MQINQ.

IBM i *CodedCharSetId (10místné celé číslo se znaménkem) na IBM i*

Identifikátor kódované znakové sady.

Tato volba definuje znakovou sadu používanou správcem front pro všechna pole řetězce znaků definovaná v rozhraní MQI, například názvy objektů a datum a čas vytvoření fronty. Znaková sada musí být taková, která má jednobajtové znaky pro znaky platné v názvech objektů. Nevztahuje se na data aplikace přenášené ve zprávě. Hodnota závisí na prostředí:

- V systému IBM ije hodnota nastavena v prostředí při prvním vytvoření správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACCSI s voláním MQINQ.

IBM i *CommandEvent (celé číslo) na IBM i*

Určuje, zda jsou při zadávání příkazů zprávy vkládány do lokální fronty.

Tato volba určuje, zda jsou zprávy zapisovány do nové fronty událostí SYSTEM.ADMIN.COMMAND.EVENT, kdykoli jsou zadány příkazy. Tato funkce je užitečná pro oznámení sledování příkazů a pro diagnostiku problémů. Chcete-li se dotázat na atribut správce front CommandEvent , použijte nový selektor atributů iacev s jednou z následujících hodnot:

- EVRENA-zprávy o událostech příkazů jsou generovány a vkládány do fronty pro všechny úspěšné příkazy.

- Zprávy událostí příkazu EVND jsou generovány a vkládány do fronty pro všechny úspěšné příkazy kromě příkazu DISPLAY (MQSC) a příkazu Inquire (PCF).
- EVRDIS-zprávy o událostech příkazu nejsou generovány nebo vkládány do fronty (jedná se o počáteční výchozí hodnotu správce front).

Chcete-li určit hodnotu tohoto atributu, použijte selektor CMDEV s voláním MQINQ.

IBM i *CommandInputQName (48bajtový znakový řetězec) na IBM i*

Název vstupní fronty příkazu.

CommandInputQName je název vstupní fronty příkazů definované v lokálním správci front. Jedná se o frontu, do které mohou uživatelé odesílat příkazy, pokud k tomu mají oprávnění. Název fronty závisí na prostředí:

- V systému IBM ije název fronty SYSTEM.ADMIN.COMMAND.QUEUEa lze do ní odesílat pouze příkazy PCF. Příkaz MQSC však může být odeslán do této fronty, pokud je příkaz MQSC uzavřen v příkazu PCF typu CMESC. Další informace o příkazu Escape viz Escape.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACMDQ s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

IBM i *CommandLevel (10místné celé číslo se znaménkem) na IBM i*

Úroveň příkazu. To označuje úroveň řídicích příkazů systému podporovaných správcem front.

Úroveň je jedna z následujících hodnot:

CML800
Úroveň 800 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími aplikacemi:

- IBM MQ for IBM i
  - V8.0

CML900
Úroveň 900 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími aplikacemi:

- IBM MQ for IBM i
  - V9.0

CML910
Úroveň 910 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími aplikacemi:

- IBM MQ for IBM i
  - verze 9.1

CML920
Úroveň 920 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími aplikacemi:

- IBM MQ for IBM i
  - verze 9.2

CML930
Úroveň 930 příkazů pro řízení systému.

Tato hodnota je vrácena následujícími aplikacemi:

- IBM MQ for IBM i

– Verze 9.3

Sada řídicích příkazů systému, která odpovídá konkrétní hodnotě atributu **CommandLevel** , se liší v závislosti na hodnotě atributu **Platform** ; obojí musí být použito k rozhodnutí, které řídicí příkazy systému jsou podporovány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACMDL s voláním MQINQ.

IBM i *ConfigurationEvent na IBM i*

Řídí, zda se události konfigurace generují a odesílají do SYSTEM.ADMIN.CONFIG.EVENT .

Atribut ConfigurationEvent může mít jednu z následujících hodnot:

- EVRENA
- EVRDIS

Je-li atribut ConfigurationEvent nastaven na hodnotu EVRENA a některé příkazy jsou úspěšně zadány příkazem runmqsc nebo PCF, jsou události konfigurace generovány a odeslány do SYSTEM.ADMIN.CONFIG.EVENT . Události pro následující příkazy jsou vydány, i když příkaz alter nezmění zahrnutý objekt. Příkazy, pro které jsou události konfigurace generovány a odesílány, jsou:

- DEFINE/ALTER AUTHINFO
- DEFINE/ALTER CHANNEL
- DEFINE/ALTER NAMELIST
- DEFINE/ALTER PROCESS
- DEFINE/ALTER QLOCAL (pokud se nejedná o dočasnou dynamickou frontu)
- DEFINE/ALTER QMODEL/QALIAS/QREMOTE
- DELETE AUTHINFO (ODSTRANĚNÍ)
- Odstranit kanál
- Odstranit seznam názvů
- Odstranit proces
- DELETE QLOCAL (pokud se nejedná o dočasnou dynamickou frontu)
- ODSTRANĚNÍ QMODEL/QALIAS/QREMOTE
- ALTER QMGR (pokud není atribut CONFIGEV zakázán a není změněn na povolený)
- AKTUALIZOVAT SPRÁVCE FRONT
- Volání MQSET jiné než pro dočasnou dynamickou frontu.

Události nejsou generovány (jsou-li povoleny) za následujících okolností:

- Příkaz nebo volání MQSET selhává.
- Správce front nemůže vložit zprávu události do fronty událostí. Příkaz by měl být i nadále úspěšně dokončen.
- Dočasné dynamické fronty.
- Změny interních atributů provedené přímo nebo implicitně (nikoli pomocí příkazu MQSET nebo příkazu); to ovlivňuje TRIGGER, CURDEPTH, IPPROCS, OPPROCS, QDPHIEV, QDPLOEV, QDPMAXEV, QSVCIEV.
- Když se změní fronta událostí konfigurace, i když se pro tuto změnu vygeneruje zpráva události, když se požaduje aktualizace.
- Klastrování změn pomocí příkazů REFRESH/RESET CLUSTER a RESUME/SUSPEND QMGR.
- Vytvoření nebo odstranění správce front.

IBM i *DeadLetterQName (48bajtový znakový řetězec) na IBM i*

Název fronty nedoručených zpráv.

Jedná se o název fronty definované v lokálním správci front. Zprávy jsou odesílány do této fronty, pokud je nelze směrovat na správné místo určení.

Zprávy jsou například vloženy do této fronty, když:

- Zpráva dorazí do správce front určeného pro frontu, která dosud není v daném správci front definována.
- Zpráva dorazí do správce front, ale fronta, pro kterou je určena, ji nemůže přijmout, protože je možné, že:
  - Fronta je plná
  - Požadavky na vložení jsou zablokovány
  - Odesílající uzel nemá oprávnění vkládat zprávy do fronty.

Aplikace mohou také vkládat zprávy do fronty nedoručených zpráv.

Se zprávami sestavy se zachází stejně jako s běžnými zprávami; pokud zprávu sestavy nelze doručit do cílové fronty (obvykle do fronty určené polem *MDRQ* v deskriptoru původní zprávy), zpráva sestavy se umístí do fronty nedoručených zpráv (nedoručených zpráv).

Poznámka: Zprávy, které překročily svůj čas vypršení platnosti (viz pole *MDEXP* popsané v části "MQMD (deskriptor zprávy) na IBM i" na stránce 1105). **nejsou** přeneseny do této fronty, když jsou vyřazeny. Zpráva sestavy vypršení platnosti (ROEXP) je však stále generována a odeslána do fronty *MDRQ* , pokud to vyžaduje odesílající aplikace.

Zprávy nejsou vloženy do fronty nedoručených zpráv (nedoručených zpráv) v případě, že aplikace, která vydala požadavek na vložení, byla synchronně upozorněna na problém s kódem příčiny vráceným voláním MQPUT nebo MQPUT1 (například zpráva vložená do lokální fronty, pro kterou jsou požadavky na vložení blokována).

Zprávy ve frontě nedoručených zpráv mají někdy svá data zpráv aplikace s předponou se strukturou MQDLH. Tato struktura obsahuje další informace, které označují, proč byla zpráva umístěna do fronty nedoručených zpráv (nedoručených zpráv). Další podrobnosti o této struktuře viz "MQDLH (záhlaví nedoručených dopisů) na IBM i" na stránce 1061 .

Tato fronta musí být lokální fronta s atributem **Usage** USNORM.

Není-li fronta nedoručených zpráv (nedoručených zpráv) podporována správcem front nebo není-li definována, je název prázdný. Všichni správci front IBM MQ podporují frontu nedoručených zpráv (nedoručených zpráv), ale standardně není definována.

Není-li fronta nedoručených zpráv definována nebo je plná nebo nepoužitelná z jiného důvodu, je zpráva, která by do ní byla přenesena agentem kanálu zpráv, uchována v přenosové frontě.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CADLQ s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

DefClusterXmitQueueTyp (10místné celé číslo se znaménkem)

Atribut `DefClusterXmitQueue` řídí, která přenosová fronta je standardně vybrána odesílacími kanály klastru pro získání zpráv, pro odeslání zpráv přijímacím kanálům klastru.

Hodnoty **`DefClusterXmitQueueType`** jsou `MQCLXQ_SCTQ` nebo `MQCLXQ_CHANNEL`.

`MQCLXQ_SCTQ`

Všechny odesílací kanály klastru odesílají zprávy z produktu `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. `correlID` zpráv uvedený v přenosové frontě identifikuje, pro který odesílací kanál klastru je zpráva určena.

Parametr `SCTQ` je nastaven při definování správce front.

`MQCLXQ_CHANNEL`

Každý odesílací kanál klastru posílá zprávy z různých přenosových front. Každá přenosová fronta je vytvořena jako trvalá dynamická fronta z modelové fronty `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`.

Pokud je atribut správce front DefClusterXmitQueueTypenastaven na hodnotu CHANNEL, Výchozí konfigurace se změnila na odesílací kanály klastru přidružené k jednotlivým přenosovým frontám klastru. Přenosové fronty jsou trvalé dynamické fronty vytvořené z modelové fronty SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE. Každá přenosová fronta je přidružená k jednomu odesílacímu kanálu klastru. Protože přenosovou frontu klastru obsluhuje jeden odesílací kanál klastru, obsahuje přenosová fronta zprávy pouze pro jednoho správce front v jednom klastru. Klastry můžete nakonfigurovat tak, aby každý správce front z klastru obsahoval pouze jednu frontu klastru. V takovém případě se zprávy ze správce front budou do každé fronty klastru přenášet odděleně od zpráv do jiných front.

Chcete-li zadat dotaz na hodnotu, zavolejte funkci MQINQnebo odešlete příkaz PCF správce front Inquire (MQCMD_INQUIRE_Q_MGR), nastavte selektor MQIA_DEF_CLUSTER_XMIT_Q_TYPE . Chcete-li změnit hodnotu, odešlete příkaz Změnit správce front (MQCMD_CHANGE_Q_MGR) PCF a nastavte selektor MQIA_DEF_CLUSTER_XMIT_Q_TYPE .

Související odkazy

Změnit správce front

Zjistit správce front

“MQINQ (dotazovat se na atributy objektu) na IBM i” na stránce 1298
Volání MQINQ vrací pole celých čísel a sadu znakových řetězců obsahujících atributy objektu.

IBM i *DefXmitQName (48bajtový znakový řetězec) na IBM i*

Výchozí název přenosové fronty.

Jedná se o název přenosové fronty, která se používá pro přenos zpráv do vzdálených správců front, pokud neexistuje jiná indikace, kterou přenosovou frontu použít.

Pokud neexistuje žádná výchozí přenosová fronta, název je zcela prázdný. Počáteční hodnota tohoto atributu je prázdná.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CADXQN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQN.

IBM i *DistLists (10místné celé číslo se znaménkem) na IBM i*

Podpora distribučního seznamu.

Označuje, zda lokální správce front podporuje distribuční seznamy ve voláních MQPUT a MQPUT1 . Může mít jednu z následujících hodnot:

DLSUPP
Distribuční seznamy jsou podporovány.

DLNSUP
Distribuční seznamy nejsou podporovány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADIST s voláním MQINQ.

IBM i *InhibitEvent (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda jsou generovány události blokování (Inhibit Get a Inhibit Put).

Může mít jednu z následujících hodnot:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování a výkon.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAINHE s voláním MQINQ.

IBM i *LocalEvent (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda jsou generovány lokální chybové události.

Hodnota je jedna z následujících:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí .

Chcete-li určit hodnotu tohoto atributu, použijte selektor IALCLE s voláním MQINQ.

IBM i *LoggerEvent (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda se generují události modulu protokolování zotavení.

Může mít jednu z následujících hodnot:

POVOLENO
Události modulu protokolování jsou generovány.

VYPNUTO
Události modulu protokolování nejsou generovány. Jedná se o počáteční výchozí hodnotu správce front.

Další informace o událostech viz Monitorování a výkon.

IBM i *MaxHandles (10místné celé číslo se znaménkem) na IBM i*

Maximální počet popisovačů.

Jedná se o maximální počet otevřených popisovačů, které může kterákoli úloha souběžně používat. Každé úspěšné volání MQOPEN pro jednu frontu (nebo pro objekt, který není frontou) používá jeden popisovač. Tento popisovač bude po zavření objektu k dispozici pro opětovné použití. Při otevření distribučního seznamu je však každé frontě v distribučním seznamu přidělen samostatný manipulátor, takže volání MQOPEN používá stejný počet manipulátorů jako fronty v distribučním seznamu. To je třeba vzít v úvahu při rozhodování o vhodné hodnotě pro *MaxHandles*.

Volání MQPUT1 provádí v rámci svého zpracování volání MQOPEN; v důsledku toho produkt MQPUT1 používá tolik manipulátorů, kolik by měl MQOPEN, ale manipulátory se používají pouze po dobu trvání samotného volání MQPUT1 .

Hodnota je v rozsahu 1 až 999 999 999 999. V systému IBM ije výchozí hodnota 256.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMHND s voláním MQINQ.

IBM i *MaxMsgDélka (10místné celé číslo se znaménkem) na IBM i*

Maximální délka zprávy v bajtech.

Jedná se o délku nejdelší *fyzické* zprávy, kterou může správce front zpracovat. Protože však lze atribut správce front **MaxMsgLength** nastavit nezávisle na atributu fronty **MaxMsgLength** , nejdelší fyzická zpráva, kterou lze umístit do fronty, je nižší z těchto dvou hodnot.

Pokud správce front podporuje segmentaci, je možné, aby aplikace vložila *logickou* zprávu, která je delší než nižší z obou atributů **MaxMsgLength** , ale pouze v případě, že aplikace v deskriptoru MQMD určuje příznak MFSEGA. Je-li uveden tento příznak, horní limit pro délku logické zprávy je 999 999 999 bajtů, ale obvykle, omezení prostředků uložená operačním systémem nebo prostředím, ve kterém je aplikace spuštěna, bude mít za následek dolní limit.

Dolní limit pro atribut **MaxMsgLength** je 32 kB (32 768 bajtů). V systému IBM ije maximální délka zprávy 100 MB (104 857 600 bajtů).

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMLEN s voláním MQINQ.

IBM i *MaxPriority (10místné celé číslo se znaménkem) na IBM i*

Maximální priorita.

Jedná se o maximální prioritu zpráv podporovanou správcem front. Priority jsou v rozsahu od nuly (nejnižší) do *MaxPriority* (nejvyšší).

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMPRI s voláním MQINQ.

IBM i *MaxUncommittedPočet zpráv (10místné celé číslo se znaménkem) IBM i*

Maximální počet nepotvrzených zpráv v rámci pracovní jednotky.

Jedná se o maximální počet nepotvrzených zpráv, které mohou existovat v rámci pracovní jednotky. Počet nepotvrzených zpráv je součtem následujících od spuštění aktuální pracovní jednotky:

- Zprávy vkládané aplikací s volbou PMSYP
- Zprávy načtené aplikací s volbou GMSYP
- Zprávy spouštěče a zprávy sestavy COA generované správcem front pro zprávy vkládané s volbou PMSYP
- Zprávy sestavy COD generované správcem front pro zprávy načtené pomocí volby GMSYP

Následující zprávy se nepočítají jako nepotvrzené:

- Zprávy vkládané nebo načtené aplikací mimo pracovní jednotku
- Zprávy spouštěče nebo zprávy sestavy COA/COD generované správcem front v důsledku zpráv vložených nebo načtených mimo pracovní jednotku
- Zprávy sestavy vypršení platnosti generované správcem front (i v případě, že volání způsobující zprávu sestavy vypršení platnosti uvádí GMSYP)
- Zprávy událostí generované správcem front (i v případě, že volání způsobilo, že zpráva události určila PMSYP nebo GMSYP).

Poznámka:

1. Zprávy sestavy výjimek jsou generovány agentem MCA (Message Channel Agent) nebo aplikací, a jsou tedy zpracovávány stejným způsobem jako běžné zprávy vkládané nebo načítané aplikací.
2. Když je zpráva nebo segment vložen s volbou PMSYP, počet nepotvrzených zpráv se zvýší o jednu, bez ohledu na to, kolik fyzických zpráv je skutečně výsledkem vložení. (Pokud správce front potřebuje zprávu nebo segment dále rozdělit, může dojít k více než jedné fyzické zprávě.)
3. Když je distribuční seznam vložen s volbou PMSYP, počet nepotvrzených zpráv se zvýší o jeden *pro každou vygenerovanou fyzickou zprávu*. To může být tak malý jako jeden, nebo stejně velký jako počet destinací v distribučním seznamu.

Dolní limit pro tento atribut je 1; horní limit je 999 999 999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMUNC s voláním MQINQ.

IBM i *PerformanceEvent (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda jsou generovány události související s výkonem.

PerformanceEvent může mít jednu z následujících hodnot:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAPFME s voláním MQINQ.

IBM i *Platforma (10místné celé číslo se znaménkem) na IBM i*

Platforma, na které je spuštěn správce front.

To označuje operační systém, na kterém je spuštěn správce front. Hodnota je:

PL400
IBM i.

IBM i *PubSub(10místné celé číslo se znaménkem) na IBM i*

Zda je spuštěn stroj publikování/odběru a rozhraní publikování/odběru ve frontě, což umožňuje aplikacím publikovat/odebírat pomocí rozhraní API a front, které jsou monitorovány rozhraním publikování/odběru ve frontě.

Může mít jednu z následujících hodnot:

PSMCP
Stroj pro publikování/odběr je spuštěn. Proto je možné publikovat/přihlásit se k odběru pomocí rozhraní API. Rozhraní publikování/odběru ve frontě není spuštěno, proto není žádná zpráva vložená do front, které jsou monitorovány rozhraním publikování/odběru ve frontě, zpracovávána. Toto nastavení se používá pro kompatibilitu s produktem WebSphere Message Broker V6 nebo s dřívějšími verzemi, které používají tohoto správce front, protože musí číst stejné fronty, ze kterých běžně čte rozhraní publikování/odběru zařazené ve frontě.

PSMDS
Stroj pro publikování/odběr a rozhraní pro publikování/odběr ve frontě nejsou spuštěny. Proto není možné publikovat/odebírat pomocí rozhraní API. Žádné zprávy publikování/odběru, které jsou vkládány do front, které jsou monitorovány rozhraním pro publikování/odběr ve frontě, se nepoužívají.

PSMEN
Stroj publikování/odběru a rozhraní publikování/odběru ve frontě jsou spuštěny. Proto je možné publikovat/odebírat pomocí rozhraní API a front, které jsou monitorovány rozhraním publikování/odběru zařazeným do fronty. Jedná se o počáteční výchozí hodnotu správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor PSMODE s voláním MQINQ.

IBM i *QMgrDesc (64bitový znakový řetězec) na IBM i*

Popis správce front.

Toto je pole, které lze použít pro popisný komentář. Obsah pole nemá pro správce front žádný význam, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může toto pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou obsaženy ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

V systému IBM ije výchozí hodnota prázdná.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAQMD s voláním MQINQ. Délka tohoto atributu je dána LNQMD.

IBM i *QMgrIdentifier (48bajtový znakový řetězec) na IBM i*

Jedinečný interně generovaný identifikátor správce front.

Jedná se o interně generovaný jedinečný název pro správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAQMID s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQMID.

IBM i *QMgrName (48bajtový znakový řetězec) na IBM i*

Název správce front.

Jedná se o název lokálního správce front, tj. název správce front, ke kterému je aplikace připojena.

Prvních 12 znaků názvu se používá k vytvoření jedinečného identifikátoru zprávy (viz pole *MDMID* popsané v části “MQMD (deskriptor zprávy) na IBM i” na stránce 1105). Správci front, kteří mohou komunikovat, proto musí mít názvy, které se liší v prvních 12 znacích, aby byly identifikátory zpráv v síti správců front jedinečné.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAQMN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQMN.

IBM i *RemoteEvent (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda jsou generovány vzdálené chybové události.

Hodnota je jedna z následujících:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IARMTE s voláním MQINQ.

IBM i *RepositoryName (48bajtový znakový řetězec) na IBM i*

Název klastru, pro který tento správce front poskytuje služby úložiště.

Jedná se o název klastru, pro který tento správce front poskytuje službu správce úložiště. Pokud správce front poskytuje tuto službu pro více než jeden klastr, parametr *RepositoryNamelist* určuje název objektu seznamu názvů, který identifikuje klastry, a parametr *RepositoryName* je prázdný. Alespoň jeden z *RepositoryName* a *RepositoryNamelist* musí být prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CARPN s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNQMN.

IBM i *RepositoryNamelist (48bajtový znakový řetězec) na IBM i*

Název objektu seznamu názvů obsahujícího názvy klastrů, pro které tento správce front poskytuje služby úložiště.

Jedná se o název objektu seznamu názvů, který obsahuje názvy klastrů, pro které tento správce front poskytuje službu správce úložiště. Pokud správce front poskytuje tuto službu pouze pro jeden klastr, obsahuje objekt seznamu názvů pouze jeden název. Alternativně lze *RepositoryName* použít k určení názvu klastru, v takovém případě je *RepositoryNamelist* prázdný. Alespoň jeden z *RepositoryName* a *RepositoryNamelist* musí být prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CARPNL s voláním MQINQ. Délka tohoto atributu je dána hodnotou LNNLN.

IBM i *SSLEvent (znakový řetězec) na IBM i*

Určuje, zda jsou generovány události TLS.

Hodnota je jedna z následujících:

- EVRENA (událost MQINQ/PCF/config) ENABLED (MQSC): Jsou generovány události TLS (tj. je generována událost RC2371).
- EVRDIS (událost MQINQ/PCF/config) DISABLED (MQSC): Události TLS nejsou generovány. Jedná se o počáteční výchozí hodnotu správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASSLE s voláním MQINQ.

IBM i *SSLKeyResetPočet (celé číslo) na IBM i*

Určuje celkový počet nešifrovaných bajtů, které jsou odeslány a přijaty v rámci konverzace TLS, než je znovu vyjednán tajný klíč. Počet bajtů zahrnuje řídicí informace odeslané agentem kanálu zpráv (MCA).

Tuto hodnotu používají pouze certifikační autority kanálu TLS, které zahajují komunikaci z tohoto správce front (tj. odesílací kanál MCA v párování odesílacího a přijímacího kanálu).

Pokud je hodnota tohoto atributu větší než 0 a pro kanál jsou povoleny prezenční signály kanálu, je tajný klíč také znovu vyjednán před odesláním nebo přijetím dat po synchronizačním signálu kanálu. Počet bajtů do doby, než se po každém úspěšném opětném vyjednávání vynuluje další opětovné vyjednávání tajného klíče.

Hodnota může být v rozsahu 0 až 999 999 999 999. Hodnota 0 pro tento atribut označuje, že tajný klíč není nikdy znovu vyjednán. Zadáte-li počet resetů tajného klíče TLS v rozsahu 1 bajt až 32 kB, budou kanály TLS používat počet resetů tajného klíče 32 kB. Tím se vyhnete nákladům na zpracování nadměrných resetů klíčů, které by se vyskytly pro malé hodnoty resetu tajného klíče TLS.

Je-li serverem SSL správce front IBM MQ a je-li povolen reset tajného klíče i synchronizační signály kanálu, dojde k opětovnému vyjednávání ihned po každém synchronizačním signálu kanálu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASSRC s voláním MQINQ.

IBM i *StartStopUdálost (10místné celé číslo se znaménkem) na IBM i*

Řídí, zda jsou generovány události spuštění a zastavení.

Tento atribut může mít některou z následujících hodnot:

EVRDIS
Hlášení událostí zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech viz Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASSE s voláním MQINQ.

IBM i *SyncPoint (10místné celé číslo se znaménkem) na IBM i*

Dostupnost synchronizačního bodu.

Označuje, zda lokální správce front podporuje pracovní jednotky a synchronizaci s voláními MQGET, MQPUT a MQPUT1 .

SPAVL
K dispozici jsou jednotky práce a synchronizace.

SPNAVL
Pracovní jednotky a synchronizační funkce nejsou k dispozici.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASYNC s voláním MQINQ.

IBM i *TraceRouteZáznam (10místné celé číslo se znaménkem) na IBM i*

Tato volba určuje, zda mají být při průchodu správcem front zaznamenávány informace o zprávách.

Hodnota je jedna z následujících:

- RECDD: Není povoleno připojení ke zprávám trasy trasování
- RECDQ: zprávy jsou vkládány do pevné pojmenované fronty
- RECDM: určení pomocí zprávy (toto je výchozí nastavení)

Chcete-li zabránit tomu, aby zpráva trasovací trasy zůstala v systému, nastavte na ní hodnotu vypršení platnosti, která je větší než nula, a uveďte volbu sestavy RODISC. Chcete-li zabránit tomu, aby zprávy sestavy nebo odpovědi zůstaly v systému, nastavte volbu sestavy ROPDAE. Další informace viz “Volby sestavy a příznaky zpráv na IBM i” na stránce 1425.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGI s voláním MQINQ.

IBM i *TreeLifeČas (10místné celé číslo se znaménkem) na IBM i*

Doba životnosti neadministrativních témat v sekundách.

Neadministrativní témata jsou ta, která jsou vytvořena, když aplikace publikuje nebo odebírá jako řetězec tématu, který neexistuje jako administrativní uzel. Tento parametr určuje, jak dlouho bude správce front čekat, než tento neadministrativní uzel odebere v případě, že již nebude obsahovat žádné aktivní odběry. Po recyklaci správce front jsou zachována pouze neadministrativní témata, která jsou používána trvalým odběrem.

Zadejte hodnotu v rozsahu 0 až 604 000. Hodnota 0 znamená, že správce front neadministrativní témata neodebírá. Počáteční výchozí hodnota správce front je 1800.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRLFT s voláním MQINQ.

IBM i *TriggerInterval (10místné celé číslo se znaménkem) na IBM i*

Interval zpráv spouštěče.

Jedná se o časový interval (v milisekundách) používaný k omezení počtu zpráv spouštěče. To je relevantní pouze v případě, že *TriggerType* je TTFRST. V tomto případě jsou zprávy spouštěče obvykle generovány pouze tehdy, když do fronty dorazí vhodná zpráva a fronta byla dříve prázdná. Za určitých okolností však může být generována další zpráva spouštěče se spouštěcím TTFRST i v případě, že fronta nebyla prázdná. Tyto další zprávy spouštěče se negenerují častěji než každých *TriggerInterval* milisekund.

Další informace o spouštění naleznete v tématu Spouštěcí kanály.

Hodnota je v rozsahu od 0 do 999 999 999 999. Výchozí hodnota je 999 999 999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGI s voláním MQINQ.

Aplikace

Tyto informace popisují ukázkové programy dodávané s produktem IBM MQ for IBM i pro RPG. Také se dozvíte, jak sestavit spustitelné aplikace z programů, které píšete.

Sestavení aplikace

Publikace IBM i popisují, jak sestavit spustitelné aplikace z programů, které píšete. Toto téma popisuje další úlohy a změny standardních úloh, které musíte provést při sestavování aplikací IBM MQ for IBM i , které mají být spuštěny v produktu IBM i.

Kromě kódování volání MQI ve zdrojovém kódu musíte přidat příslušné jazykové příkazy, které budou zahrnovat kopírované soubory IBM MQ for IBM i pro jazyk RPG. Měli byste se seznámit s obsahem těchto souborů; jejich jména a stručný popis jejich obsahu jsou uvedeny v následujícím textu.

IBM i *IBM MQ kopírovat soubory na IBM i*

Produkt IBM MQ for IBM i poskytuje soubory kopie, které vám pomohou se zápisem aplikací v programovacím jazyku RPG. Jsou vhodné pro použití s kompilátorem ILE RPG 4 Compiler sady vývojových nástrojů WebSphere (5722 WDS).

Soubory kopie, které produkt IBM MQ for IBM i poskytuje jako asistenci při psaní uživatelských procedur kanálu, jsou popsány v tématu Programy uživatelských procedur kanálu pro kanály systému zpráv.

Názvy souborů kopie IBM MQ for IBM i pro RPG mají předponu CMQ. Mají příponu G nebo H. Existují samostatné kopírované soubory obsahující pojmenované konstanty a jeden soubor pro každou ze struktur. Soubory kopie jsou uvedeny v seznamu “Jazykové aspekty” na stránce 1005.

Poznámka: Pro ILE RPG/400jsou dodávány jako členy souboruQRPGLESRC v knihovně QMQM.

Deklarace struktury neobsahují příkazy DS . To umožňuje aplikaci deklarovat datovou strukturu (nebo datovou strukturu s více výskyty) kódováním příkazu DS a použitím příkazu /COPY ke zkopírování ve zbytku deklarace:

Pro ILE RPG/400 je příkaz:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7
D* Declare an MQMD data structure
D MQMD       DS
D/COPY CMQMDG
```

Příprava programů ke spuštění

Chcete-li vytvořit spustitelnou aplikaci IBM MQ for IBM i , musíte zkompilovat zdrojový kód, který jste napsali.

Chcete-li to provést pro ILE RPG/400, můžete použít typické příkazy IBM i , CRTRPGMOD a CRTPGM.

Po vytvoření *MODULE musíte zadat BNDSRVPGM(QMQM/LIBMQM) v příkazu CRTPGM. To zahrnuje různé procedury IBM MQ ve vašem programu.

Při provádění kompilace se ujistěte, že knihovna obsahující kopírované soubory (QMQM) je v seznamu knihoven.

Další informace týkající se aspektů programování, včetně režimů klienta, viz "Jazykové aspekty" na stránce 1005.

Rozhraní pro externího správce synchronizačního bodu IBM i

IBM MQ for IBM i používá nativní IBM i vázané zpracování jako externí koordinátor synchronizačních bodů.

Další informace o možnostech vázaného zpracování produktu IBM inaleznete v příručce *IBM i Programming: Backup and Recovery Guide* .

Chcete-li spustit prostředky vázaného zpracování IBM i , použijte systémový příkaz STRCMTCTL. Chcete-li ukončit vázané zpracování, použijte systémový příkaz ENDCMTCTL.

Poznámka: Výchozí hodnota *Rozsah definice závazku* je *ACTGRP. Toto musí být definováno jako *JOB pro IBM MQ pro IBM i. Příklad:

```
STRCMTCTL LCKLVL(*ALL) CMTSCOPE(*JOB)
```

Pokud voláte MQPUT, MQPUT1nebo MQGET a zadáte PMSYP nebo GMSYP, po spuštění vázaného zpracování se IBM MQ for IBM i přidá jako prostředek vázaného zpracování rozhraní API do definice vázaného zpracování. Toto je obvykle první takové volání v úloze. I když existují nějaké prostředky vázaného zpracování rozhraní API registrované pod konkrétní definicí vázaného zpracování, nemůžete ukončit vázané zpracování pro tuto definici.

Produkt IBM MQ for IBM i odebírá svou registraci jako prostředek vázaného zpracování rozhraní API při odpojení od správce front za předpokladu, že v aktuální transakci nejsou žádné nevyřízené operace MQI.

Pokud se odpojíte od správce front v době, kdy existují nevyřízené operace MQPUT, MQPUT1nebo MQGET v aktuální transakci, produkt IBM MQ for IBM i zůstane registrován jako prostředek vázaného zpracování rozhraní API, aby byl upozorněn na další potvrzení nebo odvolání. Po dosažení dalšího synchronizačního bodu produkt IBM MQ potvrdí nebo odvolá změny podle potřeby. Je možné, aby se aplikace odpojila a znovu připojila ke správci front během aktivní pracovní jednotky a provedla další operace MQGET a MQPUT ve stejné pracovní jednotce (jedná se o nevyřízené odpojení).

Pokud se pokusíte vydat systémový příkaz ENDCMTCTL pro tuto definici vázaného zpracování, vydá se zpráva CPF8355 , která označuje, že nevyřízené změny byly aktivní. Tato zpráva se také objeví v protokolu úlohy po ukončení úlohy. Chcete-li se tomu vyhnout, ujistěte se, že jste potvrdili nebo odvolali všechny nevyřízené operace IBM MQ a že jste se odpojili od správce front. Proto by použití příkazů COMMIT nebo ROLLBACK před příkazem ENDCMTCTL mělo umožnit úspěšné dokončení vázaného zpracování.

Je-li řízení vázaného zpracování IBM i používáno jako externí koordinátor synchronizačních bodů, nemusí být volání MQCMIT, MQBACK a MQBEGIN vydána. Volání těchto funkcí se nezdaří s kódem příčiny RC2012.

Chcete-li potvrdit nebo odvolat (tj. odvolat) svou pracovní jednotku, použijte jeden z programovacích jazyků, který podporuje vázané zpracování. Příklad:

- Příkazy CL: COMMIT a ROLLBACK
- Funkce programování v C ILE: _Rcommit a _Rrollback
- RPG/400: COMMIT a ROLBK
- COBOL/400: COMMIT a ROLLBACK

Synchronizační body v CICS pro aplikace IBM i

IBM MQ for IBM i se podílí na jednotkách práce s CICS. Rozhraní MQI v rámci aplikace CICS můžete použít k vložení a získání zpráv v rámci aktuální pracovní jednotky.

K zavedení synchronizačního bodu, který zahrnuje operace IBM MQ for IBM i , můžete použít příkaz EXEC CICS SYNCPOINT. Chcete-li vrátit všechny změny až do předchozího synchronizačního bodu, můžete použít příkaz EXEC CICS SYNCPOINT ROLLBACK.

Používáte-li volbu MQPUT, MQPUT1nebo MQGET s volbou PMSYP nebo GMSYP nastavenou v aplikaci CICS , nemůžete se odhlásit CICS , dokud produkt IBM MQ for IBM i neodebere svou registraci jako prostředek vázaného zpracování rozhraní API. Proto byste měli před odpojením od správce front potvrdit nebo vrátit zpět všechny nevyřízené operace vložení nebo získání. To vám umožní odhlásit se od CICS.

Ukázkové programy na IBM i

Toto téma popisuje ukázkové programy dodávané s produktem IBM MQ for IBM i pro RPG. Ukázky demonstrují typická použití rozhraní MQI (Message Queue Interface).

Ukázky nejsou určeny k demonstraci obecných programovacích technik, takže některé kontroly chyb, které byste mohli chtít zahrnout do produkčního programu, byly vynechány. Tyto ukázky jsou však vhodné pro použití jako základ pro vlastní programy pro řazení zpráv do front.

Zdrojový kód všech ukázek je poskytován s produktem; tento zdroj obsahuje komentáře, které vysvětlují techniky řazení zpráv do front, které jsou demonstrovány v programech.

Existuje jedna sada ukázkových programů ILE:

1. **Programy používající protokolová volání rozhraní MQI (statická vázaná volání)**

   Zdroj existuje v QMQMSAMP/QRPGLESRC. Členové mají název AMQ3xxx4, kde xxx označuje ukázkovou funkci. V QMQM/QRPGLESRC existují členové kopie. Každý název člena má příponu G nebo H.

Tabulka 811 na stránce 1407 poskytuje úplný seznam ukázkových programů dodávaných s produktem IBM MQ for IBM ia zobrazuje názvy programů v jednotlivých podporovaných programovacích jazycích. Všimněte si, že všechny jejich názvy začínají předponou AMQ, čtvrtý znak v názvu označuje programovací jazyk.

Tabulka 811. Názvy ukázkových programů

	RPG (ILE)
Vložit vzorky	AMQ3PUT4
Procházet ukázky	AMQ3GBR4
Získat ukázky	AMQ3GET4
Požadovat ukázky	AMQ3REQ4
Ukázky ozvěny	AMQ3ECH4

Tabulka 811. Názvy ukázkových programů (pokračování)	
	RPG (ILE)
Vyšetřovat vzorky	AMQ3INQ4
Nastavit ukázky	AMQ3SET4
Ukázka monitoru spouštěčů	AMQ3TRG4
Ukázka spouštěcího serveru	AMQ3SRV4

Kromě těchto možností obsahuje ukázková volba IBM MQ for IBM i ukázkový datový soubor AMQSDATA, který lze použít jako vstup pro určité ukázkové programy a ukázkové CL programy, které demonstrují úlohy administrace. Ukázky CL jsou popsány v tématu Administrace IBM i . Můžete použít ukázkový CL program k vytvoření front pro použití s ukázkovými programy popsanými v tomto tématu.

Informace o tom, jak spustit ukázkové programy, viz "Příprava a spuštění ukázkových programů na systému IBM i" na stránce 1409.

Funkce demonstrované v ukázkových programech na IBM i

Tabulka, která zobrazuje techniky demonstrované ukázkovými programy IBM MQ for IBM i .

Některé techniky se vyskytují ve více než jednom ukázkovém programu, ale v tabulce je uveden pouze jeden program. Všechny ukázky otevírají a zavírají fronty pomocí volání MQOPEN a MQCLOSE, takže tyto techniky nejsou v tabulce uvedeny odděleně.

Tabulka 812. Ukázkové programy předvádějící použití rozhraní MQI	
Technika	**RPG (ILE)**
Použití volání MQCONN a MQDISC	AMQ3ECH4 nebo AMQ3INQ4
Implicitní připojení a odpojení	AMQ3PUT4
Vkládání zpráv pomocí volání MQPUT	AMQ3PUT4
Vložení jedné zprávy pomocí volání MQPUT1	AMQ3ECH4 nebo AMQ3INQ4
Odpověď na zprávu požadavku	AMQ3INQ4
Získávání zpráv (bez čekání)	AMQ3GBR4
Získávání zpráv (čekat s časovým limitem)	AMQ3GET4
Získávání zpráv (s převodem dat)	AMQ3ECH4
Procházení fronty	AMQ3GBR4
Použití sdílené vstupní fronty	AMQ3INQ4
Použití výlučné vstupní fronty	AMQ3REQ4
Použití volání MQINQ	AMQ3INQ4
Použití volání MQSET	AMQ3SET4
Použití fronty pro odpověď	AMQ3REQ4
Požadování zpráv výjimek	AMQ3REQ4
Přijetí zkrácené zprávy	AMQ3GBR4
Použití rozlišeného názvu fronty	AMQ3GBR4

Tabulka 812. Ukázkové programy předvádějící použití rozhraní MQI (pokračování)	
Technika	**RPG (ILE)**
Zpracování spouštěče	AMQ3SRV4 nebo AMQ3TRG4

Poznámka: Všechny ukázkové programy vytvoří soubor pro souběžný tisk, který obsahuje výsledky zpracování.

Příprava a spuštění ukázkových programů na systému IBM i

Než budete moci spustit ukázkové programy IBM MQ for IBM i , musíte je zkompilovat jako všechny ostatní aplikace IBM MQ for IBM i . K tomu můžete použít příkazy IBM i CRTRPGMOD a CRTPGM.

Při vytváření programů AMQ3xxx4 musíte v příkazu CRTPGM zadat BNDSRVPGM (QMQM/LIBMQM). To zahrnuje různé procedury IBM MQ ve vašem programu.

Ukázkové programy jsou poskytovány v knihovně QMQMSAMP jako členové QRPGLESRC. Používají kopírované soubory poskytované v knihovně QMQM, takže se při kompilaci ujistěte, že je tato knihovna v seznamu knihoven. Kompilátor RPG poskytuje informační zprávy, protože ukázky nepoužívají mnoho proměnných, které jsou deklarovány v kopírovaných souborech.

Spuštění ukázkových programů

Při spouštění ukázek můžete použít vlastní fronty, nebo můžete zkompilovat a spustit AMQSAMP4 a vytvořit některé ukázkové fronty. Zdroj pro tento program je dodáván v souboru QCLSRC v knihovně QMQMSAMP. Lze jej zkompilovat pomocí příkazu CRTCLPGM.

Chcete-li volat jeden z ukázkových programů, použijte příkaz jako:

```
CALL PGM(QMQMSAMP/AMQ3PUT4) PARM('Queue_Name','Queue_Manager_Name')
```

Kde `Queue_Name` a `Queue_Manager_Name` musí mít délku 48 znaků, což lze dosáhnout vyplňováním znaků `Queue_Name` a `Queue_Manager_Name` požadovaným počtem mezer.

Pro ukázkové programy Inquire a Set ukázkové definice vytvořené pomocí AMQSAMP4 způsobí, že se spustí verze C těchto ukázek. Chcete-li spustit verze RPG, musíte změnit definice procesu SYSTEM.SAMPLE.ECHOPROCESS a SYSTEM.SAMPLE.INQPROCESS a SYSTEM.SAMPLE.SETPROCESS. Můžete použít příkaz CHGMQMPRC (popsaný v části Změna MQ Proces (CHGMQMPRC)). nebo upravte a spusťte AMQSAMP4 s alternativní definicí.

Ukázkový program Put na systému IBM i

Ukázkový program Put AMQ3PUT4vkládá zprávy do fronty pomocí volání MQPUT.

Chcete-li spustit program, zavolejte program a jako parametr programu zadejte název cílové fronty. Program vloží do fronty sadu pevných zpráv; tyto zprávy jsou převzaty z datového bloku na konci zdrojového kódu programu. Ukázkový program vložení je AMQ3PUT4 v knihovně QMQMSAMP.

Pomocí tohoto ukázkového programu je příkaz:

```
CALL PGM(QMQMSAMP/AMQ3PUT4) PARM('Queue_Name','Queue_Manager_Name')
```

Kde `Queue_Name` a `Queue_Manager_Name` musí mít délku 48 znaků, což lze dosáhnout vyplňováním znaků `Queue_Name` a `Queue_Manager_Name` požadovaným počtem mezer.

Návrh vzorového programu Put

Program používá volání MQOPEN s volbou OOOUT k otevření cílové fronty pro vkládání zpráv. Výsledky se vypisují do souboru pro souběžný tisk. Pokud nemůže otevřít frontu, program zapíše chybovou zprávu

obsahující kód příčiny vrácený voláním MQOPEN. Chcete-li zachovat jednoduchost programu, při tomto a následných voláních MQI použije program pro mnoho voleb výchozí hodnoty.

Pro každý řádek dat obsažený ve zdrojovém kódu program načte text do vyrovnávací paměti a použije volání MQPUT k vytvoření zprávy datagramu obsahující text tohoto řádku. Program pokračuje, dokud nedosáhne konce vstupu nebo dokud neselže volání MQPUT. Pokud program dosáhne konce vstupu, uzavře frontu pomocí volání MQCLOSE.

Ukázkový program Procházet na systému IBM i

Ukázkový program Procházet, AMQ3GBR4, prochází zprávy ve frontě pomocí volání MQGET.

Program načte kopie všech zpráv ve frontě, kterou jste uvedli při volání programu; zprávy zůstanou ve frontě. Můžete použít dodanou frontu SYSTEM.SAMPLE.LOCAL; nejprve spusťte ukázkový program Put, abyste vložili některé zprávy do fronty. Můžete použít frontu SYSTEM.SAMPLE.ALIAS, což je název aliasu pro stejnou lokální frontu. Program pokračuje, dokud nedosáhne konce fronty nebo dokud neselže volání MQI.

Příklad příkazu pro volání programu RPG je:

```
CALL PGM(QMQMSAMP/AMQ3GBR4) PARM('Queue_Name','Queue_Manager_Name')
```

Kde `Queue_Name` a `Queue_Manager_Name` musí mít délku 48 znaků, což lze dosáhnout vyplňováním znaků `Queue_Name` a `Queue_Manager_Name` požadovaným počtem mezer. Proto, pokud používáte SYSTEM.SAMPLE.LOCAL jako cílová fronta, budete potřebovat 29 prázdných znaků.

Návrh ukázkového programu Procházet

Program otevře cílovou frontu pomocí volání MQOPEN s volbou OOBRW. Pokud nemůže otevřít frontu, program zapíše chybovou zprávu do svého souboru pro souběžný tisk, který obsahuje kód příčiny vrácený voláním MQOPEN.

Pro každou zprávu ve frontě program použije volání MQGET ke zkopírování zprávy z fronty a poté zobrazí data obsažená ve zprávě. Volání MQGET používá tyto volby:

GMBRWN
Po volání MQOPEN je kurzor procházení umístěn logicky před první zprávou ve frontě, takže tato volba způsobí vrácení *první* zprávy při prvním volání.

GMNWT
Program nečeká, pokud ve frontě nejsou žádné zprávy.

GMATM
Volání MQGET určuje vyrovnávací paměť pevné velikosti. Pokud je zpráva delší než tato vyrovnávací paměť, program zobrazí zkrácenou zprávu spolu s varováním, že zpráva byla zkrácena.

Program demonstruje, jak musíte vymazat pole *MDMID* a *MDCID* struktury MQMD po každém volání MQGET, protože volání nastaví tato pole na hodnoty obsažené ve zprávě, kterou načte. Vymazání těchto polí znamená, že následná volání MQGET načítají zprávy v pořadí, ve kterém jsou zprávy zadrženy ve frontě.

Program pokračuje na konec fronty; zde volání MQGET vrátí kód příčiny RC2033 (není k dispozici žádná zpráva) a program zobrazí varovnou zprávu. Pokud volání MQGET selže, program zapíše chybovou zprávu, která obsahuje kód příčiny v souboru pro souběžný tisk.

Program poté uzavře frontu pomocí volání MQCLOSE.

Ukázkový program Get na webu IBM i

Ukázkový program Get AMQ3GET4získává zprávy z fronty pomocí volání MQGET.

Když je program volán, odebere zprávy z uvedené fronty. Můžete použít dodanou frontu SYSTEM.SAMPLE.LOCAL; nejprve spusťte ukázkový program Put, abyste vložili některé zprávy do fronty. Můžete použít SYSTEM.SAMPLE.ALIAS , což je alias pro stejnou lokální frontu. Program pokračuje, dokud není fronta prázdná nebo dokud neselže volání MQI.

Příklad příkazu pro volání programu RPG je:

```
CALL PGM(QMQMSAMP/AMQ3GET4) PARM('Queue_Name','Queue_Manager_Name')
```

kde `Queue_Name` a `Queue_Manager_Name` musí mít délku 48 znaků, což lze dosáhnout vyplňováním znaků `Queue_Name` a `Queue_Manager_Name` požadovaným počtem mezer. Proto, pokud používáte SYSTEM.SAMPLE.LOCAL jako cílová fronta, budete potřebovat 29 prázdných znaků.

Návrh ukázkového programu Get

Program otevře cílovou frontu pro získávání zpráv; použije volání MQOPEN s volbou OOINPQ. Pokud nemůže otevřít frontu, program zapíše chybovou zprávu obsahující kód příčiny vrácený voláním MQOPEN v jeho souboru pro souběžný tisk.

Pro každou zprávu ve frontě program použije volání MQGET k odebrání zprávy z fronty; poté zobrazí data obsažená ve zprávě. Volání MQGET používá volbu GMWT, která uvádí interval čekání (*GMWI*) 15 sekund, takže program čeká po tuto dobu, pokud ve frontě není žádná zpráva. Pokud před vypršením tohoto intervalu nepřijde žádná zpráva, volání selže a vrátí kód příčiny RC2033 (žádná zpráva není k dispozici).

Program demonstruje, jak musíte vymazat pole *MDMID* a *MDCID* struktury MQMD po každém volání MQGET, protože volání nastaví tato pole na hodnoty obsažené ve zprávě, kterou načte. Vymazání těchto polí znamená, že následná volání MQGET načítají zprávy v pořadí, ve kterém jsou zprávy zadrženy ve frontě.

Volání MQGET určuje vyrovnávací paměť pevné velikosti. Pokud je zpráva delší než tato vyrovnávací paměť, volání selže a program se zastaví.

Program pokračuje, dokud buď volání MQGET nevrátí kód příčiny RC2033 (není k dispozici žádná zpráva), nebo dokud volání MQGET neselže. Pokud volání selže, program zobrazí chybovou zprávu, která obsahuje kód příčiny.

Program poté uzavře frontu pomocí volání MQCLOSE.

Ukázkový program Požadavek na systému IBM i

Ukázkový program Požadavek, AMQ3REQ4, demonstruje zpracování klient/server. Ukázka je klient, který vkládá zprávy požadavků do fronty, která je zpracována programem serveru. Čeká na program serveru, aby vložil zprávu odpovědi do fronty pro odpověď.

Ukázka požadavku vkládá řadu zpráv požadavků do fronty pomocí volání MQPUT. Tyto zprávy uvádějí SYSTEM.SAMPLE.REPLY jako fronta pro odpovědi. Program čeká na zprávy s odpovědí a poté je zobrazí. Odpovědi jsou odesílány pouze v případě, že cílová fronta (kterou budeme volat *frontu serveru*) je zpracováván serverovou aplikací, nebo pokud je aplikace spuštěna pro tento účel (vzorové programy Inquire a Set jsou navrženy tak, aby byly spuštěny). Vzorek čeká 5 minut na doručení první odpovědi (aby byl čas na spuštění serverové aplikace) a 15 sekund na následné odpovědi, ale může skončit bez získání odpovědí.

Chcete-li spustit program, zavolejte program a jako parametr programu zadejte název cílové fronty. Program vloží do fronty sadu pevných zpráv; tyto zprávy jsou převzaty z datového bloku na konci zdrojového kódu programu.

Návrh vzorového programu požadavku

Program otevře frontu serveru, aby mohl vkládat zprávy. Používá volání MQOPEN s volbou OOOUT. Pokud nemůže otevřít frontu, program zobrazí chybovou zprávu obsahující kód příčiny vrácený voláním MQOPEN.

Program poté otevře frontu pro odpověď s názvem SYSTEM.SAMPLE.REPLY , aby mohla získat zprávy odpovědi. K tomu program používá volání MQOPEN s volbou OOINPX. Pokud nemůže otevřít frontu, program zobrazí chybovou zprávu obsahující kód příčiny vrácený voláním MQOPEN.

Pro každý řádek vstupu pak program načte text do vyrovnávací paměti a použije volání MQPUT k vytvoření zprávy požadavku obsahující text tohoto řádku. Při tomto volání program použije volbu sestavy ROEXCD

k požadavku, aby všechny zprávy sestavy odeslané o zprávě požadavku obsahovaly prvních 100 bajtů dat zprávy. Program pokračuje, dokud nedosáhne konce vstupu nebo dokud neselže volání MQPUT.

Program poté pomocí volání MQGET odebere zprávy odpovědí z fronty a zobrazí data obsažená v odpovědích. Volání MQGET používá volbu GMWT s uvedením intervalu čekání (*GMWI*) 5 minut pro první odpověď (aby byl čas na spuštění serverové aplikace) a 15 sekund pro následné odpovědi. Program čeká na tato období, pokud ve frontě není žádná zpráva. Pokud před vypršením tohoto intervalu nepřijde žádná zpráva, volání selže a vrátí kód příčiny RC2033 (žádná zpráva není k dispozici). Volání také používá volbu GMATM, takže zprávy delší než deklarovaná velikost vyrovnávací paměti jsou oříznuty.

Program demonstruje, jak musíte vymazat pole *MDMID* a *MDCOD* struktury MQMD po každém volání MQGET, protože volání nastaví tato pole na hodnoty obsažené ve zprávě, kterou načte. Vymazání těchto polí znamená, že následná volání MQGET načítají zprávy v pořadí, ve kterém jsou zprávy zadrženy ve frontě.

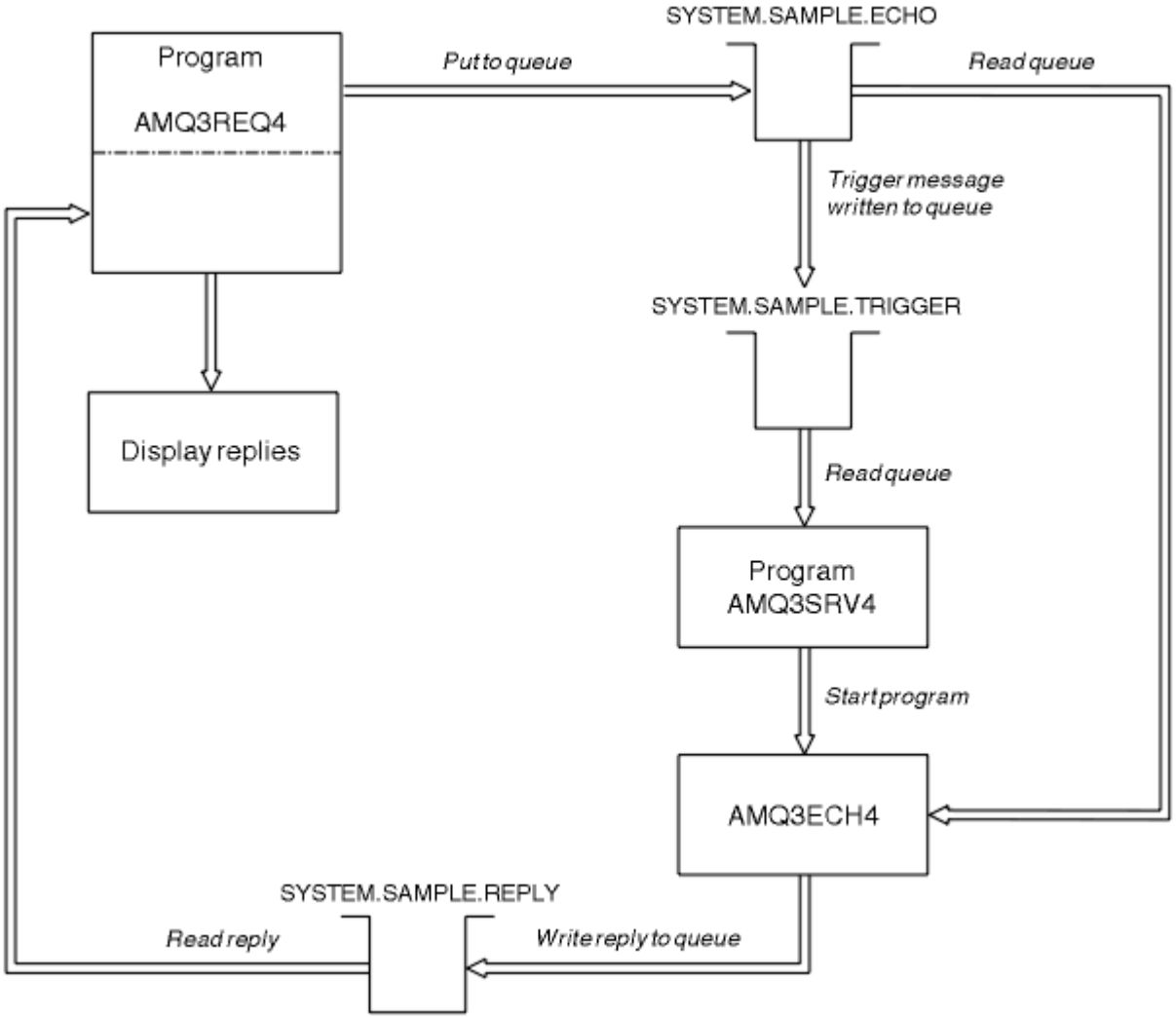
Program pokračuje, dokud buď volání MQGET nevrátí kód příčiny RC2033 (není k dispozici žádná zpráva), nebo dokud volání MQGET neselže. Pokud volání selže, program zobrazí chybovou zprávu, která obsahuje kód příčiny.

Program poté zavře frontu serveru i frontu odpovědi pomocí volání MQCLOSE. Tabulka 813 na stránce 1412 zobrazuje změny ukázkového programu Echo, které jsou nezbytné pro spuštění ukázkových programů Inquire a Set.

Poznámka: Podrobnosti pro ukázkový program Echo jsou zahrnuty jako odkaz.

Tabulka 813. Podrobnosti ukázkového programu klient/server

Název programu	**SYSTÉM/VZOROVÁ fronta**	**Program byl spuštěn**
Ozvěna	ECHO	AMQ3ECH4
Zjišťovat	inq	AMQ3INQ4
Nastavit	SET	AMQ3SET4



Obrázek 9. Diagram toku ukázkového programu klient/server (Echo)

IBM i *Použití spouštěče s ukázkou požadavku na IBM i*

Chcete-li spustit ukázku pomocí spouštěče, spusťte program spouštěcího serveru AMQ3SRV4proti požadované inicializační frontě v jedné úloze, pak spusťte AMQ3REQ4 v jiné úloze.

To znamená, že spouštěcí server je připraven, když ukázkový program požadavku odešle zprávu.

Poznámka:

1. Ukázky používají frontu SYSTEM SAMPLE TRIGGER jako inicializační frontu pro SYSTEM.SAMPLE.ECHO, SYSTEM.SAMPLE.INQnebo lokální fronty SYSTEM.SAMPLE.SET . Případně můžete definovat vlastní inicializační frontu.
2. Ukázkové definice vytvořené pomocí AMQSAMP4 způsobí spuštění verze C ukázky. Chcete-li spustit verzi RPG, musíte změnit definice procesu SYSTEM.SAMPLE.ECHOPROCESS a SYSTEM.SAMPLE.INQPROCESS a SYSTEM.SAMPLE.SETPROCESS. K tomu můžete použít příkaz CHGMQMPRC (viz Změnit proces MQ (CHGMQMPRC)), nebo můžete upravit a spustit vlastní verzi produktu AMQSAMP4.
3. Musíte zkompilovat program spouštěcího serveru ze zdroje poskytnutého v QMQMSAMP/QRPGLESRC.

V závislosti na procesu spouštěče, který chcete spustit, by měl být volán AMQ3REQ4 s parametrem uvádějícím zprávy požadavků, které se mají umístit do jedné z těchto ukázkových front serveru:

- SYSTEM.SAMPLE.ECHO (pro ukázkové programy Echo)
- SYSTEM.SAMPLE.INQ (pro ukázkové programy Inquire)

- SYSTEM.SAMPLE.SET (pro ukázkové programy Set)

Vývojový diagram pro SYSTEM.SAMPLE.ECHO je zobrazen v souboru Obrázek 9 na stránce 1413. Použití příkladu příkazu k zadání požadavku na program RPG na tento server je:

```
CALL PGM(QMQMSAMP/AMQ3REQ4) PARM('SYSTEM.SAMPLE.ECHO
+ 30 blank characters','Queue_Manager_Name')
```

protože název fronty a název správce front musí mít délku 48 znaků.

Poznámka: Tato vzorová fronta má typ spouštěče FIRST, takže pokud již existují zprávy ve frontě před spuštěním ukázky Požadavek, nejsou serverové aplikace spouštěny odesílanými zprávami.

Chcete-li se pokusit o další příklady, můžete zkusit následující variace:

- Použijte AMQ3TRG4 místo AMQ3SRV4 , abyste místo toho odeslali úlohu, ale potenciální prodlevy při odesílání úlohy by mohly zlehčovat sledování toho, co se děje.
- Použijte soubor SYSTEM.SAMPLE.INQ a SYSTEM.SAMPLE.SET . Pomocí ukázkového datového souboru jsou příkazy pro vydání požadavků programu RPG na tyto servery:

  ```
  CALL PGM(QMQMSAMP/AMQ3INQ4) PARM('SYSTEM.SAMPLE.INQ
  + 31 blank characters')
  CALL PGM(QMQMSAMP/AMQ3SET4) PARM('SYSTEM.SAMPLE.SET
  + 31 blank characters')
  ```

  protože název fronty musí mít délku 48 znaků.

  Tyto ukázkové fronty mají také typ spouštěče FIRST.

Ukázkový program Echo na systému IBM i

Ukázkové programy Echo vracejí zprávu odeslanou do fronty odpovědí. Program má název AMQ3ECH4

Aby spouštěcí proces fungoval, musíte se ujistit, že ukázkový program Echo, který chcete použít, je spuštěn zprávami přicházejícími do fronty SYSTEM.SAMPLE.ECHO. Chcete-li to provést, zadejte název ukázkového programu Echo, který chcete použít, do pole *ApplId* definice procesu SYSTEM.SAMPLE.ECHOPROCESS. (K tomu můžete použít příkaz CHGMQMPRC popsaný v části Administrace IBM i .) Ukázková fronta má typ spouštěče FIRST, takže pokud již existují zprávy ve frontě před spuštěním ukázky požadavku, není ukázka Echo spuštěna odesílanými zprávami.

Když nastavíte definici správně, nejprve spusťte AMQ3SRV4 v jedné úloze, pak spusťte AMQ3REQ4 v jiné úloze. Můžete použít AMQ3TRG4 místo AMQ3SRV4, ale potenciální prodlevy při odesílání úloh by mohly zlehčovat sledování toho, co se děje.

Ukázkové programy požadavků použijte k odeslání zpráv do fronty SYSTEM.SAMPLE.ECHO. Ukázkové programy Echo odešlou zprávu odpovědi obsahující data ve zprávě požadavku do fronty pro odpověď uvedené ve zprávě požadavku.

Návrh ukázkového programu Echo

Po spuštění se program explicitně připojí k výchozímu správci front pomocí volání MQCONN. Ačkoli to v systému IBM inení nutné, znamená to, že byste mohli používat stejný program na jiných platformách bez změny zdrojového kódu.

Program pak otevře frontu pojmenovanou ve struktuře zprávy spouštěče, která byla předána při jejím spuštění. (Pro přehlednost toto nazýváme *frontou požadavků*.) Program používá volání MQOPEN k otevření této fronty pro sdílený vstup.

Program používá volání MQGET k odebrání zpráv z této fronty. Toto volání používá volby GMATM a GMWT s intervalem čekání 5 sekund. Program testuje deskriptor každé zprávy, aby zjistil, zda se jedná o zprávu požadavku; pokud není, program zprávu vyřadí a zobrazí varovnou zprávu.

Pro každou zprávu požadavku odebranou z fronty požadavků program používá volání MQPUT k vložení zprávy odpovědi do fronty pro odpověď. Tato zpráva obsahuje obsah zprávy požadavku.

Pokud ve frontě požadavků nezbývají žádné zprávy, program tuto frontu zavře a odpojí se od správce front.

Tento program může také odpovídat na zprávy odeslané do fronty z jiných platforem než IBM i, ačkoli pro tuto situaci není uveden žádný vzorek. Aby program ECHO fungoval, můžete:

- Chcete-li odesílat zprávy textových požadavků, napište program se správným zadáním polí *Format*, *Encoding*a *CCSID* .

  Program ECHO požádá správce front o provedení konverze dat zpráv, je-li to potřeba.
- Uveďte CONVERT (*YES) na odesílacím kanálu IBM MQ for IBM i , pokud program, který jste napsali, neposkytuje podobnou konverzi pro odpověď.

Ukázkový program pro dotazování na webu IBM i

Ukázkový program dotazu AMQ3INQ4zjišťuje některé atributy fronty pomocí volání MQINQ.

Program je určen ke spuštění jako spuštěný program, takže jeho jediným vstupem je struktura MQTMC (zpráva spouštěče). Tato struktura obsahuje název cílové fronty s atributy, které mají být zjišťovány.

Aby spouštěcí proces fungoval, musíte se ujistit, že ukázkový program Inquire je spouštěn zprávami přicházejícími do fronty SYSTEM.SAMPLE.INQ. Chcete-li provést ao, zadejte název ukázkového programu Inquire do pole *ApplId* v souboru SYSTEM.SAMPLE.INQPROCESS . (K tomu můžete použít příkaz CHGMQMPRC popsaný v části Změnit proces MQ (CHGMQMPRC)). Vzorová fronta má typ spouštěče FIRST, takže pokud již existují zprávy ve frontě před spuštěním ukázky požadavku, není vzorek dotazování spuštěn odesílanými zprávami.

Když nastavíte definici správně, nejprve spusťte AMQ3SRV4 v jedné úloze, pak spusťte AMQ3REQ4 v jiné úloze. Můžete použít AMQ3TRG4 místo AMQ3SRV4, ale potenciální prodlevy při odesílání úloh mohou vést k menšímu usnadnění sledování toho, co se děje.

Ukázkový program Požadavek použijte k odeslání zpráv požadavků, z nichž každá obsahuje pouze název fronty, do fronty SYSTEM.SAMPLE.INQ. Pro každou zprávu požadavku vzorový program Inquire odešle zprávu odpovědi obsahující informace o frontě uvedené ve zprávě požadavku. Odpovědi se odešlou do fronty pro odpověď uvedené ve zprávě požadavku.

Návrh vzorového programu Inquire

Po spuštění se program explicitně připojí k výchozímu správci front pomocí volání MQCONN. Ačkoli to není v systému IBM inutné, tato funkce návrhu znamená, že byste mohli používat stejný program na jiných platformách bez změny zdrojového kódu.

Program pak otevře frontu pojmenovanou ve struktuře zprávy spouštěče, která byla předána při jejím spuštění. (Pro přehlednost toto nazýváme *frontou požadavků*.) Program používá volání MQOPEN k otevření této fronty pro sdílený vstup.

Program používá volání MQGET k odebrání zpráv z této fronty. Toto volání používá volby GMATM a GMWT s intervalem čekání 5 sekund. Program testuje deskriptor každé zprávy, aby zjistil, zda se jedná o zprávu požadavku; pokud není, program zprávu vyřadí a zobrazí varovnou zprávu.

Pro každou zprávu požadavku odebranou z fronty požadavků program přečte název fronty (kterou budeme volat *cílovou frontu*). v datech a otevře tuto frontu pomocí volání MQOPEN s volbou OOINQ. Program poté pomocí volání MQINQ zjišťuje hodnoty atributů **InhibitGet**, **CurrentQDepth**a **OpenInputCount** cílové fronty.

Je-li volání MQINQ úspěšné, program použije volání MQPUT k vložení zprávy odpovědi do fronty pro odpověď. Tato zpráva obsahuje hodnoty tří atributů.

Pokud je volání MQOPEN nebo MQINQ neúspěšné, program použije volání MQPUT k vložení zprávy *report* do fronty pro odpovědi. V poli *MDFB* deskriptoru této zprávy sestavy je uveden kód příčiny vrácený voláním MQOPEN nebo MQINQ v závislosti na tom, který z nich se nezdařil.

Po volání MQINQ program zavře cílovou frontu pomocí volání MQCLOSE.

Pokud ve frontě požadavků nezbývají žádné zprávy, program tuto frontu zavře a odpojí se od správce front.

Ukázkový program Set na systému IBM i

Ukázkový program AMQ3SET4blokuje operace vložení do fronty pomocí volání MQSET pro změnu atributu **InhibitPut** fronty.

Program je určen ke spuštění jako spouštěný program, takže jeho jediným vstupem je struktura MQTMC (zpráva spouštěče), která obsahuje název cílové fronty s atributy, které mají být zjišťovány.

Aby spouštěcí proces fungoval, musíte se ujistit, že je ukázkový program Set spuštěn zprávami přicházejícími do fronty SYSTEM.SAMPLE.SET. Chcete-li tak učinit, zadejte název ukázkového programu Set do pole *ApplId* definice procesu SYSTEM.SAMPLE.SETPROCESS. (K tomu můžete použít příkaz CHGMQMPRC popsaný v tématu Administrace IBM i .) Ukázková fronta má typ spouštěče FIRST, takže pokud již existují zprávy ve frontě před spuštěním ukázky požadavku, není ukázka Set spuštěna odesílanými zprávami.

Když nastavíte definici správně, nejprve spusťte AMQ3SRV4 v jedné úloze, pak spusťte AMQ3REQ4 v jiné úloze. Můžete použít AMQ3TRG4 místo AMQ3SRV4, ale potenciální prodlevy při odesílání úloh by mohly zlehčovat sledování toho, co se děje.

Ukázkový program Požadavek použijte k odeslání zpráv požadavků, z nichž každá obsahuje pouze název fronty, do fronty SYSTEM.SAMPLE.SET. Pro každou zprávu požadavku odešle ukázkový program Set zprávu odpovědi obsahující potvrzení, že operace vložení byly v uvedené frontě zablokovány. Odpovědi se odešlou do fronty pro odpověď uvedené ve zprávě požadavku.

Návrh vzorového programu Set

Po spuštění se program explicitně připojí k výchozímu správci front pomocí volání MQCONN. Ačkoli to není v systému IBM inutné, znamená to, že můžete použít stejný program na jiných platformách bez změny zdrojového kódu.

Program pak otevře frontu pojmenovanou ve struktuře zprávy spouštěče, která byla předána při jejím spuštění. (Pro přehlednost toto nazýváme *frontou požadavků*.) Program používá volání MQOPEN k otevření této fronty pro sdílený vstup.

Program používá volání MQGET k odebrání zpráv z této fronty. Toto volání používá volby GMATM a GMWT s intervalem čekání 5 sekund. Program testuje deskriptor každé zprávy, aby zjistil, zda se jedná o zprávu požadavku; pokud není, program zprávu vyřadí a zobrazí varovnou zprávu.

Pro každou zprávu požadavku odebranou z fronty požadavků program přečte název fronty (kterou budeme volat *cílovou frontu*). obsažené v datech a otevře tuto frontu pomocí volání MQOPEN s volbou OOSET. Program poté pomocí volání MQSET nastaví hodnotu atributu **InhibitPut** cílové fronty na hodnotu QAPUTI.

Pokud je volání MQSET úspěšné, program použije volání MQPUT k vložení zprávy odpovědi do fronty pro odpověď. Tato zpráva obsahuje řetězec `PUT inhibited`.

Pokud je volání MQOPEN nebo MQSET neúspěšné, program použije volání MQPUT k vložení zprávy *report* do fronty pro odpověď. V poli *MDFB* deskriptoru této zprávy sestavy je uveden kód příčiny vrácený voláním MQOPEN nebo MQSET v závislosti na tom, který z nich se nezdařil.

Po volání MQSET program zavře cílovou frontu pomocí volání MQCLOSE.

Pokud ve frontě požadavků nezbývají žádné zprávy, program tuto frontu zavře a odpojí se od správce front.

Ukázkové programy spouštění na systému IBM i

Produkt IBM MQ for IBM i dodává dva ukázkové programy spouštěče, které jsou napsány v ILE/RPG.

Jedná se o tyto programy:

AMQ3TRG4
: Jedná se o monitor spouštěčů pro prostředí IBM i . Odešle úlohu IBM i pro spuštění aplikace, ale to znamená, že ke každé zprávě spouštěče jsou přidruženy další náklady na zpracování.

AMQ3SRV4
Toto je spouštěcí server pro prostředí IBM i . Pro každou zprávu spouštěče tento server spustí příkaz spuštění ve své vlastní úloze, aby spustil uvedenou aplikaci. Spouštěcí server může volat transakce CICS .

Jazykové verze jazyka C těchto ukázek jsou také k dispozici jako spustitelné programy v knihovně QMQM s názvem AMQSTRG4 a AMQSERV4.

Ukázkový monitor spouštěčů AMQ3TRG4 na systému IBM i
AMQ3TRG4 je monitor spouštěčů. Vyžaduje jeden parametr: název inicializační fronty, kterou má obsluhovat. AMQSAMP4 definuje inicializační frontu vzorku, SYSTEM.SAMPLE.TRIGGER, který můžete použít při pokusu o ukázkové programy.

AMQ3TRG4 zadá úlohu IBM i pro každou platnou zprávu spouštěče, kterou získá z inicializační fronty.

Návrh monitoru spouštěčů

Monitor spouštěčů otevře inicializační frontu a získá zprávy z fronty s uvedením neomezeného intervalu čekání.

Monitor spouštěčů odešle úlohu IBM i ke spuštění aplikace určené ve zprávě spouštěče a předá strukturu MQTMC (znaková verze zprávy spouštěče). Data prostředí ve zprávě spouštěče se používají jako parametry odeslání úlohy.

Nakonec program uzavře inicializační frontu.

Ukázkový spouštěcí server AMQ3SRV4
AMQ3SRV4 je spouštěcí server. Vyžaduje jeden parametr: název inicializační fronty, kterou má obsluhovat. AMQSAMP4 definuje inicializační frontu vzorku, SYSTEM.SAMPLE.TRIGGER, který můžete použít při pokusu o ukázkové programy.

Pro každou zprávu spouštěče AMQ3SRV4 spustí příkaz start ve své vlastní úloze ke spuštění uvedené aplikace.

Pomocí vzorové fronty spouštěčů je příkaz, který se má vydat, následující:

```
CALL PGM(QMQM/AMQ3SRV4) PARM('Queue Name')
```

Kde `Queue Name` musí mít délku 48 znaků, což lze dosáhnout vyplňováním názvu fronty požadovaným počtem mezer. Proto, pokud používáte SYSTEM.SAMPLE.TRIGGER jako cílovou frontu, budete potřebovat 28 prázdných znaků.

Návrh spouštěcího serveru

Návrh spouštěcího serveru je podobný jako u monitoru spouštěčů, s výjimkou spouštěcího serveru:

- Umožňuje aplikacím CICS i IBM i .
- Nepoužívá data prostředí ze zprávy spouštěče
- Volá aplikace IBM i ve své vlastní úloze (nebo používá STRCICSUSR ke spuštění aplikací CICS) a neodesílá úlohu IBM i .
- Otevře inicializační frontu pro sdílený vstup, takže mnoho spouštěcích serverů může běžet současně.

Poznámka: Programy spuštěné pomocí AMQ3SRV4 nesmí používat volání MQDISC, protože tím dojde k zastavení spouštěcího serveru. Pokud programy spuštěné pomocí AMQ3SRV4 používají volání MQCONN, získají kód příčiny RC2002 .

Ukončení ukázkových programů spouštěče v systému IBM i
Program monitoru spouštěčů lze ukončit pomocí volby sysrequest 2 (ENDRQS) nebo blokováním získání z fronty spouštěčů.

Pokud je použita vzorová fronta spouštěčů, příkaz je:

```
CHGMQMQ QNAME('SYSTEM.SAMPLE.TRIGGER') GETENBL(*NO)
```

Poznámka: Chcete-li znovu spustit spouštění v této frontě, musíte zadat příkaz:

```
CHGMQMQ QNAME('SYSTEM.SAMPLE.TRIGGER') GETENBL(*YES)
```

Spuštění ukázek pomocí vzdálených front v systému IBM i

Vzdálené řazení do front lze demonstrovat spuštěním ukázek v připojených správcích front zpráv.

Program AMQSAMP4 poskytuje lokální definici vzdálené fronty (SYSTEM.SAMPLE.REMOTE), který používá vzdáleného správce front s názvem 0THER. Chcete-li použít tuto ukázkovou definici, změňte hodnotu 0THER na název druhého správce front zpráv, kterého chcete použít. Je třeba také nastavit kanál zpráv mezi dvěma správci front zpráv. Informace o tom, jak to provést, naleznete v tématu Programy uživatelské procedury kanálu pro kanály systému zpráv.

Ukázkový program požadavku vloží vlastní název lokálního správce front do pole *MDRM* zpráv, které odesílá. Ukázky inquire a Set odesílají zprávy odpovědí do fronty a správce front zpráv uvedených v polích *MDRQ* a *MDRM* zpráv požadavků, které zpracovávají.

Návratové kódy pro IBM i (ILE RPG)

Tyto informace popisují návratové kódy přidružené k MQI a MQAI.

Návratové kódy přidružené k:

- Příkazy PCF (Programmable Command Format) jsou uvedeny v seznamu Odkaz na formáty programovatelných příkazů.
- Volání C++ jsou uvedena v seznamu Použití C++.

Pro každé volání vrátí správce front nebo uživatelská procedura kód dokončení a kód příčiny označující úspěch nebo selhání volání.

Aplikace nesmí záviset na chybách, které jsou kontrolovány v určitém pořadí, s výjimkou případů, kdy je to výslovně uvedeno. Pokud z volání může vzniknout více než jeden kód dokončení nebo kód příčiny, konkrétní ohlášená chyba závisí na implementaci.

Kódy dokončení pro IBM i (ILE RPG)

Parametr kódu dokončení (*CMPCOD*) umožňuje volajícímu rychle zjistit, zda bylo volání úspěšně dokončeno, částečně dokončeno nebo selhalo.

CCOK
(MQCC_OK na jiných platformách)

Úspěšné dokončení.

Volání bylo zcela dokončeno; všechny výstupní parametry byly nastaveny. Parametr **REASON** má v tomto případě vždy hodnotu RCNONE.

CCWARN (varování)
(MQCC_WARN na jiných platformách)

Varování (částečné dokončení).

Volání bylo částečně dokončeno. Kromě výstupních parametrů *CMPCOD* a *REASON* mohly být nastaveny některé výstupní parametry. Parametr **REASON** poskytuje další informace o částečném dokončení.

CCFAIL
(MQCC_FAIL na jiných platformách)

Volání selhalo.

Zpracování volání nebylo dokončeno a stav správce front je obvykle nezměněn. Výjimky jsou výslovně uvedeny. Výstupní parametry *CMPCOD* a *REASON* byly nastaveny; ostatní parametry jsou nezměněny, kromě případů, kdy je to uvedeno.

Příčinou může být porucha v aplikačním programu, nebo může být výsledkem nějaké situace mimo program, například oprávnění uživatele mohlo být odvoláno. Parametr **REASON** poskytuje další informace o chybě.

Kódy příčiny pro IBM i (ILE RPG)

Parametr kódu příčiny (*REASON*) je kvalifikace na parametr kódu dokončení (*CMPCOD*).

Pokud neexistuje žádný zvláštní důvod pro hlášení, vrátí se RCNONE. Úspěšné volání vrátí CCOK a RCNONE.

Pokud je kód dokončení buď CCWARN, nebo CCFAIL, správce front vždy ohlásí oprávněnou příčinu; podrobnosti jsou uvedeny pod každým popisem volání.

Pokud uživatelské procedury nastavují kódy dokončení a příčiny, měly by dodržovat tato pravidla. Kromě toho by všechny hodnoty speciálních příčin definované uživatelskými programy měly být menší než nula, aby se zajistilo, že nebudou v konfliktu s hodnotami definovanými správcem front. Uživatelské procedury mohou nastavit příčiny, které jsou již definovány správcem front, kde jsou vhodné.

Kódy příčiny se také vyskytují v:

- Pole *DLREA* struktury MQDLH
- Pole *MDFB* struktury MQMD

Úplný seznam kódů příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Chcete-li najít kód příčiny IBM i v tomto seznamu, odeberte "RC" zepředu, například RC2002 se stane 2002. Také kódy dokončení jsou zobrazeny tak, jak jsou na jiných platformách:

Tabulka 814. Názvy kódů příčiny na systému IBM i a na jiných platformách

IBM i	**Ostatní platformy**
CCOK	MQCC_OK
CCWARN (varování)	MQCC_WARN
CCFAIL	MQCC_FAIL

Pravidla pro ověření voleb MQI pro IBM i (ILE RPG)

Toto téma poskytuje informace o situacích, které vytvářejí kód příčiny RC2046 z volání MQOPEN, MQPUT, MQPUT1, MQGET nebo MQCLOSE.

Volání MQOPEN na IBM i

Pro volby volání MQOPEN:

- *Musí být uveden alespoň jeden* z následujících:
  - OOBRW
  - OOINPQ
  - OOINPX
  - OOINPS
  - OOINQ
  - OOOUT

  - OOSET
- Je povolena pouze *jedna* z následujících možností:
  - OOINPQ
  - OOINPX
  - OOINPS
- Je povolena pouze *jedna* z následujících možností:
  - OOBNDO
  - OOBNDN
  - OOBNDQ

  Poznámka: Dříve uvedené volby se vzájemně vylučují. Avšak vzhledem k tomu, že hodnota OOBNDQ je nula, její uvedení s některou z dalších dvou voleb vazby nevede ke kódu příčiny RC2046. OOBNDQ je poskytován jako pomůcka pro programovou dokumentaci.
- Je-li zadána volba OOSAVA, musí být zadána také jedna z voleb OOINP*.
- Je-li zadána jedna z voleb OOSET* nebo OOPAS*, musí být uveden také OOOUT.

Volání MQPUT na systému IBM i

Pro volby vložení zprávy:

- Kombinace PMSYP a PMNSYP není povolena.
- Je povolena pouze *jedna* z následujících možností:
  - PMDEFC
  - PMNOC
  - PMPASA
  - PMPASI
  - PMSETA
  - PMSETI
- PMALTU není povolen (je platný pouze pro volání MQPUT1).

MQPUT1 volání IBM i

Pro volby put-message jsou pravidla stejná jako pro volání MQPUT, s výjimkou následujících voleb:

- PMALTU je povoleno.
- PMLOGO není povoleno.

Volání MQGET na IBM i

Pro volby get-message:

- Je povolena pouze *jedna* z následujících voleb:
  - GMNSYP
  - GMSYP
  - GMPSYP
- Je povolena pouze *jedna* z následujících voleb:
  - GMBRWF
  - GMBRWC
  - GMBRWN

  - GMMUC
- GMSYP není povolen s žádnou z následujících voleb:
  - GMBRWF
  - GMBRWC
  - GMBRWN
  - GMLK
  - GMUNLK
- GMPSYP není povolen s žádnou z následujících voleb:
  - GMBRWF
  - GMBRWC
  - GMBRWN
  - GMCMPM
  - GMUNLK
- Je-li zadána volba GMLK, musí být zadána také jedna z následujících voleb:
  - GMBRWF
  - GMBRWC
  - GMBRWN
- Je-li zadána volba GMUNLK, jsou povoleny pouze následující volby:
  - GMNSYP
  - GMNWT

Volání MQCLOSE na IBM i

- Pro volby volání MQCLOSE. Kombinace CODEL a COPURG není povolena.
- Je povolena pouze jedna z následujících možností:
  - COKPSB
  - CORMSB

Volání MQSUB na IBM i

Pro volby volání MQSUB:

- Musí být uveden alespoň jeden z následujících:
- Musí být uveden alespoň jeden z následujících:
  - ALT
  - SORES
  - SOCRT
- Je povolena pouze jedna z následujících možností:
  - SODUR
  - SONDUR

  Poznámka: Dříve uvedené volby se vzájemně vylučují. Avšak vzhledem k tomu, že hodnota SOnDUR je nula, uvedení s SODUR nevede ke kódu příčiny RC2046. SONDUR je k dispozici jako pomůcka pro programovou dokumentaci.
- Kombinace SOGRP a SOMAN není povolena.
- Parametr SOGRP vyžaduje zadání parametru SOSCID.
- Je povolena pouze jedna z následujících možností: SOAUID SOFUID

- Kombinace SONEWP a SOPUBR není povolena.
- SONEWP je povolen pouze v kombinaci s SOCRT.
- Je povolena pouze jedna z následujících možností:
  - SOWCHR
  - SOWTOP

Kódování počítačů na IBM i

Pomocí těchto informací získáte informace o struktuře pole *MDENC* v deskriptoru zprávy.

Další informace o deskriptoru zprávy viz "MQMD (deskriptor zprávy) na IBM i" na stránce 1105.

Pole *MDENC* je 32bitové celé číslo, které je rozděleno do čtyř samostatných dílčích polí; tato dílčí pole identifikují:

- Kódování použité pro binární celá čísla
- Kódování použité pro pakovaná desetinná celá čísla
- Kódování použité pro čísla s pohyblivou řádovou čárkou
- Vyhrazené bity

Každé podpole je identifikováno bitovou maskou, která má na pozicích odpovídajících podpoli 1 bity a jinde 0 bity. Bity jsou očíslovány tak, že bit 0 je nejvýznamnější bit, a bit 31 nejméně významný bit. Jsou definovány následující masky:

ENIMSK
Maska pro kódování binárních čísel.

Toto podpole zabírá pozice bitů 28 až 31 v poli *MDENC* .

ENDMSK
Maska pro kódování packed-decimal-integer.

Toto podpole zabírá bitové pozice 24 až 27 v poli *MDENC* .

ENFMSK
Maska pro kódování s pohyblivou řádovou čárkou.

Toto podpole zabírá pozice bitů 20 až 23 v poli *MDENC* .

ENRMSK
Maska pro vyhrazené bity.

Toto podpole zabírá pozice bitů 0 až 19 v poli *MDENC* .

IBM i Kódování binárních čísel na IBM i

Platné hodnoty pro kódování binárních čísel.

Následující hodnoty jsou platné pro kódování binárního celého čísla:

ENIUND
Nedefinované kódování celých čísel.

Binární celá čísla jsou reprezentována pomocí nedefinovaného kódování.

ENINOR
Normální kódování celých čísel.

Binární celá čísla jsou reprezentována konvenčním způsobem:

- Nejméně významný bajt v čísle má nejvyšší adresu libovolného z bajtů v čísle; nejvýznamnější bajt má nejnižší adresu.
- Nejméně významný bit v každém bajtu je vedle bajtu s další vyšší adresou; nejvýznamnější bit v každém bajtu je vedle bajtu s další nižší adresou.

ENIREV
Obrácené kódování celých čísel.

Binární celá čísla jsou reprezentována stejným způsobem jako ENINOR, ale s bajty uspořádanými v obráceném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako ENINOR.

IBM i Kódování packed-decimal-integer na IBM i

Platné hodnoty pro kódování packed-decimal-integer

Následující hodnoty jsou platné pro kódování packed-decimal-integer:

ENDUND
Nedefinované kódování packed-decimal.

Celá čísla ve formátu packed-decimal jsou reprezentována nedefinovaným kódováním.

ENDNOR
Normální pakované desetinné kódování.

Pakovaná desetinná celá čísla jsou reprezentována konvenčním způsobem:

- Každá desetinná číslice v tisknutelném tvaru čísla je reprezentována v desítkovém balení jednou hexadecimální číslicí v rozsahu X' 0 'až X' 9'. Každá hexadecimální číslice zabírá 4 bity, a tak každý bajt v pakovaném desítkovém čísle představuje dvě desetinná čísla v tisknutelné podobě čísla.
- Nejméně významný bajt v pakovaných desetinných číslech je bajt, který obsahuje nejméně významnou desetinnou číslici. V rámci tohoto bajtu nejvýznamnější 4 bity obsahují nejméně významnou desetinnou číslici a nejméně významné 4 bity obsahují znaménko. Znaménko je buď X'C '(kladné), X 'D' (záporné), nebo X'F ' (nepodepsané).
- Nejméně významný bajt v čísle má nejvyšší adresu libovolného z bajtů v čísle; nejvýznamnější bajt má nejnižší adresu.
- Nejméně významný bit v každém bajtu je vedle bajtu s další vyšší adresou; nejvýznamnější bit v každém bajtu je vedle bajtu s další nižší adresou.

ENDREV
Obrácené kódování pakovaných desetinných míst.

Pakovaná desetinná celá čísla jsou reprezentována stejným způsobem jako ENDNOR, ale s bajty uspořádanými v obráceném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako ENDNOR.

IBM i Kódování s pohyblivou řádovou čárkou na systému IBM i

Platné hodnoty pro kódování s pohyblivou řádovou čárkou

Následující hodnoty jsou platné pro kódování s pohyblivou řádovou čárkou:

ENFUND
Nedefinované kódování s pohyblivou řádovou čárkou.

Čísla s pohyblivou řádovou čárkou jsou reprezentována pomocí nedefinovaného kódování.

ENFNOR
Normální IEEE (Ústav elektrických a elektronických inženýrů) float kódování.

Čísla s pohyblivou řádovou čárkou jsou reprezentována pomocí standardního formátu IEEE s pohyblivou řádovou čárkou, přičemž bajty jsou uspořádány takto:

- Nejméně významný bajt v mantise má nejvyšší adresu libovolného z bajtů v čísle; bajt obsahující exponent má nejnižší adresu
- Nejméně významný bit v každém bajtu je vedle bajtu s další vyšší adresou; nejvýznamnější bit v každém bajtu je vedle bajtu s další nižší adresou.

Podrobnosti o kódování typu float IEEE lze nalézt v normě IEEE 754.

ENFREV
Obrácené kódování typu float IEEE.

Čísla s pohyblivou řádovou čárkou jsou reprezentována stejným způsobem jako ENFNOR, ale s bajty uspořádanými v obráceném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako ENFNOR.

ENF390
System/390 kódování s pohyblivou řádovou čárkou.

Čísla s pohyblivou řádovou čárkou jsou reprezentována pomocí standardního formátu System/390 s pohyblivou řádovou čárkou. Tento formát používá také systém System/370.

IBM i Konstrukce kódování na IBM i

Chcete-li vytvořit hodnotu pro pole *MDENC* v deskriptoru MQMD, je třeba přidat příslušné konstanty, které popisují požadovaná kódování.

Nezapomeňte kombinovat pouze jedno z kódování ENI* s jedním z kódování END* a jedním z kódování ENF*.

IBM i Analýza kódování na IBM i

Pole *MDENC* obsahuje dílčí pole. Aplikace, které potřebují prozkoumat celé číslo, pakované desetinné číslo nebo kódování s pohyblivou řádovou čárkou, by proto měly používat techniku popsanou v tomto tématu.

Použití aritmetické

Následující kroky by měly být provedeny pomocí celočíselné aritmetiky:

1. Vyberte jednu z následujících hodnot podle typu požadovaného kódování:
   - 1 pro kódování binárních celých čísel
   - 16 pro kódování pakovaného desetinného celého čísla
   - 256 pro kódování s pohyblivou řádovou čárkou

   Zavolejte hodnotu A.
2. Vydělte hodnotu pole *MDENC* hodnotou A ; Zavolejte výsledek B.
3. Vydělte B 16; vyvolejte výsledek C.
4. Vynásobte C číslem 16 a odečtěte od B ; Zavolejte výsledek D.
5. Vynásobte D hodnotou A ; Zavolejte výsledek E.
6. E je požadované kódování a lze jej testovat na shodu s každou z hodnot, které jsou platné pro daný typ kódování.

IBM i Souhrn kódování architektury počítače na IBM i

Tabulka shrnující kódování pro architektury počítačů.

Kódování pro architektury počítačů jsou uvedeny v části Tabulka 815 na stránce 1424.

Tabulka 815. Souhrn kódování pro architektury počítačů

Architektura počítače	Kódování binárních celých čísel	Kódování celých čísel v pakované desítkové soustavě	Kódování s pohyblivou řádovou čárkou
IBM i	Normální	Normální	IEEE normální
Intel x86	Převrácené	Převrácené	IEEE obráceno

Tabulka 815. Souhrn kódování pro architektury počítačů (pokračování)			
Architektura počítače	**Kódování binárních celých čísel**	**Kódování celých čísel v pakované desítkové soustavě**	**Kódování s pohyblivou řádovou čárkou**
PowerPC	Normální	Normální	IEEE normální
System/390	Normální	Normální	System/390

IBM i Volby sestavy a příznaky zpráv na IBM i

Toto téma se týká polí *MDREP* a *MDMFL* , která jsou součástí deskriptoru zprávy MQMD určeného ve voláních MQGET, MQPUT a MQPUT1 .

Další informace o deskriptoru zprávy viz “MQMD (deskriptor zprávy) na IBM i” na stránce 1105. Tyto informace popisují:

- Struktura pole sestavy a způsob, jakým ji správce front zpracovává
- Jak by měla aplikace analyzovat pole sestavy
- Struktura pole příznaků zprávy

Struktura pole sestavy

Pole *MDREP* je 32bitové celé číslo, které je rozděleno do tří samostatných dílčích polí.

Tato podpole identifikují:

- Volby sestavy, které jsou odmítnuty, pokud je lokální správce front nerozpozná
- Volby sestavy, které jsou vždy přijaty, i když je lokální správce front nerozpozná
- Volby sestavy, které jsou přijaty pouze v případě, že jsou splněny určité další podmínky

Každé podpole je identifikováno bitovou maskou, která má na pozicích odpovídajících podpoli 1 bity a jinde 0 bity. Všimněte si, že bity v podpoli nemusí nutně sousedící. Bity jsou očíslovány tak, že bit 0 je nejvýznamnější bit, a bit 31 nejméně významný bit. Pro identifikaci dílčích polí jsou definovány následující masky:

RORUM
Maska pro nepodporované volby sestavy, které jsou odmítnuty.

Tato maska identifikuje bitové pozice v poli *MDREP* , kde volby sestavy, které nejsou podporovány lokálním správcem front, způsobí selhání volání MQPUT nebo MQPUT1 s kódem dokončení CCFAIL a kódem příčiny RC2061.

Toto podpole zabírá bitové pozice 3 a 11 až 13.

ROAUM
Maska pro nepodporované volby sestavy, které jsou přijaty.

Tato maska identifikuje bitové pozice v poli *MDREP* , kde budou volby sestavy, které nejsou podporovány lokálním správcem front, přesto přijaty ve volání MQPUT nebo MQPUT1 . V tomto případě se vrátí kód dokončení CCWARN s kódem příčiny RC2104 .

Toto podpole zabírá bitové pozice 0 až 2, 4 až 10 a 24 až 31.

V tomto podpoli jsou zahrnuty následující volby sestavy:

- ROCMTC
- RODLQ
- RODISC
- ROEXC
- ROEXCD

- ROEXCF
- ROEXP
- ROEXPD
- ROEXPF
- RONAN
- RONMI
- RONONE
- ROPAN
- ROPCI
- ROPMI

ROAUXM
Maska pro nepodporované volby sestavy, které jsou přijaty pouze za určitých okolností.

Tato maska identifikuje bitové pozice v poli *MDREP* , kde budou volby sestavy, které nejsou podporovány lokálním správcem front, přesto přijaty ve volání MQPUT nebo MQPUT1 *za předpokladu* , že jsou splněny obě následující podmínky:

- Zpráva je určena pro vzdáleného správce front.
- Aplikace nevkládá zprávu přímo do lokální přenosové fronty (tj. fronta určená poli *ODMN* a *ODON* v deskriptoru objektu určeném ve volání MQOPEN nebo MQPUT1 není lokální přenosovou frontou).

Kód dokončení CCWARN s kódem příčiny RC2104 se vrátí, pokud jsou tyto podmínky splněny, a CCFAIL s kódem příčiny RC2061 , pokud ne.

Toto podpole zabírá bitové pozice 14 až 23.

V tomto podpoli jsou zahrnuty následující volby sestavy:

- ROCOA-Přepočet
- ROCOAD
- ROCOAF
- ROCOD
- ROCODD
- ROCODF

Pokud jsou v poli *MDREP* zadány nějaké volby, které správce front nerozpozná, správce front postupně zkontroluje jednotlivá dílčí pole pomocí bitové operace AND a zkombinuje pole *MDREP* s maskou pro dané dílčí pole. Není-li výsledek této operace nulový, vrátí se dříve popsaný kód dokončení a kódy příčiny.

Je-li vrácena hodnota CCWARN, není definováno, který kód příčiny je vrácen, pokud existují jiné varovné podmínky.

Schopnost určit a mít přijaté volby sestavy, které nejsou rozpoznány lokálním správcem front, je užitečná v případě, že je nutné odeslat zprávu s volbou sestavy, která bude rozpoznána a zpracována *vzdáleným* správcem front.

IBM i Analýza pole sestavy na IBM i

Pole MDREP obsahuje dílčí pole. Z tohoto důvodu některé aplikace musí zkontrolovat, zda odesilatel zprávy požadoval určitou sestavu. Tyto aplikace by měly používat techniku popsanou v tomto tématu.

Použití aritmetické

Následující kroky by měly být provedeny pomocí celočíselné aritmetiky:

1. Vyberte jednu z následujících hodnot podle typu sestavy, která se má zkontrolovat:
   - Sestava ROCOA pro COA

- ROCOD pro sestavu COD
- ROEXC pro sestavu výjimek
- ROEXP pro sestavu vypršení platnosti

Zavolejte hodnotu A.

2. Pole *MDREP* vydělte hodnotou A ; Zavolejte výsledek B.
3. Vydělte B hodnotou 8 ; Zavolejte výsledek C.
4. Vynásobte C hodnotou 8 a odečtěte od B ; Zavolejte výsledek D.
5. Vynásobte D hodnotou A ; Zavolejte výsledek E.
6. Otestujte E na shodu s každou z hodnot, které jsou možné pro daný typ sestavy.

   Pokud je například A ROEXC, otestujte E pro shodu s každým z následujících, abyste určili, co bylo uvedeno odesilatelem zprávy:

   - RONONE
   - ROEXC
   - ROEXCD
   - ROEXCF

   Testy lze provádět v jakémkoli pořadí, které je pro logiku aplikace nejvhodnější.

Následující pseudokód ilustruje tuto techniku pro zprávy hlášení výjimek:

```
A = ROEXC
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

Podobnou metodu lze použít k testování voleb ROPMI nebo ROPCI; vyberte hodnotu A podle toho, která z těchto dvou konstant je vhodná, a pak pokračujte podle popisu výše, ale nahraďte hodnotu 8 v předchozích krocích hodnotou 2.

IBM i Struktura pole příznaků zprávy na IBM i

Pole *MDMFL* je 32bitové celé číslo, které je rozděleno do tří samostatných dílčích polí.

Tato podpole identifikují:

- Příznaky zpráv, které jsou odmítnuty, pokud je lokální správce front nerozpozná
- Příznaky zpráv, které jsou vždy přijaty, a to i v případě, že je lokální správce front nerozpozná.
- Příznaky zprávy, které jsou přijaty pouze v případě, že jsou splněny určité další podmínky

Poznámka: Všechna dílčí pole v souboru *MDMFL* jsou vyhrazena pro použití správcem front.

Každé podpole je identifikováno bitovou maskou, která má na pozicích odpovídajících podpoli 1 bity a jinde 0 bity. Bity jsou očíslovány tak, že bit 0 je nejvýznamnější bit, a bit 31 nejméně významný bit. Pro identifikaci dílčích polí jsou definovány následující masky:

MFRUM
Maska pro nepodporované příznaky zpráv, které jsou odmítnuty.

Tato maska identifikuje bitové pozice v poli *MDMFL* , kde příznaky zpráv, které nejsou podporovány lokálním správcem front, způsobí selhání volání MQPUT nebo MQPUT1 s kódem dokončení CCFAIL a kódem příčiny RC2249.

Toto podpole zabírá bitové pozice 20 až 31.

V tomto dílčím poli jsou zahrnuty následující příznaky zprávy:

- MFLMIG

- MFLSEG-rozšíření
- MFMIG
- MFSEG
- MFSEGA
- MFSEGI

MFAUM
Maska pro nepodporované příznaky zprávy, které jsou přijaty.

Tato maska identifikuje bitové pozice v poli *MDMFL* , na kterých budou přesto při volání MQPUT nebo MQPUT1 přijímány příznaky zpráv, které nejsou podporovány lokálním správcem front. Kód dokončení je CCOK.

Toto podpole zabírá bitové pozice 0 až 11.

MFAUXM
Maska pro nepodporované příznaky zpráv, které jsou přijaty pouze za určitých okolností.

Tato maska identifikuje bitové pozice v poli *MDMFL* , kde budou příznaky zpráv, které nejsou podporovány lokálním správcem front, přesto přijaty ve volání MQPUT nebo MQPUT1 *za předpokladu* , že jsou splněny obě následující podmínky:

- Zpráva je určena pro vzdáleného správce front.
- Aplikace nevkládá zprávu přímo do lokální přenosové fronty (tj. fronta určená poli *ODMN* a *ODON* v deskriptoru objektu určeném ve volání MQOPEN nebo MQPUT1 není lokální přenosovou frontou).

Kód dokončení CCOK se vrátí, pokud jsou tyto podmínky splněny, a CCFAIL s kódem příčiny RC2249 , pokud ne.

Toto podpole zabírá bitové pozice 12 až 19.

Pokud jsou v poli *MDMFL* uvedeny příznaky, které správce front nerozpozná, správce front postupně zkontroluje jednotlivá dílčí pole pomocí bitové operace AND a zkombinuje pole *MDMFL* s maskou pro dané dílčí pole. Není-li výsledek této operace nulový, vrátí se dříve popsaný kód dokončení a kódy příčiny.

IBM i Převod dat na IBM i

Toto téma popisuje rozhraní uživatelské procedury pro převod dat a zpracování prováděné správcem front v případě, že je vyžadován převod dat.

Uživatelská procedura převodu dat je vyvolána jako součást zpracování volání MQGET. Používá se k převodu dat zprávy aplikace na reprezentaci požadovanou přijímající aplikací. Převod dat zprávy aplikace je volitelný a vyžaduje zadání volby GMCONV ve volání MQGET.

Jsou popsány následující aspekty převodu dat:

- Zpracování provedené správcem front v reakci na volbu GMCONV; viz “Zpracování převodu na IBM i” na stránce 1429.
- Konvence zpracování používané správcem front při zpracování vestavěného formátu. Tyto konvence se doporučují i pro uživatelské procedury napsané uživatelem. Viz “Konvence zpracování v systému IBM i” na stránce 1430.
- Speciální pokyny pro převod zpráv sestavy; viz “Převod zpráv sestavy na IBM i” na stránce 1433.
- Parametry předané uživatelské proceduře pro převod dat; viz “MQCONVX (uživatelská procedura převodu dat) na systému IBM i” na stránce 1444.
- Volání, které lze použít z uživatelské procedury pro převod znakových dat mezi různými reprezentacemi; viz “MQXCNVC (Převod znaků) na systému IBM i” na stránce 1439.
- Parametr datové struktury, který je specifický pro uživatelskou proceduru; viz “MQDXP (parametr uživatelské procedury pro převod dat) na systému IBM i” na stránce 1434.

IBM i Zpracování převodu na IBM i

Tyto informace popisují zpracování prováděné správcem front v reakci na volbu GMCONV.

Je-li ve volání MQGET zadána volba GMCONV a existuje-li zpráva, která má být vrácena aplikaci, provede správce front následující akce:

1. Je-li splněna jedna nebo více následujících podmínek, není převod nutný:
   - Data zprávy jsou již ve znakové sadě a kódování vyžadované aplikací vydávající volání MQGET. Aplikace musí před zadáním volání nastavit pole *MDCSI* a *MDENC* v parametru **MSGDSC** volání MQGET na požadované hodnoty.
   - Délka dat zprávy je nula.
   - Délka parametru **BUFFER** volání MQGET je nula.

   V těchto případech je zpráva vrácena bez převodu do aplikace vydávající volání MQGET; hodnoty *MDCSI* a *MDENC* v parametru **MSGDSC** jsou nastaveny na hodnoty v řídicích informacích ve zprávě a volání je dokončeno jednou z následujících kombinací kódu dokončení a kódu příčiny:

   Kód dokončení
   Kód příčiny

   CCOK
   RCNONE

   CCWARN (varování)
   RC2079

   CCWARN (varování)
   RC2080

Následující kroky jsou provedeny pouze v případě, že se znaková sada nebo kódování dat zprávy liší od odpovídající hodnoty v parametru **MSGDSC** a existují data, která mají být převedena:

1. Pokud má pole *MDFMT* v řídicích informacích ve zprávě hodnotu FMNONE, zpráva se nepřevede s kódem dokončení CCWARN a kódem příčiny RC2110.

   Ve všech ostatních případech zpracování převodu pokračuje.
2. Zpráva je odebrána z fronty a umístěna do dočasné vyrovnávací paměti, která má stejnou velikost jako parametr **BUFFER** . Pro operace procházení je zpráva zkopírována do dočasné vyrovnávací paměti, místo aby byla odebrána z fronty.
3. Pokud má být zpráva zkrácena tak, aby se vešla do vyrovnávací paměti, provede se následující:
   - Pokud nebyla uvedena volba GMATM, zpráva se vrátí nepřevedená, s kódem dokončení CCWARN a kódem příčiny RC2080.
   - Pokud byla volba GMATM *uvedena* , kód dokončení je nastaven na CCWARN, kód příčiny je nastaven na RC2079a zpracování převodu pokračuje.
4. Pokud lze zprávu umístit do vyrovnávací paměti bez oříznutí nebo byla zadána volba GMATM, provede se následující:
   - Pokud je formát vestavěným formátem, je vyrovnávací paměť předána službě pro převod dat správce front.
   - Pokud formát není vestavěným formátem, je vyrovnávací paměť předána uživatelské proceduře, která má stejný název jako formát. Pokud uživatelskou proceduru nelze nalézt, zpráva se vrátí nepřevedená, s kódem dokončení CCWARN a kódem příčiny RC2110.

   Pokud nedojde k žádné chybě, bude výstupem ze služby pro převod dat nebo z uživatelské procedury převedená zpráva plus kód dokončení a kód příčiny, který má být vrácen aplikaci vydávající volání MQGET.
5. Je-li převod úspěšný, vrátí správce front převedené zprávy aplikaci. V tomto případě bude kód dokončení a kód příčiny vrácený voláním MQGET obvykle jednou z následujících kombinací:

Kód dokončení
Kód příčiny

CCOK
RCNONE

CCWARN (varování)
RC2079

Pokud je však konverze provedena uživatelskou procedurou, mohou být vráceny i jiné kódy příčiny, i když je konverze úspěšná.

Pokud se převod nezdaří (z jakéhokoli důvodu), vrátí správce front nepřevedenou zprávu aplikaci s poli *MDCSI* a *MDENC* v parametru **MSGDSC** nastavenými na hodnoty v řídicích informacích ve zprávě a s kódem dokončení CCWARN.

IBM i Konvence zpracování v systému IBM i

Při převodu vestavěného formátu se správce front řídí konvencemi zpracování popsanými v tomto tématu.

Zvažte použití těchto konvencí na uživatelské procedury napsané uživatelem, ačkoli to správce front nevynucuje. Vestavěné formáty převedené správcem front jsou následující:

- FMADMN
- FMMDE
- FMCICS:
- Prostředek FMPCF
- FMCMD1
- FMRMH
- FMCMD2
- FMRFH
- FMDLH
- FMRFH2
- FMDH
- FMSTR
- FMEVNT
- FMTM
- FMIMS
- FMXQH
- FMIMVS

1. Pokud se zpráva během převodu rozbalí a překročí velikost parametru **BUFFER** , provede se následující:
   - Pokud nebyla uvedena volba GMATM, zpráva se vrátí nepřevedená, s kódem dokončení CCWARN a kódem příčiny RC2120.
   - Pokud byla volba GMATM *uvedena* , zpráva je oříznuta, kód dokončení je nastaven na CCWARN, kód příčiny je nastaven na RC2079a zpracování převodu pokračuje.
2. Pokud dojde k oříznutí (buď před, nebo během konverze), je možné, aby počet platných bajtů vrácených v parametru **BUFFER** byl *menší* než délka vyrovnávací paměti.

   K tomu může dojít například v případě, že konec vyrovnávací paměti je zakončen čtyřbajtovým celým číslem nebo znakem DBCS. Neúplný prvek informací není převeden, a proto tyto bajty ve vrácené zprávě neobsahují platné informace. K tomu může dojít i v případě, že se během převodu zmenší zpráva, která byla oříznuta před převodem.

Pokud je počet vrácených platných bajtů menší než délka vyrovnávací paměti, jsou nepoužívané bajty na konci vyrovnávací paměti nastaveny na hodnotu null.

3. Pokud pole nebo řetězec straddles konec vyrovnávací paměti, co nejvíce dat, jak je to možné, je převeden; pouze konkrétní prvek pole nebo DBCS znak, který je neúplný není převeden-předchozí prvky pole nebo znaky jsou převedeny.
4. Pokud dojde k oříznutí (buď před, nebo během konverze), délka vrácená pro parametr **DATLEN** je délka *nepřevedené* zprávy před oříznutím.
5. Když jsou řetězce převáděny mezi jednobajtovými znakovými sadami (SBCS), dvoubajtovými znakovými sadami (DBCS) nebo vícebajtovými znakovými sadami (MBCS), mohou se řetězce rozšiřovat nebo uzavírat.
   - Ve formátech PCF FMADMN, FMEVNT a FMPCF se řetězce ve strukturách MQCFST a MQCFSL rozšiřují nebo uzavíhají podle potřeby tak, aby vyhovovaly řetězci po převodu.

     Pro strukturu seznamu řetězců MQCFSL se mohou řetězce v seznamu rozšiřovat nebo uzavírat v různých množstvích. Pokud k tomu dojde, správce front vyplní kratší řetězce mezerami, aby byly stejné délky jako nejdelší řetězec po převodu.
   - Ve formátu FMRMH se řetězce adresované poli RMSEO, RMSNO, RMDEOa RMDNO rozšiřují nebo uzavírá podle potřeby tak, aby vyhovovaly řetězcům po převodu.
   - Ve formátu FMRFH se pole RFNVS rozbalí nebo podle potřeby uzavírá, aby se po převodu vešel do dvojic název-hodnota.
   - Ve strukturách s pevnými velikostmi polí umožňuje správce front řetězcům rozšířit nebo uzavřít smlouvu v rámci svých pevných polí, pokud nebudou ztraceny žádné významné informace. V tomto ohledu jsou koncové mezery a znaky následující za prvním znakem null v poli považovány za nevýznamné.
     - Pokud se řetězec rozbalí, ale pouze nevýznamné znaky musí být vyřazeny, aby se do něj vešel převedený řetězec v poli, konverze se zdaří a volání se dokončí s kódem příčiny CCOK a kódem příčiny RCNONE (nepředpokládá se žádné další chyby).
     - Pokud se řetězec rozbalí, ale převedený řetězec vyžaduje vyřazení významných znaků, aby se vešel do pole, zpráva se vrátí nepřevedená a volání se dokončí s CCWARN a kódem příčiny RC2190.

       Poznámka: Kód příčiny RC2190 v tomto případě způsobí, že volba GMATM byla uvedena.
     - Pokud se řetězec uzavírá, správce front vyplní řetězec mezerami na délku pole.
6. U zpráv sestávajících z jedné nebo více struktur záhlaví IBM MQ následovaných uživatelskými daty je možné převést jednu nebo více struktur záhlaví, zatímco zbytek zprávy nikoli. Avšak se dvěma výjimkami pole MDCSI a MDENC v každé struktuře záhlaví vždy správně označují znakovou sadu a kódování dat, která následují za strukturou záhlaví.

   Dvě výjimky jsou struktury MQCIH a MQIIH, kde hodnoty v polích MDCSI a MDENC v těchto strukturách nejsou významné. Pro tyto struktury jsou data následující za strukturou ve stejné znakové sadě a kódování jako samotná struktura MQCIH nebo MQIIH.
7. Pokud pole MDCSI nebo MDENC v řídicích informacích načítané zprávy nebo v parametru **MSGDSC** uvádějí hodnoty, které nejsou definovány nebo nejsou podporovány, může správce front ignorovat chybu, pokud nedefinovaná nebo nepodporovaná hodnota nemusí být použita při převodu zprávy.

   Pokud například pole MDENC ve zprávě uvádí nepodporované kódování s pohyblivou řádovou čárkou, ale zpráva obsahuje pouze celočíselná data, nebo obsahuje data s pohyblivou řádovou čárkou, která nevyžadují převod (protože zdrojové a cílové kódování s pohyblivou řádovou čárkou jsou identické), chyba může nebo nemusí být diagnostikována.

   Pokud je chyba diagnostikována, zpráva se vrátí nepřevedená, s kódem dokončení CCWARN a jedním z kódů příčiny RC2111, RC2112, RC2113, RC2114 nebo RC2115, RC2116, RC2117, RC2118 (podle potřeby); pole MDCSI a MDENC v parametru **MSGDSC** jsou nastavena na hodnoty v řídicích informacích ve zprávě.

Není-li chyba diagnostikována a převod byl úspěšně dokončen, hodnoty vrácené v polích MDCSI a MDENC v parametru **MSGDSC** jsou hodnoty určené aplikací vydávající volání MQGET.

8. Ve všech případech, pokud je zpráva vrácena nepřevedené aplikaci, je kód dokončení nastaven na hodnotu CCWARN a pole MDCSI a MDENC v parametru **MSGDSC** jsou nastavena na hodnoty odpovídající nepřevedeným datům. To se provádí také pro FMNONE.

   Parametr **REASON** je nastaven na kód, který označuje, proč nebylo možné převod provést, pokud zpráva nebyla také oseknuta; kódy příčiny související s oseknutím mají přednost před kódy příčiny souvisejícími s převodem. (Chcete-li určit, zda byla oříznutá zpráva převedena, zkontrolujte hodnoty vrácené v polích MDCSI a MDENC v parametru **MSGDSC** .)

   Když je diagnostikována chyba, vrátí se buď specifický kód příčiny, nebo obecný kód příčiny RC2119. Vrácený kód příčiny závisí na diagnostických schopnostech základní služby pro převod dat.

9. Pokud je vrácen kód dokončení CCWARN a je relevantní více než jeden kód příčiny, pořadí priorit je následující:

   a. Následující důvod má přednost před všemi ostatními:

      - RC2079

   b. Další v pořadí je následující příčina:

      - RC2110

   c. Pořadí priorit ve zbývajících kódech příčiny není definováno.

10. Po dokončení volání MQGET:

    - Následující kód příčiny označuje, že zpráva byla úspěšně převedena:
      - RCNONE
    - Následující kód příčiny označuje, že zpráva *mohla* být úspěšně převedena (zkontrolujte pole MDCSI a MDENC v parametru **MSGDSC** , abyste zjistili):
      - RC2079
    - Všechny ostatní kódy příčiny označují, že zpráva nebyla převedena.

Následující zpracování je specifické pro vestavěné formáty; nelze je použít pro formáty definované uživatelem:

1. S výjimkou následujících formátů:

   - FMADMN
   - FMEVNT
   - FMIMVS
   - Prostředek FMPCF
   - FMSTR

   žádný z vestavěných formátů nelze převést ze znakových sad nebo do znakových sad, které nemají znaky SBCS pro znaky platné v názvech front. Pokud se pokusíte provést takový převod, zpráva se vrátí nepřevedená, s kódem dokončení CCWARN a kódem příčiny RC2111 nebo RC2115, podle potřeby.

   Znaková sada Unicode UTF-16 je příkladem znakové sady, která neobsahuje znaky SBCS pro znaky platné v názvech front.

2. Pokud jsou data zprávy pro vestavěný formát oseknuta, pole ve zprávě, která obsahují délku řetězců nebo počet prvků nebo struktur, nejsou upravena tak, aby odrážela délku dat vrácených do aplikace; hodnoty vrácené pro tato pole v datech zprávy jsou hodnoty použitelné pro zprávu před oseknutím.

   Při zpracování zpráv, jako je například zkrácená zpráva FMADMN, je třeba dbát na to, aby se aplikace nepokoušela o přístup k datům za koncem vrácených dat.

3. Je-li název formátu FMDLH, data zprávy začínají strukturou MQDLH a za nimi může následovat nula nebo více bajtů dat zprávy aplikace. Formát, znaková sada a kódování dat zprávy aplikace jsou definovány poli DLFMT, DLCSIa DLENC ve struktuře MQDLH na začátku zprávy. Vzhledem k tomu,

že struktura MQDLH a data zpráv aplikace mohou mít různé znakové sady a kódování, je možné, že jedna, druhá nebo obě struktury MQDLH a data zpráv aplikace vyžadují převod.

Správce front nejprve převede strukturu MQDLH podle potřeby. Pokud je převod úspěšný nebo struktura MQDLH převod nevyžaduje, správce front zkontroluje pole DLCSI a DLENC ve struktuře MQDLH, aby zjistil, zda je vyžadován převod dat zprávy aplikace. Je-li převod vyžadován, správce front vyvolá uživatelskou proceduru s názvem zadaným v poli DLFMT ve struktuře MQDLH nebo provede převod sám (pokud DLFMT je název vestavěného formátu).

Pokud volání MQGET vrátí kód dokončení CCWARN a kód příčiny je jedním z těch, které indikují, že převod nebyl úspěšný, použije se jedna z následujících možností:

- Strukturu MQDLH nelze převést. V tomto případě nebudou převedena ani data zprávy aplikace.
- Struktura MQDLH byla převedena, ale data zprávy aplikace nikoli.

Aplikace může zkontrolovat hodnoty vrácené v polích MDCSI a MDENC v parametru **MSGDSC** a hodnoty ve struktuře MQDLH, aby zjistila, která z předchozích hodnot se použije.

4. Je-li název formátu FMXQH, data zprávy začínají strukturou MQXQH a za nimi může následovat nula nebo více bajtů dalších dat. Tato další data jsou obvykle data zprávy aplikace (která mohou mít nulovou délku), ale na začátku dalších dat může být také jedna nebo více dalších struktur záhlaví IBM MQ .

   Struktura MQXQH musí být ve znakové sadě a kódování správce front. Formát, znaková sada a kódování dat následujících po struktuře MQXQH jsou dány poli MDFMT, MDCSIa MDENC ve struktuře MQMD obsažené v rámci MQXQH. Pro každou následnou přítomnou strukturu záhlaví IBM MQ pole MDFMT, MDCSIa MDENC ve struktuře popisují data, která následují za touto strukturou; tato data jsou buď jinou strukturou záhlaví IBM MQ , nebo daty zprávy aplikace.

   Je-li pro zprávu FMXQH uvedena volba GMCONV, data zprávy aplikace a určité struktury záhlaví MQ se převedou, ale data ve struktuře MQXQH nikoli. Při návratu z volání MQGET tedy:

   - Hodnoty polí MDFMT, MDCSIa MDENC v parametru **MSGDSC** popisují data ve struktuře MQXQH, nikoli data zprávy aplikace; hodnoty proto nebudou stejné jako hodnoty určené aplikací, která vydala volání MQGET.

     Výsledkem je, že aplikace, která opakovaně získává zprávy z přenosové fronty se zadanou volbou GMCONV, musí před každým voláním MQGET resetovat pole MDCSI a MDENC v parametru **MSGDSC** na hodnoty nezbytné pro data zprávy aplikace.
   - Hodnoty polí MDFMT, MDCSIa MDENC v poslední struktuře záhlaví MQ popisují data zprávy aplikace. Pokud nejsou k dispozici žádné jiné struktury záhlaví IBM MQ , jsou data zprávy aplikace popsána těmito poli ve struktuře MQMD v rámci struktury MQXQH. Je-li převod úspěšný, hodnoty budou stejné jako hodnoty zadané v parametru **MSGDSC** aplikací, která vydala volání MQGET.

   Pokud se jedná o zprávu distribučního seznamu, za strukturou MQXQH následuje struktura MQDH (plus její pole záznamů MQOR a MQPMR), za kterou může následovat nula nebo více dalších struktur záhlaví IBM MQ a žádný nebo více bajtů dat zprávy aplikace. Podobně jako struktura MQXQH musí být struktura MQDH ve znakové sadě a kódování správce front a není převedena ve volání MQGET, i když je zadána volba GMCONV.

   Dříve popsané zpracování struktur MQXQH a MQDH je primárně určeno pro použití agenty kanálů zpráv při získání zpráv z přenosových front.

IBM i Převod zpráv sestavy na IBM i

Zpráva sestavy může obsahovat různé množství dat zprávy aplikace podle voleb sestavy určených odesilatelem původní zprávy.

Zpráva sestavy může obsahovat zejména:

1. Žádná data zprávy aplikace
2. Některá data zprávy aplikace z původní zprávy

   K tomu dochází, když odesílatel původní zprávy uvádí RO* D a zpráva je delší než 100 bajtů.

3. Všechna data zprávy aplikace z původní zprávy

   K tomu dochází, když odesílatel původní zprávy uvádí RO* F nebo uvádí RO* D a zpráva je 100 bajtů nebo kratší.

Když správce front nebo agent kanálu zpráv vygeneruje zprávu sestavy, zkopíruje název formátu z původní zprávy do pole *MDFMT* v řídicích informacích ve zprávě sestavy. Název formátu ve zprávě sestavy proto může znamenat délku dat, která se liší od délky uvedené ve zprávě sestavy (dříve popsané případy 1 a 2).

Je-li při načítání zprávy sestavy zadána volba GMCONV:

- V případě 1 popsaném dříve nebude uživatelská procedura převodu dat vyvolána (protože zpráva sestavy nebude mít žádná data).
- Pro případ 3 popsaný dříve název formátu správně implikuje délku dat zprávy.
- V případě 2 popsaném dříve však bude vyvolána uživatelská procedura pro převod dat za účelem převodu zprávy, která je *kratší* než délka odvozená z názvu formátu.

  Kromě toho bude kód příčiny předaný uživatelské proceduře obvykle RCNONE (to znamená, že kód příčiny nebude označovat, že zpráva byla oříznuta). K tomu dochází proto, že data zprávy byla zkrácena *odesilatelem* zprávy sestavy, a nikoli správcem front příjemce v reakci na volání MQGET.

Vzhledem k těmto možnostem by uživatelská procedura převodu dat neměla používat název formátu k odpočtu délky dat, která jí byla předána; místo toho by uživatelská procedura měla zkontrolovat délku poskytnutých dat a být připravena převést méně dat, než je délka odvozená z názvu formátu. Pokud lze data úspěšně převést, uživatelská procedura by měla vrátit kód dokončení CCOK a kód příčiny RCNONE. Délka dat zprávy, která se mají převést, je předána uživatelské proceduře jako parametr **INLEN** .

Programovací rozhraní citlivé na produkt

Pokud zpráva sestavy obsahuje informace o aktivitě, která se uskutečnila, je známa jako sestava aktivity. Příklady činností jsou:

- MCA odesílající zprávu z fronty po kanálu
- MCA přijímající zprávu z kanálu a vkládající ji do fronty
- nedoručitelné zprávy MCA zařazující nedoručitelnou zprávu do fronty
- MCA-získání zprávy z fronty a její vyřazení
- obslužná rutina nedoručených zpráv, která umístí zprávu zpět do fronty
- příkazový server zpracovávající požadavek PCF-zprostředkovatel zpracovávající požadavek na publikování
- uživatelská aplikace, která získává zprávu z fronty-uživatelská aplikace, která prochází zprávu ve frontě

Každá aplikace, včetně správce front, může přidat některá data zprávy do sestavy aktivity za záhlaví sestavy. Množství dat, která by měla být dodána, pokud jsou některá odeslána, nejsou pevná a je o tom rozhodnuto v žádosti. Vrácené informace by měly být užitečné pro aplikaci, která zpracovává sestavu aktivity. Sestavy aktivity správce front vrátí všechny standardní struktury záhlaví IBM MQ (začínající na 'MQH') obsažené v původní zprávě. To zahrnuje například záhlaví MQRFH2 , která byla zahrnuta do původní zprávy. Správce front také vrátí nalezené záhlaví MQCFH, ale nikoli parametry PCF, které jsou k němu přidruženy. To dává monitorovacím aplikacím představu o tom, o čem byla zpráva.

IBM i MQDXP (parametr uživatelské procedury pro převod dat) na systému IBM i

Blok parametrů ukončení konverze dat.

Přehled

Účel: Struktura MQDXP je parametr, který správce front předá uživatelské proceduře pro převod dat při vyvolání uživatelské procedury pro převod dat zprávy v rámci zpracování volání MQGET. Podrobnosti o uživatelské proceduře pro převod dat naleznete v popisu volání MQCONVX.

Znaková sada a kódování: Znaková data v prostředí MQDXP jsou ve znakové sadě lokálního správce front. Toto je dáno atributem správce front **CodedCharSetId** . Číselná data v MQDXP jsou v nativním kódování počítače; toto je dáno ENNAT.

Použití: Uživatelská procedura může změnit pouze pole *DXLEN*, *DXCC*, *DXREA* a *DXRES* v MQDXP; změny ostatních polí se ignorují. Pole *DXLEN* však nelze změnit, pokud převáděná zpráva je segmentem, který obsahuje pouze část logické zprávy.

Když se řízení vrátí do správce front z uživatelské procedury, zkontroluje správce front hodnoty vrácené v MQDXP. Pokud vrácené hodnoty nejsou platné, správce front pokračuje ve zpracování, jako by uživatelská procedura vrátila hodnotu XRFAIL v souboru *DXRES* ; Správce front však v tomto případě ignoruje hodnoty polí *DXCC* a *DXREA* vrácené uživatelskou procedurou a místo toho použije hodnoty, které tato pole měla na *vstupu* do uživatelské procedury. Následující hodnoty v MQDXP způsobují toto zpracování:

- Pole *DXRES* není XROK a není XRFAIL
- Pole *DXCC* není CCOK a není CCWARN
- Pole *DXLEN* menší než nula nebo *DXLEN* se změnilo, když je převáděná zpráva segmentem, který obsahuje pouze část logické zprávy.
- “Pole” na stránce 1435
- “Deklarace RPG (kopírovat soubor CMQDXPH)” na stránce 1439

Pole

Struktura MQDXP obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

DXAOP (10místné celé číslo se znaménkem)

Volby aplikace.

Jedná se o kopii pole *GMOPT* struktury MQGMO určené aplikací vydávající volání MQGET. Je možné, že bude nutné tyto údaje prozkoumat, aby se zjistilo, zda byla zadána volba GMATM.

Toto je vstupní pole pro ukončení.

DXCC (10místné celé číslo se znaménkem)

Kód dokončení.

Je-li uživatelská procedura vyvolána, obsahuje kód dokončení, který bude vrácen aplikaci, která vydala volání MQGET, pokud se uživatelská procedura rozhodne nic nedělat. Vždy je to CCWARN, protože buď byla zpráva zkrácena, nebo zpráva vyžaduje konverzi, a to ještě nebylo provedeno.

Ve výstupu uživatelské procedury toto pole obsahuje kód dokončení, který má být vrácen aplikaci v parametru **CMPCOD** volání MQGET; platné jsou pouze hodnoty CCOK a CCWARN. V popisu pole *DXREA* naleznete návrhy, jak by měla uživatelská procedura nastavit toto pole na výstupu.

Toto je vstupní/výstupní pole pro ukončení.

DXCSI (10místné celé číslo se znaménkem)

Znaková sada požadovaná aplikací.

Jedná se o identifikátor kódované znakové sady znakové sady požadované aplikací vydávající volání MQGET. Další podrobnosti naleznete v poli *MDCSI* ve struktuře MQMD. Pokud aplikace určuje speciální hodnotu CSQM ve volání MQGET, správce front tuto hodnotu před vyvoláním uživatelské procedury změní na skutečný identifikátor znakové sady znakové sady používané správcem front.

Je-li převod úspěšný, uživatelská procedura by jej měla zkopírovat do pole *MDCSI* v deskriptoru zprávy.

Toto je vstupní pole pro ukončení.

DXENC (10místné celé číslo se znaménkem)

Číselné kódování požadované aplikací.

Jedná se o číselné kódování vyžadované aplikací vydávající volání MQGET. Další podrobnosti naleznete v poli *MDENC* ve struktuře MQMD.

Je-li převod úspěšný, uživatelská procedura by jej měla zkopírovat do pole *MDENC* v deskriptoru zprávy.

Toto je vstupní pole pro ukončení.

DXHCN (10místné celé číslo se znaménkem)

Manipulátor připojení.

Jedná se o manipulátor připojení, který lze použít pro volání MQXCNVC. Tento manipulátor nemusí být nutně stejný jako manipulátor určený aplikací, která zadala volání MQGET.

DXLEN (10místné celé číslo se znaménkem)

Délka dat zprávy v bajtech.

Při vyvolání uživatelské procedury toto pole obsahuje původní délku dat zprávy aplikace. Pokud byla zpráva oříznuta, aby se vešla do vyrovnávací paměti poskytnuté aplikací, velikost zprávy poskytnuté uživatelské proceduře bude *menší* než hodnota *DXLEN*. Velikost zprávy poskytnuté uživatelské proceduře je vždy dána parametrem **INLEN** uživatelské procedury, bez ohledu na případné oříznutí.

Oříznutí je označeno polem *DXREA* , které má hodnotu RC2079 na vstupu do ukončení.

Většina převodů nebude muset tuto délku měnit, ale uživatelská procedura tak může učinit v případě potřeby; hodnota nastavená uživatelskou procedurou je vrácena aplikaci v parametru **DATLEN** volání MQGET. Tuto délku však nelze změnit, pokud převáděná zpráva je segmentem, který obsahuje pouze část logické zprávy. Důvodem je, že změna délky by způsobila, že by posuny pozdějších segmentů v logické zprávě byly nesprávné.

Všimněte si, že pokud uživatelská procedura chce změnit délku dat, mějte na paměti, že správce front již rozhodl, zda se data zprávy vejdou do vyrovnávací paměti aplikace, a to na základě délky *nepřevedených* dat. Toto rozhodnutí určuje, zda je zpráva odebrána z fronty (nebo zda je přesunut kurzor procházení pro požadavek na procházení) a není ovlivněna žádnou změnou délky dat způsobenou konverzí. Z tohoto důvodu se doporučuje, aby uživatelské procedury převodu nezpůsobily změnu délky dat zprávy aplikace.

Pokud převod znaků znamená změnu délky, lze řetězec převést na jiný řetězec se stejnou délkou v bajtech, oseknout koncové mezery nebo podle potřeby vyplňovat mezerami.

Uživatelská procedura není vyvolána, pokud zpráva neobsahuje žádná data zprávy aplikace; proto je hodnota *DXLEN* vždy větší než nula.

Toto je vstupní/výstupní pole pro ukončení.

DXREA (10místné celé číslo se znaménkem)

Kód příčiny kvalifikující *DXCC*.

Je-li uživatelská procedura vyvolána, obsahuje kód příčiny, který bude vrácen aplikaci, která vydala volání MQGET, pokud se uživatelská procedura rozhodne nic nedělat. Mezi možné hodnoty patří RC2079, označující, že zpráva byla oříznuta, aby se vešla do vyrovnávací paměti poskytnuté aplikací, a RC2119, označující, že zpráva vyžaduje převod, ale ještě nebyla provedena.

Ve výstupu uživatelské procedury toto pole obsahuje důvod, proč se má vrátit aplikaci v parametru **REASON** volání MQGET; doporučuje se následující:

- Pokud *DXREA* měla hodnotu RC2079 na vstupu do uživatelské procedury, pole *DXREA* a *DXCC* by neměla být změněna, bez ohledu na to, zda je převod úspěšný nebo selže.

  (Pokud pole *DXCC* není CCOK, aplikace, která načte zprávu, může identifikovat selhání převodu porovnáním vrácených hodnot *MDENC* a *MDCSI* v deskriptoru zprávy s požadovanými hodnotami; na rozdíl od toho aplikace nemůže odlišit oseknutou zprávu od zprávy, která právě nainstalovala vyrovnávací paměť. Z tohoto důvodu by měla být funkce RC2079 vrácena přednostně před kterýmikoli příčinami, které indikují selhání převodu.)

- Pokud má parametr *DXREA* na vstupu do uživatelské procedury jinou hodnotu:
  - Pokud je převod úspěšný, *DXCC* by měl být nastaven na CCOK a *DXREA* na RCNONE.
  - Pokud se převod nezdaří nebo se zpráva rozbalí a musí být oříznuta tak, aby se vešla do vyrovnávací paměti, *DXCC* by mělo být nastaveno na hodnotu CCWARN (nebo ponecháno beze změny) a *DXREA* nastaveno na jednu z hodnot i v následujícím seznamu, aby se označila povaha selhání.

    Všimněte si, že pokud je zpráva po převodu pro vyrovnávací paměť příliš velká, měla by být zkrácena pouze v případě, že aplikace, která vydala volání MQGET, zadala volbu GMATM:

    - Pokud tato volba byla zadána, měla by být vrácena příčina RC2079 .
    - Pokud neuvádí tuto volbu, zpráva by měla být vrácena nepřevedená, s kódem příčiny RC2120.

Kódy příčiny v následujícím seznamu jsou doporučeny pro použití uživatelskou procedurou k označení důvodu, že konverze selhala, ale uživatelská procedura může vrátit jiné hodnoty ze sady kódů RC*, pokud je to považováno za vhodné. Kromě toho je rozsah hodnot RC0900 až RC0999 přidělen pro použití uživatelskou procedurou k označení podmínek, že uživatelská procedura chce komunikovat s aplikací vydávající volání MQGET.

Poznámka: Pokud zprávu nelze úspěšně převést, musí uživatelská procedura vrátit hodnotu XRFAIL do pole *DXRES* , aby správce front vrátil nepřevedenou zprávu. To platí bez ohledu na kód příčiny vrácený v poli *DXREA* .

RC0900
(900, X'384 ') Nejnižší hodnota pro kód příčiny definovaný aplikací.

RC0999
(999, X'3E7') Nejvyšší hodnota pro kód příčiny definovaný aplikací.

RC2120
(2120, X'848 ') Převedená data jsou příliš velká pro vyrovnávací paměť.

RC2119
(2119, X'847 ') Data zprávy nebyla převedena.

RC2111
(2111, X'83F') Identifikátor zdrojové kódované znakové sady není platný.

RC2113
(2113, X'841 ') kódování Packed-decimal ve zprávě nebylo rozpoznáno.

RC2114
(2114, X'842 ') kódování s pohyblivou řádovou čárkou ve zprávě nebylo rozpoznáno.

RC2112
(2112, X'840 ') kódování zdrojového celého čísla nebylo rozpoznáno.

RC2115
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

RC2117
(2117, X'845 ') kódování Packed-decimal určené příjemcem nebylo rozpoznáno.

RC2118
(2118, X'846 ') kódování s pohyblivou řádovou čárkou určené příjemcem nebylo rozpoznáno.

RC2116
(2116, X'844 ') kódování cílového celého čísla nebylo rozpoznáno.

RC2079
(2079, X'81F') Vrácena zkrácená zpráva (zpracování dokončeno).

Toto je vstupní/výstupní pole pro ukončení.

DXRES (10místné celé číslo se znaménkem)

Odezva z ukončení.

Toto je nastaveno uživatelskou procedurou, která označuje úspěch nebo neúspěch převodu. Musí to být jeden z následujících:

XROK
Převod byl úspěšný.

Pokud uživatelská procedura určuje tuto hodnotu, vrátí aplikaci, která vydala volání MQGET, následující příkaz:

- Hodnota pole *DXCC* na výstupu z ukončení
- Hodnota pole *DXREA* na výstupu z ukončení
- Hodnota pole *DXLEN* na výstupu z ukončení
- Obsah výstupní vyrovnávací paměti uživatelské procedury *OUTBUF*. Počet vrácených bajtů je menší z parametru **OUTLEN** uživatelské procedury a hodnota pole *DXLEN* na výstupu z uživatelské procedury.

Pokud jsou pole *MDENC* a *MDCSI* v parametru deskriptoru zprávy uživatelské procedury *obojí* beze změny, správce front vrátí:

- Hodnota polí *MDENC* a *MDCSI* ve struktuře MQDXP na *vstupu* do uživatelské procedury

Pokud došlo ke změně jednoho nebo obou polí *MDENC* a *MDCSI* v parametru deskriptoru zprávy uživatelské procedury, vrátí správce front:

- Hodnota polí *MDENC* a *MDCSI* v parametru deskriptoru zprávy uživatelské procedury na výstupu uživatelské procedury
-

XRFAIL
Převod byl neúspěšný.

Pokud uživatelská procedura určuje tuto hodnotu, vrátí aplikaci, která vydala volání MQGET, následující příkaz:

- Hodnota pole *DXCC* na výstupu z ukončení
- Hodnota pole *DXREA* na výstupu z ukončení
- Hodnota pole *DXLEN* na *vstupu* do uživatelské procedury
- Obsah vstupní vyrovnávací paměti uživatelské procedury *INBUF*. Počet vrácených bajtů je dán parametrem **INLEN** .

Pokud uživatelská procedura změnila hodnotu *INBUF*, výsledky nejsou definovány.

DXRES je výstupní pole z ukončení.

DXSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

DXSIDV
Identifikátor pro strukturu parametru uživatelské procedury konverze dat.

Toto je vstupní pole pro ukončení.

DXVER (10místné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

DXVER1
Číslo verze pro strukturu výstupních parametrů konverze dat.

Následující konstanta určuje číslo verze aktuální verze:

DXVERC
Aktuální verze struktury parametrů uživatelské procedury konverze dat.

Poznámka: Při zavedení nové verze této struktury se nezmění rozvržení stávající části. Uživatelská procedura by proto měla zkontrolovat, zda je pole *DXVER* stejné nebo větší než nejnižší verze, která obsahuje pole, která uživatelská procedura potřebuje použít.

Toto je vstupní pole pro ukončení.

DXXOP (10místné celé číslo se znaménkem)

Vyhrazeno.

Toto je vyhrazené pole; jeho hodnota je 0.

Deklarace RPG (kopírovat soubor CMQDXPH)

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQDXP Structure
D*
D* Structure identifier
D  DXSID                  1      4
D* Structure version number
D  DXVER                  5      8I 0
D* Reserved
D  DXXOP                  9     12I 0
D* Application options
D  DXAOP                 13     16I 0
D* Numeric encoding required by application
D  DXENC                 17     20I 0
D* Character set required by application
D  DXCSI                 21     24I 0
D* Length in bytes of message data
D  DXLEN                 25     28I 0
D* Completion code
D  DXCC                  29     32I 0
D* Reason code qualifying DXCC
D  DXREA                 33     36I 0
D* Response from exit
D  DXRES                 37     40I 0
D* Connection handle
D  DXHCN                 41     44I 0
```

IBM i MQXCNVC (Převod znaků) na systému IBM i

Volání MQXCNVC převádí znaky z jedné znakové sady na jinou.

Toto volání je součástí rozhraní DCI (IBM MQ Data Conversion Interface), které je jedním z rozhraní rámce IBM MQ . Poznámka: Toto volání lze použít pouze z uživatelské procedury pro převod dat.

- “Syntaxe” na stránce 1439
- “Parametry” na stránce 1439
- “Vyvolání RPG (ILE)” na stránce 1443

Syntaxe

```
MQXCNVC HCONN, OPTS, SRCCSI, SRCLEN, SRCBUF, TGTCSI, TGTLEN,
   TGTBUF, DATLEN, CMPCOD, REASON)
```

Parametry

Volání MQXCNVC má následující parametry:

HCONN (10místné celé číslo se znaménkem)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Obvykle by měl být popisovačem předaným uživatelské proceduře pro převod dat v poli DXHCN struktury MQDXP. Tento popisovač nemusí být nutně stejný jako popisovač určený aplikací, která vydala volání MQGET.

V systému IBM ilze pro HCONNzadat následující speciální hodnotu:

HCDEFH
Výchozí manipulátor připojení.

OPTS (10místné celé číslo se znaménkem)-vstup

Volby, které řídí akci MQXCNVC.

Lze zadat nula nebo více voleb popsaných dále v této části. Pokud se požaduje více než jedna, hodnoty lze přidat (nepřidávejte stejnou konstantu více než jednou).

Výchozí-volba převodu: Následující volba řídí použití výchozího převodu znaků:

DCCDEF
Výchozí převod.

Tato volba určuje, že lze použít výchozí znakovou konverzi, pokud není podporována jedna nebo obě znakové sady uvedené ve volání. To umožňuje správci front při převodu řetězce použít výchozí znakovou sadu určenou instalací, která se blíží určené znakové sadě.

Poznámka: Výsledkem použití přibližné znakové sady pro převod řetězce je, že některé znaky mohou být nesprávně převedeny. Tomu se lze vyhnout tím, že v řetězci budou použity pouze znaky, které jsou společné jak pro zadanou znakovou sadu, tak pro výchozí znakovou sadu.

Výchozí znakové sady jsou definovány volbou konfigurace při instalaci nebo restartování správce front.

Není-li zadán parametr DCCDEF, použije správce front k převodu řetězce pouze určené znakové sady a volání se nezdaří, není-li jedna nebo obě znakové sady podporovány.

Volba vyplnění: Následující volba umožňuje správci front zaplnit převedený řetězec mezerami nebo zahodit nevýznamné koncové znaky, aby převedený řetězec odpovídal cílové vyrovnávací paměti:

DCCFIL
Vyplnit cílovou vyrovnávací paměť.

Tato volba vyžaduje, aby se převod uskutečnil tak, aby cílová vyrovnávací paměť byla zcela vyplněna:

- Pokud se řetězec při převodu uzavírá, přidají se koncové mezery, aby se vyplnila cílová vyrovnávací paměť.
- Pokud se řetězec při převodu rozbalí, koncové znaky, které nejsou významné, se zahodí, aby převedený řetězec odpovídal cílové vyrovnávací paměti. Pokud to lze úspěšně provést, volání se dokončí s kódem příčiny CCOK a kódem příčiny RCNONE.

  Pokud existuje příliš málo nevýznamných koncových znaků, je do cílové vyrovnávací paměti umístěna velká část řetězce, která se vejde, a volání je dokončeno s kódem příčiny RC2120a CCWARN.

  Nevýznamné znaky jsou:

  – Koncové mezery
  – Znaky následující za prvním znakem null v řetězci (ale bez samotného prvního znaku null)
- Pokud je řetězec TGTCSI a TGTLEN takový, že cílovou vyrovnávací paměť nelze zcela nastavit s platnými znaky, volání selže s CCFAIL a kódem příčiny RC2144. K tomu může dojít, když je TGTCSI ryzí znaková sada DBCS (například UTF-16), ale TGTLEN uvádí délku, která je lichý počet bajtů.

- TGTLEN může být menší nebo větší než SRCLEN. Při návratu z MQXCNVC má DATLEN stejnou hodnotu jako TGTLEN.

Není-li tato volba uvedena:

- Řetězec může podle potřeby uzavřít smlouvu nebo rozšířit v cílové vyrovnávací paměti. Nevýznamné koncové znaky se nepřidávají ani neodstraňují.

  Pokud se převedený řetězec vejde do cílové vyrovnávací paměti, volání se dokončí s kódem příčiny CCOK a kódem příčiny RCNONE.

  Pokud je převedený řetězec pro cílovou vyrovnávací paměť příliš velký, je do cílové vyrovnávací paměti umístěna velká část řetězce, která se vejde, a volání je dokončeno s kódem příčiny RC2120a CCWARN. Všimněte si, že v tomto případě lze vrátit méně než *TGTLEN* bajtů.
- TGTLEN může být menší nebo větší než SRCLEN. Při návratu z MQXCNVC je hodnota DATLEN menší nebo rovna hodnotě TGTLEN.

Volby kódování: Následující volby lze použít k určení celočíselného kódování zdrojového a cílového řetězce. Příslušné kódování se použije pouze tehdy, když odpovídající identifikátor znakové sady označuje, že reprezentace znakové sady v hlavní paměti závisí na kódování použitém pro binární celá čísla. To se týká pouze určitých vícebajtových znakových sad (například znakových sad UTF-16).

Kódování je ignorováno, pokud je znaková sada jednobajtová znaková sada (SBCS) nebo vícebajtová znaková sada se znázorněním v hlavní paměti, která není závislá na kódování celých čísel.

Měla by být zadána pouze jedna z hodnot DCCS* v kombinaci s jednou z hodnot DCCT*:

DCCSNA
Kódování zdroje je výchozí pro prostředí a programovací jazyk.

DCCSNO
Kódování zdroje je normální.

DCCSRE
Kódování zdroje je obrácené.

DCCSUN
Kódování zdroje není definováno.

DCCTNA
Cílové kódování je výchozí pro prostředí a programovací jazyk.

DCCTNO
Cílové kódování je normální.

DCCTRE
Cílové kódování je obrácené.

DCCTUN
Cílové kódování není definováno.

Dříve definované hodnoty kódování lze přidat přímo do pole OPTS . Pokud je však zdrojové nebo cílové kódování získáno z pole MDENC v deskriptoru MQMD nebo jiné struktuře, je třeba provést následující zpracování:

1. Kódování celých čísel musí být extrahováno z pole MDENC odstraněním kódování typu float a packed-decimal. Podrobnosti o tomto postupu naleznete v části “Analýza kódování na IBM i” na stránce 1424 .
2. Kódování celých čísel, které je výsledkem kroku 1, musí být před přidáním do pole OPTS vynásobeno příslušným faktorem. Tyto faktory jsou:

   DCCSFA
   Faktor pro kódování zdroje

   DCCTFA
   Faktor pro cílové kódování

Není-li uvedeno, volby kódování jsou standardně nedefinované (DCC* UN). Ve většině případů to neovlivní úspěšné dokončení volání MQXCNVC. Avšak pokud je odpovídající znaková sada vícebajtová znaková sada se znázorněním, které závisí na kódování (například znaková sada UTF-16), volání selže s kódem příčiny RC2112 nebo RC2116 podle potřeby.

Výchozí volba: Pokud není zadána žádná z dříve popsaných voleb, lze použít následující volbu:

DCCNON
Nejsou uvedeny žádné volby.

DCCNON je definována pro pomoc s dokumentací programu. Není zamýšleno, aby tato volba byla použita s jinými, ale protože její hodnota je nula, nelze takové použití zjistit.

SRCCSI (10místné celé číslo se znaménkem)-vstup

Identifikátor kódované znakové sady řetězce před převodem.

Jedná se o identifikátor kódované znakové sady vstupního řetězce v souboru SRCBUF.

SRCLEN (10místné celé číslo se znaménkem)-vstup

Délka řetězce před převodem.

Jedná se o délku vstupního řetězce v bajtech v souboru SRCBUF ; musí být nula nebo větší.

SRCBUF (1bajtový znakový řetězec x SRCLEN)-vstup

Řetězec, který má být převeden.

Jedná se o vyrovnávací paměť obsahující řetězec, který má být převeden z jedné znakové sady na jinou.

TGTCSI (10místné celé číslo se znaménkem)-vstup

Identifikátor kódované znakové sady řetězce po převodu.

Jedná se o identifikátor kódované znakové sady znakové sady, na kterou se má převést SRCBUF .

TGTLEN (10místné celé číslo se znaménkem)-vstup

Délka výstupní vyrovnávací paměti.

Jedná se o délku výstupní vyrovnávací paměti v bajtech TGTBUF ; musí být nula nebo větší. Může být menší nebo větší než SRCLEN.

TGTBUF (1bajtový znakový řetězec x TGTLEN)-výstup

Řetězec po převodu.

Jedná se o řetězec poté, co byl převeden na znakovou sadu definovanou parametrem TGTCSI. Převedený řetězec může být kratší nebo delší než nepřevedený řetězec. Parametr **DATLEN** označuje počet vrácených platných bajtů.

DATLEN (10místné celé číslo se znaménkem)-výstup

Délka výstupního řetězce.

Jedná se o délku řetězce vráceného ve výstupní vyrovnávací paměti TGTBUF. Převedený řetězec může být kratší nebo delší než nepřevedený řetězec.

CMPCOD (10místné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

CCOK
Úspěšné dokončení.

CCWARN (varování)
Varování (částečné dokončení).

CCFAIL
Volání selhalo.

REASON (10místné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující CMPCOD.

Pokud je CMPCOD CCOK:

RCNONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr CMPCOD hodnotu CCWARN:

RC2120
(2120, X'848 ') Převedená data jsou příliš velká pro vyrovnávací paměť.

Má-li parametr CMPCOD hodnotu CCFAIL:

RC2010
(2010, X'7DA') Parametr délky dat není platný.

RC2150
(2150, X'866 ') Řetězec DBCS není platný.

RC2018
(2018, X'7E2') popisovač připojení není platný.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2102
(2102, X'836 ') Nedostatek dostupných systémových prostředků.

RC2145
(2145, X'861 ') Parametr zdrojové vyrovnávací paměti není platný.

RC2111
(2111, X'83F') Identifikátor zdrojové kódované znakové sady není platný.

RC2112
(2112, X'840 ') kódování zdrojového celého čísla nebylo rozpoznáno.

RC2143
(2143, X'85F') Parametr délky zdroje není platný.

RC2071
(2071, X'817 ') Nedostatek dostupného úložiště.

RC2146
(2146, X'862 ') Parametr cílové vyrovnávací paměti není platný.

RC2115
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

RC2116
(2116, X'844 ') kódování cílového celého čísla nebylo rozpoznáno.

RC2144
(2144, X'860 ') Parametr délky cíle není platný.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Další informace o těchto kódech příčiny viz “Návratové kódy pro IBM i (ILE RPG)” na stránce 1418.

Vyvolání RPG (ILE)

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     MQXCNVC(HCONN : OPTS : SRCCSI :
C                                       SRCLEN : SRCBUF : TGTCSI :
```

```
C                                       TGTLEN : TGTBUF : DATLEN :
C                                       CMPCOD : REASON)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
DMQXCNVC          PR                  EXTPROC('MQXCNVC')
D* Connection handle
D HCONN                         10I 0 VALUE
D* Options that control the action of MQXCNVC
D OPTS                          10I 0 VALUE
D* Coded character set identifier of string before conversion
D SRCCSI                        10I 0 VALUE
D* Length of string before conversion
D SRCLEN                        10I 0 VALUE
D* String to be converted
D SRCBUF                          *   VALUE
D* Coded character set identifier of string after conversion
D TGTCSI                        10I 0 VALUE
D* Length of output buffer
D TGTLEN                        10I 0 VALUE
D* String after conversion
D TGTBUF                          *   VALUE
D* Length of output string
D DATLEN                        10I 0
D* Completion code
D CMPCOD                        10I 0
D* Reason code qualifying CMPCOD
D REASON                        10I 0
```

IBM i MQCONVX (uživatelská procedura převodu dat) na systému IBM i

Tato definice volání popisuje parametry, které se předávají uživatelské proceduře pro převod dat.

Správce front neposkytl žádný vstupní bod s názvem MQCONVX (viz poznámka k použití “11” na stránce 1446).

Tato definice je součástí rozhraní DCI (IBM MQ Data Conversion Interface), které je jedním z rozhraní rámce IBM MQ .

Syntaxe

MQCONVX *(MQDXP, MQMD, INLEN, INBUF, OUTLEN, OUTBUF)*

Poznámky k použití

1. Uživatelská procedura pro převod dat je uživatelská procedura, která přijímá řízení během zpracování volání MQGET. Funkce prováděná uživatelskou procedurou pro převod dat je definována poskytovatelem uživatelské procedury; uživatelská procedura však musí být v souladu s pravidly popsanými zde a v přidružené struktuře parametrů MQDXP.

   Programovací jazyky, které lze použít pro uživatelskou proceduru převodu dat, jsou určeny prostředím.
2. Uživatelská procedura je vyvolána pouze v případě, že *všechny* z následujících příkazů jsou pravdivé:
   - Volba GMCONV je určena ve volání MQGET.
   - Pole *MDFMT* v deskriptoru zprávy není FMNONE.
   - Zpráva již není v požadované reprezentaci, tj. jedna nebo obě hodnoty *MDCSI* a *MDENC* zprávy se liší od hodnoty určené aplikací v deskriptoru zprávy zadaném ve volání MQGET.

- Správce front dosud úspěšně neprovedl převod.
- Délka vyrovnávací paměti aplikace je větší než nula.
- Délka dat zprávy je větší než nula.
- Kód příčiny během operace MQGET je RCNONE nebo RC2079 .

3. Při zápisu uživatelské procedury je třeba zvážit kódování uživatelské procedury způsobem, který jí umožní převést zprávy, které byly oseknuty. Zkrácené zprávy mohou vzniknout následujícími způsoby:

   - Přijímající aplikace poskytuje vyrovnávací paměť, která je menší než zpráva, ale určuje volbu GMATM pro volání MQGET.

     V tomto případě bude mít pole *DXREA* v parametru **MQDXP** na vstupu do uživatelské procedury hodnotu RC2079.

   - Odesilatel zprávy ji před odesláním ořízne. K tomu může dojít například u zpráv sestavy (další podrobnosti viz "Převod zpráv sestavy na IBM i" na stránce 1433).

     V tomto případě bude mít pole *DXREA* v parametru **MQDXP** na vstupu do uživatelské procedury hodnotu RCNONE (pokud přijímající aplikace poskytla vyrovnávací paměť, která byla dostatečně velká pro zprávu).

   Proto hodnotu pole *DXREA* na vstupu do uživatelské procedury nelze vždy použít k rozhodnutí, zda byla zpráva oříznuta.

   Rozlišovací charakteristikou zkrácené zprávy je, že délka poskytnutá uživatelské proceduře v parametru **INLEN** bude *menší* než délka odvozená z názvu formátu obsaženého v poli *MDFMT* v deskriptoru zprávy. Uživatelská procedura by proto měla před pokusem o převod libovolných dat zkontrolovat hodnotu *INLEN* ; uživatelská procedura *by neměla* předpokládat, že bylo poskytnuto úplné množství dat odvozených z názvu formátu.

   Pokud nebyla uživatelská procedura zapsána pro převod oseknutých zpráv a hodnota **INLEN** je menší než očekávaná hodnota, měla by uživatelská procedura vrátit hodnotu XRFAIL v poli *DXRES* parametru **MQDXP** s hodnotou pole *DXCC* nastavenou na CCWARN a hodnotu pole *DXREA* nastavenou na RC2110.

   Pokud byla uživatelská procedura *byla* zapsána pro převod oseknutých zpráv, měla by uživatelská procedura převést co nejvíce dat (viz další poznámka o použití), přičemž je třeba dbát na to, aby se nepokoušela zkoumat nebo převádět data po skončení *INBUF*. Pokud je převod úspěšně dokončen, uživatelská procedura by měla ponechat pole *DXREA* v parametru **MQDXP** beze změny. Vrátí hodnotu RC2079 , pokud byla zpráva oseknuta správcem front příjemce, a hodnotu RCNONE, pokud byla zpráva oseknuta odesilatelem zprávy.

   Je také možné, aby zpráva rozbalila *během převodu* do bodu, kde je větší než *OUTBUF*. V tomto případě musí uživatelská procedura rozhodnout, zda zprávu oříznout; pole *DXAOP* v parametru **MQDXP** bude označovat, zda přijímající aplikace zadala volbu GMATM.

4. Obecně se doporučuje, aby byla převedena všechna data ve zprávě poskytnuté uživatelské proceduře v produktu *INBUF* , nebo aby žádná z nich nebyla. Výjimka se však vyskytne, pokud je zpráva oseknuta, buď před konverzí, nebo během konverze; v tomto případě může být na konci vyrovnávací paměti neúplná položka (například jeden bajt dvoubajtového znaku nebo 3 bajty 4bajtového celého čísla). V této situaci se doporučuje, aby byla neúplná položka vynechána a nepoužívané bajty v souboru *OUTBUF* nastaveny na hodnoty null. Avšak úplné prvky nebo znaky v poli nebo řetězci *by měly* být převedeny.
5. Když je uživatelská procedura poprvé potřebná, pokusí se správce front načíst objekt, který má stejný název jako formát (kromě rozšíření). Načtený objekt musí obsahovat uživatelskou proceduru, která zpracovává zprávy s tímto názvem formátu. Doporučuje se, aby název uživatelské procedury a název objektu, který uživatelskou proceduru obsahuje, byly identické, ačkoli to ne všechna prostředí vyžadují.
6. Nová kopie uživatelské procedury se načte, když se aplikace pokusí načíst první zprávu, která používá tento soubor *MDFMT* od připojení aplikace ke správci front. Nová kopie může být také načtena jindy, pokud správce front zrušil dříve načtenou kopii. Z tohoto důvodu by se uživatelská procedura

neměla pokoušet o použití statického úložiště ke komunikaci informací z jednoho vyvolání uživatelské procedury do dalšího-uživatelská procedura může být mezi těmito dvěma vyvoláními uvolněna.

7. Pokud existuje uživatelská procedura se stejným názvem jako jeden z vestavěných formátů podporovaných správcem front, uživatelská procedura nenahradí vestavěnou převodní rutinu. Jedinými okolnostmi, za kterých je takový výstup vyvolán, jsou:
   - Pokud vestavěná převodní rutina nemůže zpracovat převody do nebo ze zapojených *MDCSI* nebo *MDENC* , nebo
   - Pokud se vestavěné převodní rutině nepodařilo převést data (například proto, že existuje pole nebo znak, který nelze převést).
8. Rozsah uživatelské procedury je závislý na prostředí. Názvy *MDFMT* by měly být vybrány tak, aby se minimalizovalo riziko kolizí s jinými formáty. Doporučuje se, aby začínali znaky, které identifikují aplikaci definující název formátu.
9. Uživatelská procedura převodu dat je spuštěna v prostředí, jako je prostředí programu, který vydal volání MQGET; prostředí zahrnuje adresní prostor a profil uživatele (je-li to možné). Program může být agentem kanálu zpráv odesílajícím zprávy do cílového správce front, který nepodporuje převod zpráv. Uživatelská procedura nemůže ohrozit integritu správce front, protože není spuštěna v prostředí správce front.
10. Jediným voláním MQI, které může uživatelská procedura použít, je MQXCNVC; pokus o použití jiných volání MQI selže s kódem příčiny RC2219nebo s jinými nepředvídatelnými chybami.
11. Správce front neposkytl žádný vstupní bod s názvem MQCONVX. Název uživatelské procedury by měl být stejný jako název formátu (název obsažený v poli *MDFMT* v deskriptoru MQMD), ačkoli to není vyžadováno ve všech prostředích.

Parametry

Volání MQCONVX má následující parametry:

MQDXP (MQDXP)-vstupní/výstupní

Blok parametrů ukončení konverze dat.

Tato struktura obsahuje informace týkající se vyvolání uživatelské procedury. Uživatelská procedura nastaví informace v této struktuře, aby označila výsledek převodu. Podrobnosti o polích v této struktuře viz “MQDXP (parametr uživatelské procedury pro převod dat) na systému IBM i” na stránce 1434 .

MQMD (MQMD)-vstupní/výstupní

Deskriptor zprávy.

Při vstupu do uživatelské procedury se jedná o deskriptor zprávy, který by byl vrácen aplikaci, pokud by nebyla provedena žádná konverze. Proto obsahuje *MDFMT*, *MDENC*a *MDCSI* nepřevedené zprávy obsažené v souboru *INBUF*.

Poznámka: Parametr **MQMD** předaný uživatelské proceduře je vždy nejnovější verzí MQMD podporovanou správcem front, který uživatelskou proceduru vyvolává. Pokud má být uživatelská procedura přenositelná mezi různými prostředími, měla by uživatelská procedura zkontrolovat pole *MDVER* v souboru *MQMD* a ověřit, že pole, ke kterým uživatelská procedura potřebuje přístup, jsou ve struktuře přítomna.

V systému IBM ije uživatelské proceduře předáno rozhraní MQMD version-2 .

Na výstupu by uživatelská procedura měla změnit pole *MDENC* a *MDCSI* na hodnoty požadované aplikací, pokud byl převod úspěšný; tyto změny se odrazí zpět do aplikace. Jakékoli další změny, které uživatelská procedura provede ve struktuře, se ignorují; neprojeví se zpět do aplikace.

Pokud uživatelská procedura vrátí hodnotu XROK v poli *DXRES* struktury MQDXP, ale nezmění pole *MDENC* nebo *MDCSI* v deskriptoru zprávy, vrátí správce front pro tato pole hodnoty, které měla odpovídající pole ve struktuře MQDXP na vstupu do uživatelské procedury.

INLEN (10místné celé číslo se znaménkem)-vstup

Délka v bajtech *INBUF*.

Jedná se o délku vstupní vyrovnávací paměti *INBUF*a určuje počet bajtů, které má uživatelská procedura zpracovat. *INLEN* je menší z hodnot délky dat zprávy před převodem a délky vyrovnávací paměti poskytované aplikací při volání MQGET.

Hodnota je vždy větší než nula.

INBUF (1bajtový bitový řetězec x INLEN)-vstup

Vyrovnávací paměť obsahující nepřevedenou zprávu.

Obsahuje data zprávy před převodem. Pokud uživatelská procedura nemůže převést data, vrátí po dokončení uživatelské procedury aplikaci obsah této vyrovnávací paměti.

Poznámka: Ukončení by nemělo měnit *INBUF* ; pokud je tento parametr změněn, výsledky nejsou definovány.

OUTLEN (10místné celé číslo se znaménkem)-vstup

Délka v bajtech *OUTBUF*.

Jedná se o délku výstupní vyrovnávací paměti *OUTBUF*a je shodná s délkou vyrovnávací paměti poskytované aplikací při volání MQGET.

Hodnota je vždy větší než nula.

OUTBUF (1bajtový bitový řetězec x OUTLEN)-výstup

Vyrovnávací paměť obsahující převedené zprávy.

Pokud byl při výstupu z ukončení převod úspěšný (jak uvádí hodnota XROK v poli *DXRES* parametru **MQDXP**), **OUTBUF** obsahuje data zprávy, která mají být doručena aplikaci, v požadovaném znázornění. Pokud byl převod neúspěšný, všechny změny, které uživatelská procedura provedla v této vyrovnávací paměti, jsou ignorovány.

Vyvolání RPG (ILE)

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                   CALLP     exitname(MQDXP : MQMD : INLEN :
C                                      INBUF : OUTLEN : OUTBUF)
```

Definice prototypu pro volání je:

```
D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
Dexitname         PR                  EXTPROC('exitname')
D* Data-conversion exit parameter block
D MQDXP                          44A
D* Message descriptor
D MQMD                          364A
D* Length in bytes of INBUF
D INLEN                          10I 0 VALUE
D* Buffer containing the unconverted message
D INBUF                           *   VALUE
D* Length in bytes of OUTBUF
D OUTLEN                         10I 0 VALUE
D* Buffer containing the converted message
D OUTBUF                          *   VALUE
```

Ukončení programovacího rozhraní citlivého na produkt

Uživatelské procedury, uživatelské procedury rozhraní API a odkaz na instalovatelné služby

Informace v této části vám pomohou při vývoji uživatelských procedur, uživatelských procedur rozhraní API a instalovatelných aplikací služeb:

Související pojmy

Uživatelské procedury, uživatelské procedury rozhraní API a instalovatelné služby IBM MQ

Související úlohy

Rozšíření prostředků správce front

Struktura MQIEP

Struktura MQIEP obsahuje vstupní bod pro každé volání funkce, které má povoleno provádět uživatelské procedury.

Pole

StrucId

Typ: MQCHAR4 -vstup

Identifikátor struktury. Hodnota je následující:

MQIEP_STRUC_ID

Verze

Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQIEP_VERSION_1
Číslo verze struktury verze 1.

MQIEP_CURRENT_VERSION
Aktuální verze struktury.

StrucLength

Typ: MQLONG

Velikost struktury MQIEP v bajtech. Hodnota je následující:

MQIEP_LENGTH_1

Příznaky

Typ: MQLONG

Poskytuje informace o adresách funkce. Příznak, který označuje, zda je knihovna v podprocesu, lze použít s příznakem, který označuje, zda je knihovna knihovnou klienta nebo serveru.

Následující hodnota se používá pro neuvedení informací o knihovně:

MQIEPF_NONE

Jedna z následujících hodnot se používá k určení, zda je sdílená knihovna podprocesová nebo nepodprocesová:

MQIEPF_NON_THREADED_LIBRARY
Sdílená knihovna bez podprocesů

MQIEPF_THREADED_LIBRARY
Sdílená knihovna s podporou podprocesů

Jedna z následujících hodnot se používá k určení, zda je sdílená knihovna klientem nebo sdílenou knihovnou serveru:

MQIEPF_CLIENT_LIBRARY
Klientská sdílená knihovna

MQIEPF_LOCAL_LIBRARY
Sdílená knihovna serveru

Vyhrazeno
Typ: MQPTR

Volání MQBACK_Call
Typ: PMQ_BACK_CALL

Adresa volání MQBACK.

Volání MQBEGIN_Call
Typ: PMQ_BEGIN_CALL

Adresa volání MQBEGIN.

Volání MQBUFMH_Call
Typ: PMQ_BUFMH_CALL

Adresa volání MQBUFMH.

Volání MQCB_Call
Typ: PMQ_CB_CALL

Adresa volání MQCB.

Volání MQCLOSE_Call
Typ: PMQ_CLOSE_CALL

Adresa volání MQCLOSE.

Volání MQCMIT_Call
Typ: PMQ_CMIT_CALL

Adresa volání MQCMIT.

Volání MQCONN_Call
Typ: PMQ_CONN_CALL

Adresa volání MQCONN.

Volání_MQCONNX_Call
Typ: PMQ_CONNX_CALL

Adresa volání MQCONNX.

Volání MQCRTMH_Call
Typ: PMQ_CRTMH_CALL

Adresa volání MQCRTMH.

Volání MQCTL_Call
Typ: PMQ_CTL_CALL

Adresa volání MQCTL.

Volání MQDISC_Call
Typ: PMQ_DISC_CALL

Adresa volání MQDISC.

Volání MQDLTMH_Call
Typ: PMQ_DLTMH_CALL

Adresa volání MQDLTMH.

Volání MQDLTMP_Call
Typ: PMQ_DLTMP_CALL

Adresa volání MQDLTMP.

Volání MQGET_Call
Typ: PMQ_GET_CALL

Adresa volání MQGET.

Volání MQINQ_Call
Typ: PMQ_INQ_CALL

Adresa volání MQINQ.

Volání MQINQMP_Call
Typ: PMQ_INQMP_CALL

Adresa volání MQINQMP.

Volání MQMHBUF_Call
Typ: PMQ_MHBUF_CALL

Adresa volání MQMHBUF.

Volání MQOPEN_Call
Typ: PMQ_OPEN_CALL

Adresa volání MQOPEN.

Volání MQPUT_Call
Typ: PMQ_PUT_CALL

Adresa volání MQPUT.

MQPUT1_Call
Typ: PMQ_PUT1_CALL

Adresa volání MQPUT1 .

Volání MQSET_Call
Typ: PMQ_SET_CALL

Adresa volání MQSET.

Volání MQSETMP_Call
Typ: PMQ_SETMP_CALL

Adresa volání MQSETMP.

Volání MQSTAT_Call
Typ: PMQ_STAT_CALL

Adresa volání MQSTAT.

Volání MQSUB_Call
Typ: PMQ_SUB_CALL

Adresa volání MQSUB.

Volání MQSUBRQ_Call
Typ: PMQ_SUBRQ_CALL

Adresa volání MQSUBRQ.

Volání MQXCNVC_Call
Typ: PMQ_XCNVC_CALL

Adresa volání MQXCNVC.

Volání MQXCLWLN_Call
Typ: PMQ_XCLWLN_CALL

Adresa volání MQXCLWLN.

Volání MQXDX_Call
Typ: PMQ_XDX_CALL

Adresa volání MQXDX.

Volání MQXEP_Call
Typ: PMQ_XEP_CALL

Adresa volání MQXEP.

Volání MQZEP_Call
Typ: volání PMQ_ZEP_CALL

Adresa volání MQZEP.

C Prohlášení

```
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 */
};
```

Odkaz na uživatelskou proceduru převodu dat

Pro systém z/OSmusíte zapsat uživatelské procedury pro převod dat v jazyku assembleru. Pro ostatní platformy se doporučuje používat programovací jazyk C.

Pro usnadnění vytvoření uživatelského programu pro převod dat jsou dodány následující prostředky:

- Zdrojový soubor kostry
- Volání konvertovat znaky

- Obslužný program, který vytváří fragment kódu, který provádí převod dat ve strukturách datových typů. Tento obslužný program přijímá pouze vstup jazyka C. V systému z/OSvytváří kód assembleru.

Postup při psaní programů viz:

- IBM i Zápis uživatelského programu pro převod dat pro IBM MQ for IBM i
- z/OS Zápis uživatelského programu pro převod dat pro IBM MQ for z/OS
- Zápis uživatelské procedury pro převod dat pro systémy IBM MQ for AIX or Linux
- Zápis uživatelské procedury pro převod dat pro IBM MQ for Windows

Zdrojový soubor kostry

Ty mohou být použity jako výchozí bod při psaní uživatelského programu pro převod dat.

Dodané soubory jsou uvedeny v seznamu Tabulka 816 na stránce 1452.

Tabulka 816. Zdrojové soubory kostry

Platforma	Soubor
AIX AIX	amqsvfc0.c
IBM i IBM i	QMQMSAMP/QCSRC (AMQSVFC4)
Linux Linux	amqsvfc0.c
Windows Windows systémy	amqsvfc0.c
z/OS z/OS	CSQ4BAX8 (“1” na stránce 1452) CSQ4BAX9 (“2” na stránce 1452) CSQ4CAX9 (“3” na stránce 1452)

Notes:

1. Ilustruje volání MQXCVNC.
2. Modul wrapper pro fragmenty kódu generované obslužným programem pro použití ve všech prostředích s výjimkou prostředí CICS.
3. Modul wrapper pro fragmenty kódu generované obslužným programem pro použití v prostředí CICS .

Volání převodu znaků

Pomocí volání MQXCNVC (převod znaků) z uživatelského programu pro převod dat převeďte znaková data zprávy z jedné znakové sady do jiné. Pro určité vícebajtové znakové sady (například znakové sady UTF-16) musí být použity příslušné volby.

Z uživatelské procedury nelze provést žádná jiná volání MQI; pokus o provedení takového volání selže s kódem příčiny MQRC_CALL_IN_PROGRESS.

Další informace o volání MQXCNVC a příslušných volbách naleznete v části “MQXCNVC-Převod znaků” na stránce 917 .

Obslužný program pro vytvoření kódu ukončení převodu

Pomocí těchto informací získáte další informace o vytváření kódu ukončení převodu.

Příkazy pro vytvoření kódu uživatelské procedury převodu jsou:

IBM i
CVTMQMDTA (Převést datový typ IBM MQ)

Systémy AIX, Linux, and Windows
`crtmqcvx` (Vytvořit IBM MQ převod-ukončit)

z/OS
CSQUCVX

Příkaz pro vaši platformu vytváří fragment kódu, který provádí převod dat na strukturách datových typů pro použití ve vašem uživatelském programu pro převod dat. Příkaz vezme soubor obsahující jednu nebo více definic struktury jazyka C. V systému z/OSpak vygeneruje datovou sadu obsahující fragmenty kódu sestavovacího modulu a převodní funkce. Na jiných platformách generuje soubor s funkcí C pro převod každé definice struktury. V systému z/OSvyžaduje obslužný program přístup k běhové knihovně LE/370 SCEERUN.

Vyvolání obslužného programu CSQUCVX v systému z/OS

Obrázek 10 na stránce 1453 ukazuje příklad JCL použitého k vyvolání obslužného programu CSQUCVX.

```
//CVX    EXEC PGM=CSQUCVX
//STEPLIB  DD DISP=SHR,DSN=thlqual.SCSQANLE
//         DD DISP=SHR,DSN=thlqual.SCSQLOAD
//         DD DISP=SHR,DSN=le370qual.SCEERUN
//SYSPRINT DD SYSOUT=*
//CSQUINP  DD DISP=SHR,DSN=MY.MQSERIES.FORMATS(MSG1)
//CSQUOUT  DD DISP=OLD,DSN=MY.MQSERIES.EXITS(MSG1)
```

Obrázek 10. Ukázkový soubor JCL použitý k vyvolání obslužného programu CSQUCVX

Příkazy definice dat z/OS

Obslužný program CSQUCVX vyžaduje příkazy DD s následujícími názvy definic dat uvedenými v souboru Tabulka 817 na stránce 1453:

Tabulka 817. Názvy a popisy příkazů definice dat

Příkaz DD	Popis
SYSPRINT	Uvádí datovou sadu nebo třídu tiskového souboru pro souběžný tisk pro sestavy a chybové zprávy.
CSQUINP	Uvádí rozdělenou datovou sadu obsahující definice datových struktur, které se mají převést.
CSQUOUT	Určuje rozdělenou datovou sadu, do které mají být zapsány fragmenty převodního kódu. Délka logického záznamu (LRECL) musí být 80 a formát záznamu (RECFM) musí být FB.

Chybové zprávy v systémech AIX, Linux, and Windows

Příkaz `crtmqcvx` vrátí zprávy v rozsahu AMQ7953 až AMQ7970.

Tyto zprávy jsou uvedeny v části Zprávy a kódy příčiny *IBM MQ Zprávy*.

Existují dva hlavní typy chyb:

- Hlavní chyby, jako např. chyby syntaxe, když zpracování nemůže pokračovat.

Na obrazovce se zobrazí zpráva s číslem řádku chyby ve vstupním souboru. Výstupní soubor mohl být částečně vytvořen.

- Další chyby, když se zobrazí zpráva oznamující, že byl nalezen problém, ale že syntaktická analýza struktury může pokračovat.

  Výstupní soubor byl vytvořen a obsahuje informace o chybě týkající se problémů, které se vyskytly. Tyto informace o chybě mají předponu `#error` , takže vytvořený kód není akceptován žádným kompilátorem bez zásahu do nápravy problémů.

Platná syntaxe

Váš vstupní soubor pro obslužný program musí odpovídat syntaxi jazyka C.

Pokud nejste obeznámeni s C, podívejte se na příklad C v tomto tématu.

Kromě toho mějte na paměti následující pravidla:

- typedef je rozpoznán pouze před klíčovým slovem struct.
- V deklaracích struktury je vyžadována značka struktury.
- Můžete použít prázdné hranaté závorky [] k označení pole proměnné délky nebo řetězce na konci zprávy.
- Vícerozměrná pole a pole řetězců nejsou podporována.
- Jsou rozpoznány následující další datové typy:
  - MQBOOL
  - MQBYTE
  - MQCHAR
  - MQFLOAT32
  - MQFLOAT64
  - MQSHORT
  - MQLONG
  - MQINT8
  - MQUINT8
  - MQINT16
  - MQUINT16
  - MQINT32
  - MQUINT32
  - MQINT64
  - MQUINT64

  Pole MQCHAR jsou převedená kódová stránka, ale MQBYTE, MQINT8 a MQUINT8 nejsou dotčeny. Pokud se kódování liší, jsou odpovídajícím způsobem převedeny hodnoty MQSHORT, MQLONG, MQINT16, MQUINT16, MQINT32, MQUINT32, MQINT64, MQUINT64, MQFLOAT32, MQFLOAT64 a MQBOOL.
- Nepoužívejte následující typy dat:
  - dvojitý
  - ukazatele
  - bitová pole

  Důvodem je skutečnost, že obslužný program pro vytváření kódu uživatelské procedury převodu neposkytuje prostředek pro převod těchto datových typů. Chcete-li to překonat, můžete napsat své vlastní rutiny a volat je z východu.

Další body k poznámce:

- Ve vstupní datové sadě nepoužívejte pořadová čísla.
- Pokud existují pole, pro která chcete poskytnout vlastní převodní rutiny, deklarujte je jako MQBYTE a poté nahraďte generovaná makra CMQXCFBA vlastním převodním kódem.

Příklad C

```
struct TEST { MQLONG    SERIAL_NUMBER;
              MQCHAR    ID[5];
              MQINT16   VERSION;
              MQBYTE    CODE[4];
              MQLONG    DIMENSIONS[3];
              MQCHAR    NAME[24];
            } ;
```

To odpovídá následujícím deklaracím v jiných programovacích jazycích:

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

Podporováno pouze v systému z/OS

```
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-uživatelská procedura publikování

Volání MQ_PUBLISH_EXIT může zkontrolovat a změnit zprávy doručené odběratelům.

Účel

Použijte uživatelskou proceduru publikování ke kontrole a změně zpráv doručených odběratelům:

- Zkontrolovat obsah zprávy publikované pro každého odběratele
- Upravit obsah zprávy publikované každému odběrateli
- Změnit frontu, do které je zpráva vložena

- Zastavit doručování zprávy odběrateli

Tato uživatelská procedura není v systému IBM MQ for z/OSk dispozici.

Syntaxe

MQ_PUBLISH_EXIT (*ExitParms, PubContext, SubContext*)

Parametry

***ExitParms* (MQPSXP)- Input/Output**
ExitParms obsahuje informace o vyvolání uživatelské procedury.

***PubContext* (MQPBC) - Input**
PubContext obsahuje kontextové informace o vydavateli publikace.

***SubContext* (MQSBC) - Input/Output**
SubContext obsahuje kontextové informace o odběrateli přijímajícím publikování.

MQPSXP-Datová struktura uživatelské procedury publikování

Struktura MQPSXP popisuje informace, které jsou předány uživatelské proceduře publikování a vráceny z ní.

Tabulka 818 na stránce 1456 shrnuje pole ve struktuře:

Tabulka 818. Pole v MQPSXP

Pole	Popis
StrucID	Identifikátor struktury
Version	Číslo verze struktury
ExitId	Typ volané uživatelské procedury
ExitReason	Důvod volání uživatelské procedury
ExitResponse	Odpověď od ukončení
ExitResponse2	Sekundární odezva z ukončení
Feedback	Kód zpětné vazby
ExitUserArea	Ukončit uživatelskou oblast
ExitData	Data uživatelské procedury
QMgrName	Název lokálního správce front
Hconn	Manipulátor připojení
MsgDescPtr	Adresa deskriptoru zprávy (MQMD)
MsgHandle	Manipulovat s vlastnostmi zprávy (MQHMSG)
MsgInPtr	Adresa vstupní zprávy
MsgInLength	Délka vstupní zprávy
MsgOutPtr	Adresa výstupní zprávy
MsgOutLength	Délka výstupní zprávy
pEntryPoints	Adresa struktury MQIEP

Pole

***StrucID* (MQCHAR4)**
StrucID je identifikátor struktury. Hodnota je následující:

MQPSXP_STRUCID
MQPSXP_STRUCID je identifikátor pro strukturu parametru uživatelské procedury publikování. Pro programovací jazyk C je také definována konstanta MQPSXP_STRUC_ID_ARRAY ; má stejnou hodnotu jako MQPSXP_STRUC_ID, ale je to pole znaků místo řetězce.

StrucID je vstupní pole pro ukončení.

***Version* (MQLONG)**
Version je číslo verze struktury. Hodnota je následující:

MQPSXP_VERSION_1
MQPSXP_VERSION_1 je struktura parametru uživatelské procedury publikování verze 1. Konstanta MQPSXP_CURRENT_VERSION je také definována se stejnou hodnotou.

Version je vstupní pole pro ukončení.

***ExitId* (MQLONG)**
ExitId je typ volané uživatelské procedury. Hodnota je následující:

MQXT_PUBLISH_EXIT
Uživatelská procedura publikování.

ExitId je vstupní pole pro ukončení.

***ExitReason* (MQLONG)**
ExitReason je příčinou volání uživatelské procedury. Možné hodnoty jsou:

MQXR_INIT
Uživatelská procedura pro toto připojení je volána pro inicializaci. Uživatelská procedura může získat a inicializovat prostředky, které potřebuje; například hlavní paměť.

MQXR_TERM
Uživatelská procedura pro toto připojení je volána, protože uživatelská procedura bude zastavena. Uživatelská procedura musí uvolnit všechny prostředky, které získala od své inicializace; například hlavní paměť.

MQXR_PUBLICATION
Uživatelská procedura je volána správcem front před vložením publikování do fronty zpráv odběratele. Uživatelská procedura může změnit zprávu, nevložit zprávu do fronty nebo zastavit publikování.

ExitReason je vstupní pole pro ukončení.

***ExitResponse* (MQLONG)**
Nastavte *ExitResponse* v uživatelské proceduře, abyste určili, jak musí zpracování pokračovat. *ExitResponse* je jedna z následujících hodnot:

MQXCC_OK
Nastavte MQXCC_OK , aby se zpracování pokračovalo normálně. Nastavte MQXCC_OK jako odezvu na jakékoli hodnoty ExitReason.

Pokud má ExitReason hodnotu MQXR_PUBLICATION, pole *DestinationQName* a *DestinationQMgrName* struktury MQSBC identifikují místo určení, kam se zpráva odešle.

MQXCC_FAILED
Nastavte MQXCC_FAILED , chcete-li zastavit operaci publikování. Kód dokončení MQCC_FAILED a kód příčiny 2557 (09FD) (RC2557): MQRC_PUBLISH_EXIT_ERROR je nastaven při návratu z uživatelské procedury.

MQXCC_SUPPRESS_FUNCTION
Nastavte MQXCC_SUPPRESS_FUNCTION , chcete-li zastavit normální zpracování zprávy. Hodnotu MQXCC_SUPPRESS_FUNCTION nastavte pouze v případě, že *ExitReason* má hodnotu MQXR_PUBLICATION.

Zpráva je nadále zpracovávána správcem front v souladu s volbou MQRO_DISCARD_MSG v poli *Report* v deskriptoru zprávy.

- Je-li zadána volba MQRO_DISCARD_MSG , nebude zpráva doručena odběrateli.
- Není-li zadána volba MQRO_DISCARD_MSG , bude zpráva umístěna do fronty nedoručených zpráv. Pokud neexistuje žádná fronta nedoručených zpráv nebo zprávu nelze úspěšně umístit do fronty nedoručených zpráv, publikování nebude doručeno odběrateli. Doručení publikování ostatním odběratelům závisí na hodnotách atributů objektu tématu PMSGDLV a NPMSGDLV . Vysvětlení těchto atributů naleznete v popisech parametrů příkazu DEFINE TOPIC .

ExitResponse je výstupní pole z ukončení.

***ExitResponse2* (MQLONG)**
ExitResponse2 je vyhrazeno pro budoucí použití.

***Feedback* (MQLONG)**
Feedback je kód zpětné vazby, který má být použit v případě, že uživatelská procedura vrátí hodnotu MQXCC_SUPPRESS_FUNCTION v souboru *ExitResponse*.

Při vstupu do uživatelské procedury má parametr *Feedback* vždy hodnotu MQFB_NONE. Pokud uživatelská procedura vrátí hodnotu MQXCC_SUPPRESS_FUNCTION, nastavte *Feedback* na hodnotu, která se má použít pro zprávu, když ji správce front umístí do fronty nedoručených zpráv. Pokud má správce front při návratu z uživatelské procedury původní hodnotu *Feedback* MQFB_NONE, nastaví hodnotu *Feedback* na MQFB_STOPPED_BY_PUBSUB_EXIT.

Feedback je vstupní/výstupní pole pro ukončení.

***ExitUserArea* (MQBYTE16)**
ExitUserArea je pole, které je k dispozici pro použití uživatelskou procedurou. Každé připojení má samostatný *ExitUserArea*. Délka řetězce *ExitUserArea* je dána MQ_EXIT_USER_AREA_LENGTH.

Pole *ExitReason* má hodnotu MQXR_INIT při prvním vyvolání uživatelské procedury. *ExitUserArea* se inicializuje na MQXUA_NONE při prvním vyvolání uživatelské procedury pro připojení. Následné změny v souboru *ExitUserArea* jsou zachovány ve všech vyvoláních uživatelské procedury.

ExitUserArea je vstupní/výstupní pole pro ukončení.

***ExitData* (MQCHAR32)**
ExitData jsou data pevné uživatelské procedury definovaná parametrem **PublishExitData** sekce v inicializačním souboru správce front. Data jsou vyplněna mezerami na celou délku pole. Pokud v inicializačním souboru nejsou definována žádná data pevné uživatelské procedury, je hodnota *ExitData* prázdná. Délka řetězce *ExitData* je dána MQ_EXIT_DATA_LENGTH.

ExitData je vstupní pole pro ukončení.

***QMgrName* (MQCHAR48)**
QMgrName je název lokálního správce front. Název je vyplněn mezerami na celou délku pole. Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

QMgrName je vstupní pole pro ukončení.

***Hconn* (MQHCONN)**
Hconn je manipulátor představující připojení ke správci front. Použijte *Hconn* pouze jako parametr pro volání funkce vlastností zprávy MQSETMP, MQINQMMPnebo MQDLTMP pro práci s vlastnostmi zprávy.

Hconn je vstupní pole pro ukončení.

***MsgDescPtr* (PMQMD)**
MsgDescPtr je adresa deskriptoru zprávy (MQMD) zpracovávané zprávy a je kopií deskriptoru MQMD vráceného z volání MQPUT. Uživatelská procedura může změnit obsah deskriptoru zprávy. Jakákoli změna obsahu deskriptoru zprávy musí být provedena s opatrností. Zejména v případě, že pole *SubType* struktury MQSBC má hodnotu MQSUBTYPE_PROXY, nesmí být pole *CorrelId* v deskriptoru zprávy změněno.

Pokud má parametr *ExitReason* hodnotu MQXR_INIT nebo MQXR_TERM , nepředává se uživatelské proceduře žádný deskriptor zprávy; v těchto případech je *MsgDescPtr* ukazatel Null.

MsgDescPtr je vstupní pole pro ukončení.

***MsgHandle* (MQHMSG)**
MsgHandle je popisovač pro vlastnosti zprávy. Chcete-li pracovat s vlastnostmi zprávy, použijte funkci *MsgHandle* pouze s voláními funkce vlastností zprávy MQSETMP, MQINQMMPnebo MQDLTMP .

MsgHandle je vstupní pole pro ukončení.

***MsgInPtr* (PMQVOID)**
MsgInPtr je adresa dat vstupní zprávy. Obsah vyrovnávací paměti adresované pomocí *MsgInPtr* lze upravit pomocí uživatelské procedury; viz *MsgOutPtr* .

MsgInPtr je vstupní pole pro ukončení.

***MsgInLength* (MQLONG)**
MsgInLength je délka dat zprávy předaných uživatelské proceduře v bajtech. Adresa dat je dána *MsgInPtr*.

MsgInLength je vstupní pole pro ukončení.

***MsgOutPtr* (PMQVOID)**
MsgOutPtr je adresa vyrovnávací paměti obsahující data zprávy vrácená z uživatelské procedury. Při vstupu do uživatelské procedury má parametr *MsgOutPtr* hodnotu null. Pokud je při návratu z uživatelské procedury hodnota stále null, správce front odešle zprávu určenou parametrem *MsgInPtr*s délkou určenou parametrem *MsgInLength*.

Pokud uživatelská procedura upraví data zprávy, použijte jednu z následujících procedur:

- Pokud se délka dat nezmění, lze data upravit ve vyrovnávací paměti adresované pomocí *MsgInPtr*. V tomto případě neměňte *MsgOutPtr* a *MsgOutLength*.
- Pokud jsou upravená data kratší než původní data, lze je upravit ve vyrovnávací paměti adresované produktem *MsgInPtr*. V tomto případě musí být parametr *MsgOutPtr* nastaven na adresu vstupní vyrovnávací paměti zpráv a parametr *MsgOutLength* musí být nastaven na novou délku dat zprávy.
- Pokud jsou upravená data delší nebo mohou být delší než původní data, uživatelská procedura musí získat novou vyrovnávací paměť zpráv. Zkopírujte do něj upravená data. Nastavte *MsgOutPtr* na adresu nové vyrovnávací paměti a nastavte *MsgOutLength* na délku nových dat zprávy. Uživatelská procedura je zodpovědná za uvolnění vyrovnávací paměti adresované *MsgOutPtr* při dalším volání uživatelské procedury.

Poznámka: *MsgOutPtr* je vždy ukazatel Null na vstupu do uživatelské procedury, nikoli adresa dříve získané vyrovnávací paměti zpráv. Chcete-li uvolnit dříve získanou vyrovnávací paměť, musí uživatelská procedura uložit svou adresu a délku. Uložte informace buď do adresáře *ExitUserArea*, nebo do řídicího bloku, který má svou adresu uloženou v adresáři *ExitUserArea*.

MsgOutPtr je vstupní/výstupní pole pro ukončení.

***MsgOutLength* (MQLONG)**
MsgOutLength je délka dat zprávy vrácených uživatelskou procedurou v bajtech. Při vstupu do uživatelské procedury je toto pole vždy nulové. Při návratu z uživatelské procedury je toto pole ignorováno, pokud má *MsgOutPtr* hodnotu null. Informace o úpravě dat zprávy viz *MsgOutPtr* .

MsgOutLength je vstupní/výstupní pole pro ukončení.

***pEntryPoints* (PMQIEP)**
pEntryPoints je adresa struktury MQIEP, jejímž prostřednictvím lze provádět volání MQI a DCI.

Deklarace jazyka 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-Datová struktura kontextu publikování

Struktura MQPBC obsahuje kontextové informace související s vydavatelem publikování, které jsou předány uživatelské proceduře publikování.

Tabulka 819 na stránce 1460 shrnuje pole ve struktuře:

Tabulka 819. Pole v MQPBC

Pole	Popis
StrucID	Identifikátor struktury
Version	Číslo verze struktury
PubTopicString	Publikovat řetězec tématu
MsgDescPtr	Adresa deskriptoru zprávy (MQMD)

Pole

***StrucID* (MQCHAR4)**
StrucID je identifikátor struktury. Hodnota je následující:

MQPBC_STRUCID
MQPBC_STRUCID je identifikátor pro strukturu kontextu publikování. Pro programovací jazyk C je také definována konstanta MQPBC_STRUC_ID_ARRAY ; má stejnou hodnotu jako MQPBC_STRUC_ID, ale je to pole znaků místo řetězce.

StrucID je vstupní pole pro ukončení.

***Version* (MQLONG)**
Version je číslo verze struktury. Hodnota je následující:

MQPBC_VERSION_1
MQPBC_VERSION_1 je struktura parametru uživatelské procedury publikování verze 1.

MQPBC_VERSION_2
MQPBC_VERSION_2 je struktura parametrů uživatelské procedury publikování verze 2. Konstanta MQPBC_CURRENT_VERSION je také definována se stejnou hodnotou.

Version je vstupní pole pro ukončení.

***PubTopicString* (MQCHARV)**
PubTopicString je řetězec tématu, do který se publikuje.

PubTopicString je vstupní pole pro ukončení.

***MsgDescPtr* (PMQMD)**
MsgDescPtr je adresa kopie deskriptoru zprávy (MQMD) pro zpracovávanou zprávu.

MsgDescPtr je vstupní pole pro ukončení.

Deklarace jazyka 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-Datová struktura kontextu odběru

Struktura MQSBC obsahuje kontextové informace týkající se odběratele, který přijímá publikování, které je předáno uživatelské proceduře publikování.

Tabulka 820 na stránce 1461 shrnuje pole ve struktuře:

Tabulka 820. Pole v souboru MQSBC	
Pole	**Popis**
StrucID	Identifikátor struktury
Version	Číslo verze struktury
DestinationQMgrName	Název cílového správce front
DestinationQName	Název cílové fronty
SubType	Typ odběru
SubOptions	Volby odběru
ObjectName	Název objektu
ObjectString	Řetězec objektu
SubTopicString	Řetězec tématu odběru
SubName	Název odběru
SubId	Identifikátor odběru
SelectionString	Adresa řetězce výběru
SubLevel	Úroveň odběru
PSProperties	Vlastnosti publikování/odběru

Pole

***StrucID* (MQCHAR4)**
Identifikátor struktury. Hodnota je následující:

MQSBC_STRUCID
MQSBC_STRUCID je identifikátor pro strukturu parametru uživatelské procedury publikování. Pro programovací jazyk C je definována také konstanta MQSBC_STRUC_ID_ARRAY ; MQSBC_STRUC_ID_ARRAY má stejnou hodnotu jako MQSBC_STRUC_ID, ale je to pole znaků místo řetězce.

StrucID je vstupní pole pro ukončení.

***Version* (MQLONG)**
Číslo verze struktury. Hodnota je následující:

MQSBC_VERSION_1
Struktura parametrů uživatelské procedury publikování verze 1. Konstanta MQSBC_CURRENT_VERSION je také definována se stejnou hodnotou.

Version je vstupní pole pro ukončení.

***DestinationQMgrName* (MQCHAR48)**
DestinationQMgrName je název správce front, do kterého se zpráva odesílá. Název je vyplněn mezerami na celou délku pole. Název může být změněn uživatelskou procedurou. Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

DestinationQMgrName je vstupní/výstupní pole pro ukončení; viz poznámka.

***DestinationQName* (MQCHAR48)**
DestinationQName je název fronty, do které se zpráva odesílá. Název je vyplněn mezerami na celou délku pole. Název může být změněn uživatelskou procedurou. Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH.

DestinationQName je vstupní/výstupní pole pro ukončení; viz poznámka.

***SubType* (MQLONG)**
SubType označuje, jak byl odběr vytvořen. Platné hodnoty jsou MQSUBTYPE_API, MQSUBTYPE_ADMIN a MQSUBTYPE_PROXY ; viz Stav dotazového odběru (odezva).

SubType je vstupní pole pro ukončení.

***SubOptions* (MQLONG)**
SubOptions jsou volby odběru; popis hodnot, které může toto pole obsahovat, viz “Volby (MQLONG) pro MQSD” na stránce 574 .

SubOptions je vstupní pole pro ukončení.

***ObjectName* (MQCHAR48)**
ObjectName je název objektu tématu, jak je definován v lokálním správci front. Délka tohoto pole je dána hodnotou MQ_TOPIC_NAME_LENGTH. Název objektu je název objektu administrativního tématu, který správce front přidružil k řetězci tématu. I v případě, že odběratel poskytl objekt tématu jako součást odběru, může být objekt *ObjectName* jiným objektem tématu. Přidružení objektu tématu k odběru závisí na úplném vyřešení problému *SubTopicString*.

ObjectName je vstupní pole pro ukončení.

***ObjectString* (MQCHARV)**
ObjectString je úplný řetězec tématu publikace, která byla přihlášena k odběru. Všechny zástupné znaky v původním řetězci odběru jsou vyřešeny. Liší se od pole MQSD odběr *ObjectString* popsaného v části “ObjectString (MQCHARV) pro MQSD” na stránce 584, které může obsahovat zástupné znaky, a neobsahuje žádný název objektu poskytnutý odběratelem.

ObjectString je vstupní pole pro ukončení.

***SubTopicString* (MQCHARV)**
SubTopicString je úplný řetězec tématu dodaný odběratelem. *SubTopicString* je kombinace řetězce tématu definovaného v objektu tématu a řetězce tématu. Odběratel musí poskytnout buď objekt tématu, řetězec tématu, nebo obojí. Pokud odběratel poskytne řetězec tématu, může obsahovat zástupné znaky.

SubTopicString je vstupní pole pro ukončení.

***SubName* (MQCHARV)**
SubName je název odběru poskytnutý odběratelem nebo generovaný název.

SubName je vstupní pole pro ukončení.

***SubId* (MQBYTE 24)**
SubId je jedinečný interní identifikátor odběru.

SubId je vstupní pole pro ukončení.

***SelectionString* (MQCHARV)**
SelectionString je výběrové kritérium používané při přihlašování k odběru zpráv z tématu; viz Selektory.

SelectionString je vstupní pole pro ukončení.

***SubLevel* (MQLONG)**
SubLevel je úroveň zachycení přidružená k odběru; další podrobnosti viz “SubLevel (MQLONG) pro MQSD” na stránce 587 .

SubLevel je vstupní pole pro ukončení.

***PSProperties* (MQLONG)**
PSProperties jsou vlastnosti publikování/odběru. Určují způsob přidávání vlastností zpráv souvisejících s publikování/odběrem do zpráv odesílaných v rámci tohoto odběru. Možné hodnoty jsou MQPSPROP_NONE, MQPSPROP_COMPAT, MQPSPROP_RFH2, MQPSPROP_MSGPROP. Popis těchto hodnot viz Volitelné parametry (změna, kopírování a vytvoření odběru) .

PSProperties je vstupní pole pro ukončení.

Poznámka: Kontroly autorizace se provádějí pouze na původních hodnotách *DestinationQMgrName* a *DestinationQName* před jejich předáním do uživatelské procedury publikování. Když uživatelská procedura změní cílovou frontu, neprovedou se žádné nové kontroly autorizace, a to buď změnou *DestinationQMgrName* , nebo *DestinationQName*.

Deklarace jazyka 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;
```

Volání uživatelské procedury kanálu a datové struktury

Tato kolekce témat poskytuje referenční informace o speciálních voláních IBM MQ a datových strukturách, které můžete použít při psaní programů uživatelské procedury kanálu.

Jedná se o informace o programovacím rozhraní, které jsou citlivé na produkt. Uživatelské procedury IBM MQ můžete psát v následujících programovacích jazycích:

Tabulka 821. Uživatelské procedury IBM MQ : platformy a programovací jazyky

Platforma	**Programovací jazyky**
IBM MQ for z/OS	Assembler a C (které musí odpovídat programovacímu prostředí systému C pro systémové uživatelské procedury popsané v příručce *z/OS C/C++ Programming Guide*.)
IBM MQ for IBM i	ILE C, ILE COBOL a ILE RPG
Všechny ostatní platformy IBM MQ	C

Můžete také zapsat uživatelské procedury v produktu Java pro použití pouze s aplikacemi Java a JMS . Další informace o vytváření a používání uživatelských procedur kanálu s produktem IBM MQ classes for Javanaleznete v tématu Použití uživatelských procedur kanálu v produktu IBM MQ classes for Java a v tématu IBM MQ classes for JMSv tématu Použití uživatelských procedur kanálu s produktem IBM MQ classes for JMS.

Nemůžete psát uživatelské procedury IBM MQ v TAL nebo Visual Basic. Deklarace pro strukturu MQCD je však poskytnuta v jazyku Visual Basic pro použití ve volání MQCONNX z programu IBM MQ MQI client .

V řadě případů v popisech, které následují, jsou parametry pole nebo znakové řetězce s velikostí, která není pevná. Pro tyto parametry se k reprezentaci číselné konstanty používají malá písmena "n" . Je-li deklarace pro tento parametr kódována, musí být hodnota "n" nahrazena požadovanou číselnou hodnotou. Další informace o konvencích použitých v těchto popisech viz “Základní datové typy” na stránce 235.

Soubory definice dat

Soubory definice dat jsou dodávány s produktem IBM MQ pro každý z podporovaných programovacích jazyků. Podrobnosti o těchto souborech viz Kopírovat, záhlaví, zahrnout a soubory modulu.

MQ_CHANNEL_EXIT-Ukončit kanál

Volání MQ_CHANNEL_EXIT popisuje parametry, které jsou předány jednotlivým kanálovým ukončením volaným agentem kanálu zpráv.

Správce front neposkytl žádný vstupní bod s názvem MQ_CHANNEL_EXIT. Název MQ_CHANNEL_EXIT nemá žádný zvláštní význam, protože názvy uživatelských procedur kanálu jsou uvedeny v definici kanálu MQCD.

Existuje pět typů uživatelské procedury kanálu:

- Uživatelská procedura zabezpečení kanálu
- Uživatelská procedura zpráv kanálu
- Uživatelská procedura odeslání kanálu
- Uživatelská procedura příjmu kanálu
- Zpráva kanálu-uživatelská procedura opakování

Parametry jsou podobné pro každý typ výstupu a zde uvedený popis platí pro všechny z nich, s výjimkou případů, kdy je to výslovně uvedeno.

Syntaxe

```
MQ_CHANNEL_EXIT (ChannelExitParms, ChannelDefinition, DataLength,
    AgentBufferLength, AgentBuffer, ExitBufferLength, ExitBufferAddr)
```

Parametry

Volání MQ_CHANNEL_EXIT má následující parametry.

ChannelExitParms (MQCXP)-vstup/výstup

Blok parametrů uživatelské procedury kanálu.

Tato struktura obsahuje další informace týkající se vyvolání uživatelské procedury. Uživatelská procedura nastaví informace v této struktuře, aby označila, jak bude agent MCA pokračovat.

ChannelDefinition (MQCD)-vstupní/výstupní

Definice kanálu.

Tato struktura obsahuje parametry nastavené administrátorem pro řízení chování kanálu.

DataLength (MQLONG)-vstup/výstup

Délka dat.

Data závisí na typu ukončení:

- Pro uživatelskou proceduru zabezpečení zprávy kanálu při vyvolání této uživatelské procedury tento parametr obsahuje délku jakékoli zprávy zabezpečení v poli *AgentBuffer* , pokud *ExitReason* je MQXR_SEC_MSG. Pokud není žádná zpráva, je nulová. Uživatelská procedura musí nastavit toto pole na délku jakékoli zprávy zabezpečení, která se má odeslat svému partnerovi, pokud nastaví *ExitResponse* na MQXCC_SEND_SEC_MSG nebo MQXCC_SEND_AND_REQUEST_SEC_MSG. Data zprávy jsou buď v adresáři *AgentBuffer* , nebo v adresáři *ExitBufferAddr*.

  Obsah bezpečnostních zpráv je výhradní odpovědností uživatelských procedur zabezpečení.
- Pro uživatelskou proceduru zprávy kanálu při vyvolání uživatelské procedury tento parametr obsahuje délku zprávy (včetně záhlaví přenosové fronty). Uživatelská procedura musí nastavit toto pole na délku zprávy v *AgentBuffer* nebo *ExitBufferAddr* , která má pokračovat. Tato hodnota musí být větší nebo rovna délce záhlaví přenosové fronty (MQXQH).
- V případě uživatelské procedury pro odeslání nebo přijetí kanálu tento parametr při vyvolání uživatelské procedury obsahuje délku přenosu. Uživatelská procedura musí nastavit toto pole na délku přenosu v *AgentBuffer* nebo *ExitBufferAddr* , která má pokračovat.

Pokud uživatelská procedura pro zabezpečení zprávy odešle zprávu a na druhém konci kanálu není žádná uživatelská procedura pro zabezpečení zprávy, nebo pokud druhý konec nastaví *ExitResponse* na hodnotu MQXCC_OK, inicializační uživatelská procedura se znovu vyvolá s MQXR_SEC_MSG a odpovědí s hodnotou null (*DataLength* = 0).

AgentBufferDélka (MQLONG)-vstup

Délka vyrovnávací paměti agenta.

Tento parametr může být při vyvolání větší než *DataLength* .

V případě uživatelských procedur pro zprávy kanálu, odeslání a příjem může uživatelská procedura použít nevyužitý prostor při vyvolání k rozšíření dat na svém místě. Pokud se tak stane, musí být parametr **DataLength** vhodně nastaven uživatelskou procedurou.

V programovacím jazyku C je tento parametr předáván adresou.

AgentBuffer (MQBYTE x AgentBufferdélka)-vstupní/výstupní

Vyrovnávací paměť agenta.

Obsah tohoto parametru závisí na typu ukončení:

- Pro uživatelskou proceduru pro zabezpečení zprávy kanálu při vyvolání uživatelské procedury obsahuje zprávu zabezpečení, pokud *ExitReason* je MQXR_SEC_MSG. Chcete-li odeslat zprávu zabezpečení zpět, uživatelská procedura může použít tuto vyrovnávací paměť nebo vlastní vyrovnávací paměť (*ExitBufferAddr*).
- Pro uživatelskou proceduru zprávy kanálu při vyvolání uživatelské procedury tento parametr obsahuje:
  - Záhlaví přenosové fronty (MQXQH), které zahrnuje deskriptor zprávy (který sám obsahuje informace o kontextu pro zprávu), bezprostředně následované
  - Data zprávy

  Má-li zpráva pokračovat, může uživatelská procedura provést jednu z následujících akcí:

  - Ponechat obsah vyrovnávací paměti nedotčený
  - Upravit obsah na místě (vrácení nové délky dat v souboru *DataLength* ; nesmí být větší než *AgentBufferLength*)
  - Zkopírujte obsah do adresáře *ExitBufferAddr*a proveďte požadované změny.

  Žádné změny, které uživatelská procedura provede v záhlaví přenosové fronty, nejsou kontrolovány; chybné úpravy však mohou znamenat, že zprávu nelze vložit do místa určení.

- Pro uživatelskou proceduru odeslání nebo přijetí kanálu při vyvolání uživatelské procedury obsahuje data přenosu. Uživatelská procedura může provádět jednu z následujících akcí:
  - Ponechat obsah vyrovnávací paměti nedotčený
  - Upravit obsah na místě (vrácení nové délky dat v souboru *DataLength* ; nesmí být větší než *AgentBufferLength*)
  - Zkopírujte obsah do adresáře *ExitBufferAddr*a proveďte požadované změny.

  Prvních 8 bajtů dat nesmí být změněno uživatelskou procedurou.

ExitBufferDélka (MQLONG)-vstup/výstup

Délka výstupní vyrovnávací paměti.

Při prvním vyvolání uživatelské procedury je tento parametr nastaven na nulu. Poté je při příštím vyvolání uživatelské proceduře při každém vyvolání zobrazena hodnota, která je předána zpět. Hodnota není použita agentem MCA.

Poznámka: Tento parametr nesmí používat uživatelské procedury napsané v programovacích jazycích, které nepodporují datový typ ukazatele.

ExitBufferAddr (MQPTR)-vstupní/výstupní

Adresa výstupní vyrovnávací paměti.

Tento parametr je ukazatel na adresu vyrovnávací paměti úložiště spravované uživatelskou procedurou, kde se může rozhodnout vrátit zprávu nebo data přenosu (v závislosti na typu uživatelské procedury) agentovi, pokud vyrovnávací paměť agenta je nebo nemusí být dostatečně velká, nebo pokud je pro uživatelskou proceduru vhodnější tak učinit.

Při prvním vyvolání uživatelské procedury má adresa předaná uživatelské proceduře hodnotu null. Poté bude při příštím vyvolání uživatelské proceduře při každém vyvolání předána uživatelská procedura, ať už je její adresa předána zpět.

Má-li vlastnost ExitBufferaddr hodnotu null, jsou použitá data převzata z parametru AgentBuffer .

Pokud ExitBufferAddr nemá hodnotu null, jsou použitá data převzata z vyrovnávací paměti, na kterou odkazuje parametr ExitBufferAddr.

Poznámka: Tento parametr nesmí používat uživatelské procedury napsané v programovacích jazycích, které nepodporují datový typ ukazatele.

Vyvolání jazyka C

```
exitname (&ChannelExitParms, &ChannelDefinition,
&DataLength, &AgentBufferLength, AgentBuffer,
&ExitBufferLength, &ExitBufferAddr);
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
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 */
```

Vyvolání COBOL

```
     CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION,
                           DATALENGTH, AGENTBUFFERLENGTH, AGENTBUFFER,
                           EXITBUFFERLENGTH, EXITBUFFERADDR.
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
**   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.
```

Vyvolání RPG (ILE)

```
     C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     C                    CALLP     exitname(MQCXP : MQCD : DATLEN :
     C                                       ABUFL : ABUF : EBUFL :
     C                                       EBUF)
```

Definice prototypu pro volání je:

```
     D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     Dexitname         PR                  EXTPROC('exitname')
     D* Channel exit parameter block
     D MQCXP                          160A
     D* Channel definition
     D MQCD                          1328A
     D* Length of data
     D DATLEN                          10I 0
     D* Length of agent buffer
     D ABUFL                           10I 0
     D* Agent buffer
     D ABUF                             *   VALUE
     D* Length of exit buffer
     D EBUFL                           10I 0
     D* Address of exit buffer
     D EBUF                             *
```

System/390 vyvolání assembleru

```
         CALL EXITNAME,(CHANNELEXITPARMS,CHANNELDEFINITION,DATALENGTH, X
               AGENTBUFFERLENGTH,AGENTBUFFER,EXITBUFFERLENGTH,         X
               EXITBUFFERADDR)
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
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
```

Poznámky k použití

1. Funkce prováděná uživatelskou procedurou kanálu je definována poskytovatelem uživatelské procedury. Uživatelská procedura však musí být v souladu s pravidly definovanými zde a v přidruženém řídicím bloku, MQCXP.

2. Parametr **ChannelDefinition** předaný uživatelské proceduře kanálu může být jednou z několika verzí. Další informace viz pole *Version* ve struktuře MQCD.
3. Pokud uživatelská procedura kanálu obdrží strukturu MQCD s polem *Version* nastaveným na hodnotu větší než MQCD_VERSION_1, musí uživatelská procedura použít pole *ConnectionName* v MQCD, namísto pole *ShortConnectionName* .
4. Obecně platí, že uživatelské procedury kanálu mohou měnit délku dat zprávy. K tomu může dojít v důsledku toho, že uživatelská procedura přidá data do zprávy, nebo odebere data ze zprávy, nebo komprimuje nebo zašifruje zprávu. Avšak platí speciální omezení, pokud je zpráva segmentem, který obsahuje pouze část logické zprávy. Zejména nesmí dojít k žádné čisté změně délky zprávy v důsledku činností doplňujících se odesílajících a přijímajících východů.

   Například je možné, aby odesílající výstup zkrátil zprávu komprimací, ale doplňková přijímací procedura musí obnovit původní délku zprávy tím, že ji dekomprimuje, aby nedošlo ke změně délky zprávy.

   Toto omezení je způsobeno tím, že změna délky segmentu by způsobila, že by posuny pozdějších segmentů ve zprávě byly nesprávné, což by bránilo tomu, aby správce front rozpoznal, že segmenty vytvořily úplnou logickou zprávu.

MQ_CHANNEL_AUTO_DEF_EXIT-Ukončit automatickou definici kanálu

Volání MQ_CHANNEL_AUTO_DEF_EXIT popisuje parametry, které jsou předány uživatelské proceduře automatické definice kanálu volané agentem kanálu zpráv.

Správce front neposkytl žádný vstupní bod s názvem MQ_CHANNEL_AUTO_DEF_EXIT; název MQ_CHANNEL_AUTO_DEF_EXIT nemá žádný zvláštní význam, protože názvy uživatelských procedur automatické definice jsou poskytovány ve správci front.

Syntaxe

MQ_CHANNEL_AUTO_DEF_EXIT *(ChannelExitParms, ChannelDefinition)*

Parametry

Volání MQ_CHANNEL_AUTO_DEF_EXIT má následující parametry.

ChannelExitParms (MQCXP)-vstup/výstup

Blok parametrů uživatelské procedury kanálu.

Tato struktura obsahuje další informace týkající se vyvolání uživatelské procedury.Uživatelská procedura nastaví informace v této struktuře, aby označila, jak bude agent MCA pokračovat.

ChannelDefinition (MQCD)-vstupní/výstupní

Definice kanálu.

Tato struktura obsahuje parametry nastavené administrátorem pro řízení chování kanálů, které jsou vytvářeny automaticky.Uživatelská procedura nastaví informace v této struktuře, aby upravila výchozí chování nastavené administrátorem.

Vypsaná pole MQCD nesmí být změněna uživatelskou procedurou:

- *ChannelName*
- *ChannelType*
- *StrucLength*
- *Version*

Pokud se změní jiná pole, hodnota nastavená uživatelskou procedurou musí být platná. Není-li hodnota platná, zapíše se chybová zpráva do souboru protokolu chyb nebo se zobrazí na konzole (podle toho, co je vhodné pro dané prostředí).

Upozornění: Automaticky definované kanály vytvořené uživatelskou procedurou automatické definice kanálu (CHAD) nemohou nastavit popisek certifikátu, protože k navázání komunikace TLS došlo v době vytvoření kanálu. Nastavení popisku certifikátu v uživatelské proceduře CHAD pro příchozí kanály nemá žádný vliv.

Vyvolání jazyka C

```
exitname (&ChannelExitParms, &ChannelDefinition);
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
MQCXP  ChannelExitParms;   /* Channel exit parameter block */
MQCD   ChannelDefinition;  /* Channel definition */
```

Vyvolání COBOL

```
     CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION.
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
**   Channel exit parameter block
 01  CHANNELEXITPARMS.
     COPY CMQCXPV.
**   Channel definition
 01  CHANNELDEFINITION.
     COPY CMQCDV.
```

Vyvolání RPG (ILE)

```
     C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     C                   CALLP     exitname(MQCXP : MQCD)
```

Definice prototypu pro volání je:

```
     D*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     Dexitname         PR                  EXTPROC('exitname')
     D* Channel exit parameter block
     D MQCXP                       160A
     D* Channel definition
     D MQCD                       1328A
```

System/390 vyvolání assembleru

```
         CALL EXITNAME,(CHANNELEXITPARMS,CHANNELDEFINITION)
```

Parametry předané uživatelské proceduře jsou deklarovány takto:

```
CHANNELEXITPARMS   CMQCXPA  ,  Channel exit parameter block
CHANNELDEFINITION  CMQCDA   ,  Channel definition
```

Poznámky k použití

1. Funkce prováděná uživatelskou procedurou kanálu je definována poskytovatelem uživatelské procedury. Uživatelská procedura však musí být v souladu s pravidly definovanými zde a v přidruženém řídicím bloku, MQCXP.

2. Parametr **ChannelExitParms** předaný uživatelské proceduře automatické definice kanálu má strukturu MQCXP. Předaná verze MQCXP závisí na prostředí, ve kterém je uživatelská procedura spuštěna. Podrobnosti naleznete v popisu pole *Version* v části “MQCXP-Parametr uživatelské procedury kanálu” na stránce 1512 .
3. Parametr **ChannelDefinition** předaný uživatelské proceduře automatické definice kanálu je strukturou MQCD. Předaná verze MQCD závisí na prostředí, ve kterém je uživatelská procedura spuštěna. Podrobnosti naleznete v popisu pole *Version* v části “MQCD-Definice kanálu” na stránce 1471 .

MQXWAIT-Čekání při ukončení

Volání MQXWAIT čeká na výskyt události. Lze jej použít pouze z uživatelské procedury kanálu v systému z/OS.

Použití MQXWAIT pomáhá vyhnout se problémům s výkonem, které by se jinak mohly vyskytnout, pokud uživatelská procedura kanálu provede něco, co způsobí čekání. Událost, na kterou MQXWAIT čeká, signalizuje MVS ECB (řídicí blok události). ECB je popsána v popisu řídicího bloku MQXWD.

z/OS Další informace o použití MQXWAIT a o zápisu programů pro uživatelské procedury kanálu naleznete v tématu Zápis programů pro uživatelské procedury kanálu v systému z/OS .

Syntaxe

MQXWAIT ***(Hconn, WaitDesc, CompCode, Reason)***

Parametry

Volání MQXWAIT má následující parametry.

Hconn (MQHCONN)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN vydaným ve stejném nebo dřívějším vyvolání uživatelské procedury.

WaitDesc (MQXWD)-vstup/výstup

Deskriptor čekání.

Tento parametr popisuje událost, na kterou se má čekat. Podrobnosti o polích v této struktuře viz “MQXWD-deskriptor čekání na ukončení” na stránce 1527 .

CompCode (MQLONG)-výstup

Kód dokončení.

Jedná se o jeden z následujících kódů:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup

Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_XWAIT_CANCELED
(2107, X'83B') Volání MQXWAIT bylo zrušeno.

MQRC_XWAIT_ERROR
(2108, X'83C') Vyvolání volání MQXWAIT je neplatné.

Vyvolání jazyka C

```
MQXWAIT (Hconn, &WaitDesc, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN  Hconn;     /* Connection handle */
MQXWD    WaitDesc;  /* Wait descriptor */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

System/390 vyvolání assembleru

```
         CALL MQXWAIT,(HCONN,WAITDESC,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
HCONN     DS       F  Connection handle
WAITDESC  CMQXWDA  ,  Wait descriptor
COMPCODE  DS       F  Completion code
REASON    DS       F  Reason code qualifying COMPCODE
```

MQCD-Definice kanálu

Struktura MQCD obsahuje parametry, které řídí provádění kanálu. Předává se každé uživatelské proceduře kanálu, která je volána z agenta MCA (Message Channel Agent).

Další informace o uživatelských procedur kanálu viz “MQ_CHANNEL_EXIT-Ukončit kanál” na stránce 1464. Popis v tomto tématu se týká kanálů zpráv i kanálů MQI.

Pole jména ukončení

Když je volána uživatelská procedura, příslušné pole z položek *SecurityExit*, *MsgExit*, *SendExit*, *ReceiveExit*a *MsgRetryExit* obsahuje název aktuálně vyvolávané uživatelské procedury. Význam názvu v těchto polích závisí na prostředí, ve kterém je agent MCA spuštěn. Kromě místa, kde je uvedeno, je název zarovnán vlevo v poli bez vložených mezer; název je vyplněn mezerami na délku pole. V popisech, které následují, hranaté závorky ([]) označují nepovinné informace:

AIX and Linux
Název uživatelské procedury je název dynamicky zaveditelného modulu nebo knihovny s příponou názvu funkce, která se nachází v této knihovně. Název funkce musí být uzavřen v závorkách. Název knihovny může mít volitelně předponu s cestou k adresáři:

```
[ path ] library ( function )
```

Délka názvu je omezena na maximálně 128 znaků.

z/OS
Název uživatelské procedury je název zaváděcího modulu, který je platný pro specifikaci parametru EP makra LINK nebo LOAD. Délka názvu je omezena na maximálně osm znaků.

Windows
Název uživatelské procedury je název knihovny dynamického propojení s příponou názvu funkce, která se nachází v této knihovně. Název funkce musí být uzavřen v závorkách. Název knihovny může mít volitelně předponu s cestou k adresáři a jednotkou:

```
[d:][ path ] library ( function )
```

Délka názvu je omezena na maximálně 128 znaků.

IBM i
Název uživatelské procedury je 10bajtový název programu následovaný 10bajtovým názvem knihovny. Pokud jsou názvy kratší než 10 bajtů, každý název je vyplněn mezerami, aby byl 10 bajtů. Název knihovny může být *LIBL kromě případů, kdy se volá uživatelská procedura automatické definice kanálu. V takovém případě se požaduje úplný název.

Změna polí MQCD v uživatelské proceduře kanálu

Uživatelská procedura kanálu může změnit pole v MQCD. Změněná hodnota zůstává v MQCD a je předána všem zbývajícím uživatelským procedurám v řetězci ukončení a jakékoli konverzaci sdílející instanci kanálu. Změněný MQCD je také používán agentem MCA pro jeho běžné zpracování během pokračující životnosti kanálu.

Uživatelská procedura nesmí měnit následující pole MQCD:

- ChannelName
- ChannelType
- StrucLength
- Verze

Související odkazy
“Pole” na stránce 1472
Toto téma obsahuje seznam všech polí ve struktuře MQCD a popisuje jednotlivá pole.

“C prohlášení” na stránce 1499
Tato deklarace je deklarací jazyka C pro strukturu MQCD.

“Deklarace jazyka COBOL” na stránce 1501
Tato deklarace je deklarací jazyka COBOL pro strukturu MQCD.

“Deklarace RPG (ILE)” na stránce 1503
Tato deklarace je deklarací RPG pro strukturu MQCD.

“System/390 deklarace modulu asembleru” na stránce 1506
Tato deklarace je deklarací modulu sestavení System/390 pro strukturu MQCD.

“Vizuální základní deklarace” na stránce 1507
Tato deklarace je deklarací jazyka Visual Basic struktury MQCD.

“Změna polí MQCD v uživatelské proceduře kanálu” na stránce 1509
Uživatelská procedura kanálu může změnit pole v MQCD. Tyto změny se však obvykle neprovádějí, s výjimkou uvedených okolností.

Pole
Toto téma obsahuje seznam všech polí ve struktuře MQCD a popisuje jednotlivá pole.

BatchDataLimit (MQLONG)
Toto pole určuje limit (v kilobajtech) množství dat, která lze odeslat prostřednictvím kanálu před vytvořením synchronizačního bodu.

Bod synchronizace se provede po zprávě, která způsobí dosažení limitu zpráv proteklých kanálem.

Dávka bude ukončena, je-li splněna jedna z následujících podmínek:

- **BatchSize** zpráv bylo odesláno.
- Počet odeslaných bajtů: **BatchDataLimit** .
- Přenosová fronta je prázdná a **BatchInterval** je překročena.

Hodnota musí být v rozsahu 0-999999. Výchozí hodnota je 5000.

Hodnota nula v tomto atributu znamená, že se na dávky v tomto kanálu nepoužije žádný datový limit.

Tento parametr platí pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSRCVR nebo MQCHT_CLUSSDR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_11.

BatchHeartbeat (MQLONG)
Toto pole uvádí časový interval, který se používá ke spuštění synchronizačního signálu dávky pro kanál.

Synchronizační signál dávky umožňuje odesílacím kanálům určit, zda je instance vzdáleného kanálu stále aktivní, než bude nejistá. Synchronizační signál dávky se vyskytne, pokud kanál odesilatele nekomunikoval s instancí vzdáleného kanálu v uvedeném časovém intervalu.

Hodnota je v rozsahu 0 až 999 999; jednotky jsou milisekundy. Hodnota nula označuje, že prezenční signál dávky není povolen.

Toto pole je relevantní pouze pro kanály, které mají hodnotu *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_7.

BatchInterval (MQLONG)
Toto pole určuje přibližnou dobu v milisekundách, po kterou kanál udržuje dávku otevřenou, pokud bylo v aktuální dávce přeneseno méně než *BatchSize* zpráv.

Pokud je hodnota *BatchInterval* větší než nula, dávka se ukončí podle toho, která z následujících událostí nastane dříve:

- *BatchSize* zpráv bylo odesláno, nebo
- Od spuštění dávky uplynulo *BatchInterval* milisekund.

Pokud je *BatchInterval* nula, dávka se ukončí podle toho, která z následujících událostí nastane dříve:

- *BatchSize* zpráv bylo odesláno, nebo
- přenosová fronta je prázdná.

BatchInterval musí být v rozsahu od 0 do 999 999 999 999.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

BatchSize (MQLONG)
Toto pole určuje maximální počet zpráv, které lze odeslat prostřednictvím kanálu před synchronizací kanálu.

Toto pole není relevantní pro kanály s hodnotou *ChannelType* MQCHT_SVRCONN nebo MQCHT_CLNTCONN.

CertificateLabel (MQCHAR64)
V tomto poli jsou uvedeny podrobnosti o používaném popisku certifikátu.

IBM MQ inicializuje výchozí hodnotu pro pole *CertificateLabel* jako mezery.

Tato hodnota je za běhu interpretována jako výchozí hodnota a je zpětně kompatibilní.

Například uvedení verze MQCD nižší než 11 nebo použití výchozí hodnoty mezer pro pole *CertificateLabel* znamená, že se toto pole ignoruje.

Délka tohoto pole je dána hodnotou MQ_CERT_LABEL_LENGTH.

ChannelMonitoring (MQLONG)
Toto pole určuje aktuální úroveň shromažďování dat monitorování pro daný kanál.

Toto pole není relevantní pro kanály s typem ChannelType MQCHT_CLNTCONN.

Jedná se o jednu z následujících hodnot:

- MQMON_OFF
- MQMON_LOW
- MQMON_MEDIUM
- MQMON_HIGH-vysoká

Toto je vstupní pole pro ukončení. Není přítomen, pokud je hodnota *Version* menší než MQCD_VERSION_8.

ChannelName (MQCHAR20)
Toto pole určuje název definice kanálu.

Aby bylo možné komunikovat, musí existovat definice kanálu se stejným názvem na vzdáleném počítači.

Název smí obsahovat pouze tyto znaky:

- Velká písmena A-Z
- Malá písmena a-z
- Číslice 0-9
- Tečka (.)
- Lomítko (/)
- Podtržítko (_)
- Procento (%)

a být zprava vyplněn mezerami. Vložené mezery ani mezery na začátku nejsou povoleny.

Délka tohoto pole je dána hodnotou MQ_CHANNEL_NAME_LENGTH.

ChannelStatistics (MQLONG)
Toto pole určuje aktuální úroveň shromažďování statistických dat pro daný kanál.

Toto pole není relevantní pro kanály s typem ChannelType MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

Jedná se o jednu z následujících hodnot:

- MQMON_OFF
- MQMON_LOW
- MQMON_MEDIUM
- MQMON_HIGH-vysoká

Toto je vstupní pole pro ukončení. Není přítomen, pokud je hodnota *Version* menší než MQCD_VERSION_8.

ChannelType (MQLONG)
Toto pole určuje typ kanálu.

Jedná se o jednu z následujících hodnot:

MQCHT_SENDER-odesilatel
Odesílatel.

MQCHT_SERVER
.

MQCHT_RECEIVER
Přijímač.

MQCHT_REQUESTER
Žadatel.

MQCHT_CLNTCONN
Připojení klienta.

MQCHT_SVRCONN
Server-připojení (pro použití klienty).

MQCHT_CLUSSDR
Odesilatel klastru.

MQCHT_CLUSRCVR
Přijímač klastru.

ClientChannelVáha (MQLONG)
Toto pole určuje váhu určující, která definice kanálu připojení klienta má být použita.

Atribut Váha ClientChannelse používá, aby definice kanálů klienta mohly být vybrány náhodně na základě jejich váhy, když je k dispozici více než jedna vhodná definice. Pokud klient zadá požadavek MQCONN na připojení ke skupině správců front zadáním názvu správce front začínajícím hvězdičkou a v tabulce CCDT (Client Channel Definition Table) je k dispozici více než jedna vhodná definice kanálu, bude definice, která má být použita, náhodně vybrána na základě váhy s příslušnými definicemi ClientChannelWeight (0) vybranými jako první v abecedním pořadí.

Zadejte hodnotu v rozsahu 0 - 99. Výchozí hodnota je 0.

Hodnota 0 znamená, že není prováděno žádné vyvažování zátěže a dostupné definice jsou vybírány v abecedním pořadí. Chcete-li povolit vyvažování zátěže, vyberte hodnotu v rozsahu 1 až 99, přičemž hodnota 1 znamená nejnižší a hodnota 99 nejvyšší váhu. Rozdělení zpráv mezi dva nebo více kanálů s nenulovými váhami je úměrné poměru těchto vah. Například tři kanály s hodnotou váhy ClientChannel2, 4 a 14 jsou vybrány přibližně 10%, 20% a 70% času. Tato distribuce není zaručena.

Tento atribut je platný pouze pro typ kanálu připojení klienta.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je *Verze* menší než MQCD_VERSION_9.

ClusterPtr (MQPTR)
Toto pole určuje adresu seznamu názvů klastrů.

Pokud je *ClustersDefined* větší než nula, je tato adresa adresou seznamu názvů klastrů. Kanál patří ke každému uvedenému klastru.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_5.

ClustersDefined (MQLONG)
Toto pole určuje počet klastrů, ke kterým kanál náleží.

Toto pole je počet názvů klastrů, na které odkazuje *ClusterPtr*. Je nula nebo větší.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_5.

CLWLChannelPriority (MQLONG)
Toto pole určuje prioritu kanálu pracovní zátěže klastru.

Algoritmus výběru správce pracovní zátěže vybere cíl s nejvyšší prioritou ze sady cílů vybraných na základě ohodnocení důležitosti. Pokud existují dva možní správci cílových front, lze tento atribut použít k převedení jednoho správce front na druhého správce front. Všechny zprávy přejdou do správce front s nejvyšší prioritou až do jejího ukončení, a poté přejdou do správce front s další nejvyšší prioritou.

Hodnota je v rozsahu 0 až 9. Výchozí hodnota je 0.

Toto je vstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCD_VERSION_8.

Další informace naleznete v tématu Konfigurace klastru správců front.

CLWLChannelRank (MQLONG)
Toto pole určuje úroveň kanálu pracovní zátěže klastru.

Algoritmus výběru správce pracovní zátěže vybere cíl s nejvyšším hodnocením. Je-li konečným cílem správce front v jiném klastru, můžete nastavit pořadí intermediačních správců front brány (v průniku sousedních klastrů), takže algoritmus výběru správně vybere cílového správce front, který bude blíže konečnému cíli.

Hodnota je v rozsahu 0 až 9. Výchozí hodnota je 0.

Toto je vstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCD_VERSION_8.

Další informace naleznete v tématu Konfigurace klastru správců front.

CLWLChannelWeight (MQLONG)
Toto pole určuje váhu kanálu pracovní zátěže klastru.

Váha kanálu pracovní zátěže klastru.

Algoritmus výběru správce pracovní zátěže používá atribut "váha" kanálu k posunu volby místa určení, aby bylo možné na konkrétní počítač odeslat více zpráv. Můžete například kanálu na velkém serveru UNIX přidělit větší "váhu" než jinému kanálu na malém stolním počítači a algoritmus výběru zvolí server UNIX častěji než PC.

Hodnota je v rozsahu 1 až 99. Výchozí hodnota je 50.

Toto je vstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCD_VERSION_8.

Další informace naleznete v tématu Konfigurace klastru správců front.

ConnectionAffinity (MQLONG)
Toto pole určuje, zda aplikace klienta, které se připojují vícekrát pomocí stejného názvu správce front, používají stejný kanál klienta.

Tento atribut použijte v případě, že je dostupných několik použitelných definic kanálu.

Hodnota je jedna z následujících:

MQCAFTY_PREFERRED
První připojení v procesu, které čte tabulku CCDT (Client Channel Definition Table), vytvoří seznam použitelných definic na základě váhy s jakýmikoli použitelnými definicemi CLNTWGHT (0), a to nejprve v abecedním pořadí. Každé připojení v procesu se pokusí připojit pomocí první definice v seznamu. Pokud se navázání připojení nezdaří, je použita další definice. Neúspěšné definice s jinými hodnotami CLNTWGHT než 0 se přesunou na konec seznamu. Definice CLNTWGHT(0) zůstávají na začátku seznamu a jsou vybrány jako první pro každé připojení.

Každý proces klienta se stejným názvem hostitele vždy vytvoří stejný seznam.

V případě klientských aplikací napsaných v jazyku C, C + + nebo v programovacím rámci .NET (včetně plně spravovaných .NET) se seznam aktualizuje, pokud byla tabulka CCDT od vytvoření seznamu změněna.

Tato hodnota je výchozí hodnota.

MQCAFTY_NONE
První připojení v procesu, které čte tabulku CCDT, vytvoří seznam použitelných definic. Všechny připojení v procesu vybírají aplikovatelnou definici, v závislosti na vážení s jakýmikoliv aplikovatelnými definicemi CLNTWGHT(0), vybranými jako první v abecedním pořadí.

V případě klientských aplikací napsaných v jazyku C, C + + nebo v programovacím rámci .NET (včetně plně spravovaných .NET) se seznam aktualizuje, pokud byla tabulka CCDT od vytvoření seznamu změněna.

Tento atribut je platný pouze pro typ kanálu připojení klienta.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je *Verze* menší než MQCD_VERSION_9.

ConnectionName (MQCHAR264)
Toto pole určuje název připojení pro kanál.

Pro přijímací kanály klastru (je-li zadán) se CONNAME vztahuje k lokálnímu správci front a pro ostatní kanály k cílovému správci front. Hodnota, kterou zadáte, závisí na přenosovém protokolu (*TransportType*), který se má použít:

- V případě produktu MQXPT_LU62se jedná o úplný název partnerské logické jednotky.
- Pro MQXPT_NETBIOS je to název NetBIOS definovaný na vzdáleném počítači.
- Pro MQXPT_TCP je to buď název hostitele, síťová adresa vzdáleného počítače uvedená v IPv4 desítkovém formátu s tečkami nebo IPv6 hexadecimálním formátu, nebo lokální počítač pro přijímací kanály klastru.
- Pro MQXPT_SPX se jedná o adresu ve stylu SPX, která obsahuje 4bajtovou síťovou adresu, 6bajtovou adresu uzlu a 2bajtové číslo soketu.

Při definování kanálu není toto pole relevantní pro kanály s hodnotou *ChannelType* MQCHT_SVRCONN nebo MQCHT_RECEIVER. Je-li však definice kanálu předána uživatelské proceduře, obsahuje toto pole adresu partnera bez ohledu na typ kanálu.

Délka tohoto pole je dána hodnotou MQ_CONN_NAME_LENGTH. Toto pole není k dispozici, pokud je hodnota *Version* menší než hodnota MQCD_VERSION_2.

DataConversion (MQLONG)
Toto pole určuje, zda se odesílající agent kanálu zpráv pokusí o převod dat zpráv aplikace, pokud přijímající agent kanálu zpráv nemůže provést tento převod.

Toto pole platí pouze pro zprávy, které nejsou segmenty logických zpráv; agent MCA se nikdy nepokusí převést zprávy, které jsou segmenty.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR. Jedná se o jednu z následujících položek:

MQCDC_SENDER_CONVERSION
Převod podle odesílatele.

MQCDC_NO_SENDER_CONVERSION
Žádný převod podle odesílatele.

DefReconnect (MQLONG)
Atribut kanálu DefReconnect nastavuje výchozí hodnotu atributu opětovného připojení pro kanál připojení klienta.

Volba pro výchozí automatické opětovné připojení klienta. Produkt IBM MQ MQI client můžete nakonfigurovat tak, aby znovu automaticky připojil aplikaci klienta. Produkt IBM MQ MQI client se pokusí znovu připojit ke správci front po selhání připojení. Pokusí se připojit znovu, aniž by aplikační klient vydal volání MQCONN nebo MQCONNX MQI.

Opětovné připojení je volba MQCONNX . Pomocí atributu kanálu DefReconnect můžete přidat chování při opětovném připojení k existujícím aplikacím, které používají MQCONN. Můžete také změnit chování opětovného připojení aplikací, které používají produkt MQCONNX.

Můžete také nastavit hodnotu DefRecon ze souboru mqclient.ini , chcete-li nastavit nebo upravit chování opětovného připojení. Hodnota DefRecon ze souboru mqclient.ini má přednost před atributem kanálu DefReconnect .

Syntax

DefReconnect (MQRCN_NO (default) |MQRCN_YES|MQRCN_Q_MGR| MQRCN_DISABLED)

Parametry

MQRCN_NO
MQRCN_NO je výchozí hodnota.

Pokud není přepsáno **MQCONNX**, klient není automaticky znovu připojen.

MQRCN_YES
Pokud není přepsáno **MQCONNX**, klient se automaticky znovu připojí.

MQRCN_Q_MGR
Není-li přepsáno **MQCONNX**, klient se znovu připojí automaticky, ale pouze ke stejnému správci front. Volba QMGR má stejný účinek jako MQCNO_RECONNECT_Q_MGR.

MQRCN_DISABLED
Opětovné připojení je zakázáno, a to i v případě, že to vyžaduje klientský program pomocí volání **MQCONNX** MQI.

Automatické opětovné připojení klienta není podporováno produktem IBM MQ classes for Java.

Tabulka 822. Automatické opětovné připojení závisí na hodnotách nastavených v aplikaci a definici kanálu.

DefReconnect	**Volby opětovného připojení nastavené v aplikaci**			
	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

Související pojmy
Automatické opětovné připojení klienta
Opětovné připojení kanálu a klienta
Sekce CHANNELS konfiguračního souboru klienta
Související odkazy
“Volby (MQLONG) pro MQCNO” na stránce 324
Volby, které řídí akci MQCONNX.

Popis (MQCHAR64)
Toto pole lze použít pro popisný komentář.

Obsah pole nemá pro agenty kanálu zpráv žádný význam. Musí však obsahovat pouze znaky, které lze zobrazit. Nemůže obsahovat žádné nulové znaky; je-li to nutné, je zprava vyplněn mezerami. V instalaci DBCS může pole obsahovat znaky DBCS (s maximální délkou pole 64 bajtů).

Poznámka: Pokud toto pole obsahuje znaky, které nejsou ve znakové sadě správce front (jak je definováno atributem správce front **CodedCharSetId**), mohou být tyto znaky nesprávně přeloženy, pokud je toto pole odesláno jinému správci front.

Délka tohoto pole je dána hodnotou MQ_CHANNEL_DESC_LENGTH.

DiscInterval (MQLONG)
Toto pole určuje maximální dobu v sekundách, po kterou kanál čeká na doručení zprávy do přenosové fronty před ukončením kanálu.

Jinými slovy, určuje interval odpojení.

Nulová hodnota A způsobí, že agent MCA bude čekat neomezeně dlouho.

Pro kanály připojení serveru používající protokol TCP představuje interval hodnotu odpojení neaktivity klienta určenou v sekundách. Pokud připojení serveru neobdrželo po tuto dobu žádnou komunikaci od svého partnerského klienta, ukončí připojení. Interval nečinnosti připojení serveru se používá pouze mezi voláními rozhraní IBM MQ API z klienta, takže během přerušitelného volání MQGET s čekáním není odpojen žádný klient.

Tento atribut nelze použít pro kanály připojení serveru používající jiné protokoly než TCP.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR, MQCHT_CLUSRCVR nebo MQCHT_SVRCONN.

ExitDataDélka (MQLONG)
Toto pole uvádí délku v bajtech každé položky uživatelských dat v seznamech položek uživatelských dat adresovaných poli *MsgUserDataPtr*, *SendUserDataPtr*a *ReceiveUserDataPtr* .

Tato délka nemusí být nutně stejná jako MQ_EXIT_DATA_LENGTH.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

ExitNameDélka (MQLONG)
Toto pole určuje délku jednotlivých názvů v bajtech v seznamech názvů ukončení adresovaných poli *MsgExitPtr*, *SendExitPtr*a *ReceiveExitPtr* .

Tato délka nemusí být nutně stejná jako MQ_EXIT_NAME_LENGTH.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

HdrCompSeznam [2] (MQLONG)
Toto pole určuje seznam technik komprese dat záhlaví, které jsou podporovány kanálem.

Seznam obsahuje jednu nebo více následujících hodnot:

MQCOMPRESS_NONE
Neprovádí se žádná komprese dat hlavičky.

MQCOMPRESS_SYSTEM
Provádí se komprese dat hlavičky.

MQCOMPRESS_NOT_AVAILABLE
Nepoužité hodnoty v seznamu jsou nastaveny na tuto hodnotu.

Toto je vstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCD_VERSION_8.

HeartbeatInterval (MQLONG)
Toto pole určuje dobu v sekundách mezi toky prezenčního signálu.

Interpretace tohoto pole závisí na typu kanálu, jak je uvedeno níže:

- Pro typ kanálu MQCHT_SENDER, MQCHT_SERVER, MQCHT_RECEIVER MQCHT_REQUESTER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR je toto pole doba v sekundách mezi toky prezenčního signálu předanými z vysílajícího agenta MCA, když v přenosové frontě nejsou žádné zprávy. To dává přijímajícímu MCA možnost uvést kanál do klidového stavu. Chcete-li být užitečné, *HeartbeatInterval* musí být menší než *DiscInterval*.

- Pro typ kanálu MQCHT_CLNTCONN nebo MQCHT_SVRCONN s polem Konverzace sdílení MQCD nastaveným na nulu je toto pole dobou v sekundách mezi toky synchronizačních signálů předávanými z agenta MCA serveru, když tento agent MCA zadal volání MQGET s volbou MQGMO_WAIT jménem klientské aplikace. To umožňuje, aby agent MCA serveru zpracoval situace, kdy dojde k selhání připojení klienta během operace MQGET s MQGMO_WAIT.
- Pro typ kanálu MQCHT_CLNTCONN nebo MQCHT_SVRCONN s polem Konverzace sdílení MQCD nastaveným na nenulovou hodnotu je toto pole dobou v sekundách mezi toky synchronizačních signálů, když nejsou žádné datové toky odesílány nebo přijímány. To umožňuje efektivní uvedení kanálu do klidového stavu.

Hodnota je v rozsahu 0 až 999 999. Použitá hodnota je větší z hodnot zadaných na straně odesílání a na straně příjmu, pokud není na obou stranách uvedena hodnota 0, v takovém případě nedojde k žádné výměně prezenčního signálu.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

KeepAliveInterval (MQLONG)
Toto pole uvádí hodnotu předanou do komunikačního zásobníku pro časování udržení aktivity pro kanál.

Hodnota je použitelná pro komunikační protokoly TCP/IP a SPX, ačkoli ne všechny implementace podporují tento parametr.

Hodnota je v rozsahu 0 až 99 999; jednotky jsou sekundy. Hodnota nula označuje, že udržení aktivity kanálu není povoleno, i když k udržení aktivity může dojít i v případě, že je povoleno udržení aktivity TCP/IP (spíše než udržení aktivity kanálu). Následující speciální hodnota je také platná:

MQKAI_AUTO
Automatická.

Tato hodnota označuje, že interval udržení aktivity je vypočítán z vyjednaného intervalu prezenčního signálu, jak je uvedeno níže:

- Pokud je vyjednaný interval prezenčního signálu větší než nula, použitý interval udržení aktivity je interval prezenčního signálu plus 60 sekund.
- Pokud je vyjednaný interval prezenčního signálu nula, použitý interval udržení aktivity je nula.

- V systému z/OSse udržení aktivity TCP/IP vyskytne, když je v objektu správce front uvedeno TCPKEEP (YES).
- V jiných prostředích se udržení aktivity TCP/IP vyskytne, když je parametr **KEEPALIVE=YES** uveden v sekci TCP v konfiguračním souboru distribuovaných front.

Toto pole je relevantní pouze pro kanály, které mají hodnotu *TransportType* MQXPT_TCP nebo MQXPT_SPX.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_7.

LocalAddress (MQCHAR48)
Toto pole určuje lokální adresu TCP/IP definovanou pro kanál pro odchozí komunikaci.

Toto pole je prázdné, pokud pro odchozí komunikaci není definována žádná specifická adresa. Adresa může volitelně obsahovat číslo portu nebo rozsah čísel portů. Formát této adresy je:

```
[ip-addr][(low-port[,high-port])]
```

kde hranaté závorky ([]) označují volitelné informace, `ip-addr` je uvedeno v IPv4 tečkovém desítkovém, IPv6 hexadecimálním nebo alfanumerickém tvaru a `low-port` a `high-port` jsou čísla portů uzavřená v závorkách. Všechny jsou volitelné.

Specifická adresa IP, port nebo rozsah portů pro odchozí komunikace jsou užitečné ve scénářích obnovy, kde je kanál restartován v jiném zásobníku TCP/IP.

LocalAddress má podobný tvar jako *ConnectionName*, ale nesmí být s ním zaměňován. *LocalAddress* uvádí charakteristiku lokální komunikace, zatímco *ConnectionName* uvádí, jak dosáhnout vzdáleného správce front.

V produktu IBM MQ 9.3.0bylo aktualizováno rozhraní JMQI (Java Message Queueing Interface), aby se zajistilo, že po vytvoření instance kanálu a připojení ke správci front je v objektu MQCD nastaveno pole lokální adresy. To znamená, že když uživatelská procedura kanálu zapsaná v souboru Java volá metodu `MQCD.getLocalAddress()`, tato metoda vrátí lokální adresu, kterou instance kanálu používá. Před IBM MQ 9.3.0nebyla uživatelská procedura zabezpečení kanálu schopna přistupovat k lokální adrese používané instancí kanálu a metoda `MQCD.getLocalAddress()` vrátila hodnotu null.

Toto pole je relevantní pouze pro kanály s hodnotou *TransportType* MQXPT_TCP a hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER, MQCHT_CLNTCONN, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Délka tohoto pole je dána hodnotou MQ_LOCAL_ADDRESS_LENGTH. Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_7.

LongMCAUserIdLength (MQLONG)
Toto pole určuje délku úplného identifikátoru uživatele MCA v bajtech, na který odkazuje *LongMCAUserIdPtr*.

Toto pole není relevantní pro kanály s hodnotou *ChannelType* MQCHT_CLNTCONN.

Toto je vstupní/výstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_6.

LongMCAUserIdPtr (MQPTR)
Toto pole určuje adresu dlouhého identifikátoru uživatele MCA.

Pokud je *LongMCAUserIdLength* větší než nula, toto pole je adresa úplného identifikátoru uživatele MCA. Délka úplného identifikátoru je dána *LongMCAUserIdLength*. Prvních 12 bajtů identifikátoru uživatele MCA je také obsaženo v poli *MCAUserIdentifier*.

Podrobnosti o identifikátoru uživatele MCA naleznete v popisu pole *MCAUserIdentifier* .

Toto pole není relevantní pro kanály s hodnotou *ChannelType* MQCHT_SDR, MQCHT_SVR, MQCHT_CLNTCONN nebo MQCHT_CLUSSDR.

Toto je vstupní/výstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_6.

LongRemoteUserIdDélka (MQLONG)
Toto pole uvádí délku (v bajtech) úplného identifikátoru vzdáleného uživatele, na který ukazuje *LongRemoteUserIdPtr*.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_6.

LongRemoteUserIdPtr (MQPTR)
Toto pole uvádí adresu dlouhého identifikátoru vzdáleného uživatele.

Pokud je *LongRemoteUserIdLength* větší než nula, tento příznak je adresou úplného identifikátoru vzdáleného uživatele. Délka úplného identifikátoru je dána *LongRemoteUserIdLength*. Prvních 12 bajtů identifikátoru vzdáleného uživatele je také obsaženo v poli *RemoteUserIdentifier*.

Podrobnosti o identifikátoru vzdáleného uživatele viz popis pole *RemoteUserIdentifier* .

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_6.

Počet LongRetry(MQLONG)
Toto pole určuje počet použitý po vyčerpání počtu určeného parametrem *ShortRetryCount* .

Určuje maximální počet dalších pokusů o připojení ke vzdálenému počítači v intervalech určených parametrem *LongRetryInterval*, než se zaprotokoluje chyba operátorovi.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Interval LongRetry(MQLONG)
Toto pole určuje maximální počet sekund čekání před opětovným pokusem o připojení ke vzdálenému počítači.

Interval mezi opakovanými pokusy lze prodloužit, pokud má kanál čekat na aktivaci.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

MaxInstances (MQLONG)
Toto pole určuje maximální počet souběžných instancí jednotlivého kanálu připojení serveru, který lze spustit.

Toto pole se používá pouze pro kanály připojení serveru.

Pole může mít hodnotu v rozsahu 0-999 999 999 999. Hodnota nula předchází všechny přístupy klienta.

Výchozí hodnota tohoto pole je 999 999 999.

Je-li hodnota tohoto pole snížena na počet, který je menší než počet instancí kanálu připojení serveru, které jsou aktuálně spuštěny, nebudou tyto spuštěné instance ovlivněny. Nové instance však nelze spustit, dokud nebude ukončen dostatečný počet existujících instancí, aby byl počet aktuálně spuštěných instancí menší než hodnota pole.

MaxInstancesPerClient (MQLONG)
Toto pole určuje maximální počet souběžných instancí jednotlivého kanálu připojení serveru, který lze spustit z jednoho klienta.

V tomto kontextu se připojení, která pocházejí ze stejné vzdálené síťové adresy, považují za připojení pocházející od stejného klienta.

Toto pole se používá pouze pro kanály připojení serveru.

Pole může mít hodnotu v rozsahu 0-999 999 999 999. Hodnota nula předchází všechny přístupy klienta.

Výchozí hodnota tohoto pole je 999 999 999.

Pokud je hodnota tohoto pole snížena na počet, který je menší než počet instancí kanálu připojení serveru, které jsou aktuálně spuštěny z jednotlivých klientů, nebudou tyto spuštěné instance ovlivněny. Nové instance z kteréhokoli z těchto klientů však nelze spustit, dokud nepřestane běžet dostatečný počet existujících instancí, takže počet aktuálně spuštěných instancí pocházejících z klienta, který se pokouší spustit novou instanci, je menší než hodnota pole.

MaxMsgDélka (MQLONG)
Toto pole určuje maximální délku zprávy, kterou lze přenést kanálem.

Ta je porovnána s hodnotou pro vzdálený kanál a skutečné maximum je nižší z těchto dvou hodnot.

Název MCAName (MQCHAR20)
Toto pole je vyhrazené pole.

Hodnota tohoto pole je prázdná.

Délka tohoto pole je dána hodnotou MQ_MCA_NAME_LENGTH.

MCASecurityId (MQBYTE40)
Toto pole určuje identifikátor zabezpečení pro agenta MCA.

Toto pole není relevantní pro kanály s hodnotou *ChannelType* MQCHT_CLNTCONN.

Následující speciální hodnota označuje, že neexistuje žádný identifikátor zabezpečení:

MQSID_NONE
Nebyl uveden žádný identifikátor zabezpečení.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQSID_NONE_ARRAY; tato konstanta má stejnou hodnotu jako MQSID_NONE, ale je polem znaků místo řetězce.

Toto je vstupní/výstupní pole pro ukončení. Délka tohoto pole je dána hodnotou MQ_SECURITY_ID_LENGTH. Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_6.

MCAType (MQLONG)
Toto pole určuje typ programu agenta kanálu zpráv.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Hodnota je jedna z následujících:

MQMCAT_PROCESS
process.

Agent oznamovacího kanálu je spuštěn jako oddělený proces.

MQMCAT_THREAD
Podproces (Multiplatforms).

Agent oznamovacího kanálu je spuštěn jako oddělené vlákno.

Toto pole není přítomno, pokud je *Verze* menší než MQCD_VERSION_2.

MCAUserIdentifier (MQCHAR12)
Toto pole určuje identifikátor uživatele pro agenta MCA (message channel agent).

Toto pole používá prvních 12 bajtů identifikátoru uživatele MCA a může být nastaveno agentem zabezpečení.

Existují dvě pole, která obsahují identifikátor uživatele MCA:

- *MCAUserIdentifier* obsahuje prvních 12 bajtů identifikátoru uživatele MCA a je vyplněn mezerami, pokud je identifikátor kratší než 12 bajtů. *MCAUserIdentifier* může být prázdné.
- *LongMCAUserIdPtr* ukazuje na úplný identifikátor uživatele MCA, který může být delší než 12 bajtů. Jeho délka je dána *LongMCAUserIdLength*. Úplný identifikátor neobsahuje žádné koncové mezery a není ukončen s hodnotou null. Pokud je identifikátor prázdný, *LongMCAUserIdLength* je nula a hodnota *LongMCAUserIdPtr* není definována.

  Poznámka: Parametr *LongMCAUserIdPtr* není přítomen, pokud je hodnota *Version* menší než hodnota MQCD_VERSION_6.

Pokud je identifikátor uživatele MCA neprázdný, uvádí identifikátor uživatele, který má agent kanálu zpráv použít pro autorizaci pro přístup k prostředkům IBM MQ . Pro typy kanálů MQCHT_REQUESTER, MQCHT_RECEIVER a MQCHT_CLUSRCVR platí, že pokud má parametr PutAuthority hodnotu MQPA_DEFAULT, jedná se o identifikátor uživatele používaný pro kontroly autorizace pro operace vložení do cílových front.

Pokud je identifikátor uživatele MCA prázdný, agent kanálu zpráv použije svůj výchozí identifikátor uživatele.

Identifikátor uživatele MCA může být nastaven uživatelskou procedurou pro zabezpečení zprávy tak, aby označoval identifikátor uživatele, který musí agent kanálu zpráv používat. Uživatelská procedura

může změnit buď *MCAUserIdentifier*, nebo řetězec, na který ukazuje *LongMCAUserIdPtr*. Pokud jsou obě změny změněny, ale liší se od sebe, agent MCA použije *LongMCAUserIdPtr* jako předvolbu *MCAUserIdentifier*. Pokud uživatelská procedura změní délku řetězce adresovaného pomocí *LongMCAUserIdPtr*, *LongMCAUserIdLength* musí být nastavena odpovídajícím způsobem. Pokud uživatelská procedura zvýší délku identifikátoru, uživatelská procedura musí přidělit paměť požadované délky, nastavit tuto paměť na požadovaný identifikátor a adresu této paměti umístit do adresáře *LongMCAUserIdPtr*. Uživatelská procedura je zodpovědná za uvolnění úložiště při pozdějším vyvolání uživatelské procedury s příčinou MQXR_TERM.

Pro kanály s hodnotou *ChannelType* MQCHT_SVRCONN platí, že pokud je hodnota *MCAUserIdentifier* v definici kanálu prázdná, je do ní zkopírován libovolný identifikátor uživatele přenesený z klienta. Tento identifikátor uživatele (po každé úpravě uživatelskou procedurou pro zabezpečení na serveru) je ten, pod kterým se předpokládá spuštění aplikace klienta.

Identifikátor uživatele MCA není relevantní pro kanály s hodnotou *ChannelType* MQCHT_SDR, MQCHT_SVR, MQCHT_CLNTCONN, MQCHT_CLUSSDR.

Toto je vstupní/výstupní pole pro ukončení. Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_2.

ModeName (MQCHAR8)
Toto pole určuje název režimu LU 6.2 .

Toto pole je relevantní pouze v případě, že přenosový protokol (*TransportType*) je MQXPT_LU62a *ChannelType* není MQCHT_SVRCONN nebo MQCHT_RECEIVER.

Toto pole je vždy prázdné. Informace jsou místo toho obsaženy v objektu Side komunikace.

Délka tohoto pole je dána hodnotou MQ_MODE_NAME_LENGTH.

MsgCompList [16] (MQLONG)
Toto pole určuje seznam technik komprese dat zpráv, které jsou podporovány kanálem.

Seznam obsahuje jednu nebo více následujících hodnot:

MQCOMPRESS_NONE
Neprovádí se žádná komprese dat zprávy.

MQCOMPRESS_RLE
Komprese dat zprávy se provádí pomocí kódování délky spuštění.

MQCOMPRESS_ZLIBFAST
Komprese dat zprávy se provádí pomocí techniky komprese zlib. Preferuje se rychlá komprese.

MQCOMPRESS_ZLIBHIGH
Komprese dat zprávy se provádí pomocí techniky komprese zlib. Preferuje se vysoká úroveň komprese.

MQCOMPRESS_ANY
Pro kompresi zpráv lze použít libovolnou techniku komprese podporovanou správcem front. MQCOMPRESS_ANY je platný pouze pro kanály příjemce, žadatele a připojení serveru.

MQCOMPRESS_NOT_AVAILABLE
Nepoužité hodnoty v seznamu jsou nastaveny na tuto hodnotu.

Toto je vstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCD_VERSION_8.

MsgExit (MQCHARn)
Toto pole určuje název uživatelské procedury pro zprávy kanálu.

Není-li tento název prázdný, je uživatelská procedura volána v následujících časech:

- Okamžitě po načtení zprávy z přenosové fronty (odesilatel nebo server) nebo bezprostředně před vložením zprávy do cílové fronty (příjemce nebo žadatel).

  Uživatelské proceduře je poskytnuta celá zpráva aplikace a záhlaví přenosové fronty pro úpravu.

- Při inicializaci a ukončení kanálu.

Toto pole není relevantní pro kanály s hodnotou *ChannelType* MQCHT_SVRCONN nebo MQCHT_CLNTCONN; pro tyto kanály není nikdy vyvolána uživatelská procedura pro zprávy.

Popis obsahu tohoto pole v různých prostředích viz “MQCD-Definice kanálu” na stránce 1471 .

Délka tohoto pole je dána hodnotou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro dané prostředí.

MsgExitPtr (MQPTR)
Toto pole určuje adresu prvního pole *MsgExit* .

Je-li hodnota *MsgExitsDefined* větší než nula, je tato adresa adresou seznamu názvů jednotlivých uživatelských procedur pro zprávy kanálu v řetězci.

Každý název je v poli o délce *ExitNameLength*, zprava vyplněný mezerami. Vedle sebe se nacházejí pole *MsgExitsDefined* -jedno pro každou uživatelskou proceduru.

Všechny změny provedené v těchto názvech uživatelskou procedurou jsou zachovány, ačkoli uživatelská procedura kanálu zpráv neprovádí žádnou explicitní akci-nemění, které uživatelské procedury jsou vyvolány.

Pokud je *MsgExitsDefined* nula, toto pole je ukazatel null.

Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

MsgExitsDefinováno (MQLONG)
Toto pole určuje počet uživatelských procedur zpráv kanálu definovaných v řetězu.

Je větší než nebo rovno nule.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

Počet MsgRetry(MQLONG)
Toto pole určuje, kolikrát se agent MCA pokusí vložit zprávu po prvním neúspěšném pokusu.

V tomto poli je uveden počet pokusů agenta MCA o operaci otevření nebo vložení v případě, že první operace MQOPEN nebo MQPUT selže s kódem dokončení MQCC_FAILED. Účinek tohoto atributu závisí na tom, zda je hodnota *MsgRetryExit* prázdná nebo neprázdná:

- Pokud je hodnota *MsgRetryExit* prázdná, atribut **MsgRetryCount** řídí, zda se agent MCA pokusí o opakování. Je-li hodnota atributu nula, neprovedou se žádné pokusy o opakování. Pokud je hodnota atributu větší než nula, pokusí se o opakování v intervalech daných atributem **MsgRetryInterval** .

  Opakované pokusy jsou prováděny pouze pro následující kódy příčiny:

  – MQRC_PAGESET_FULL
  – MQRC_PUT_INHIBITED
  – MQRC_Q_FULL

  V případě jiných kódů příčiny přechází agent MCA okamžitě k běžnému zpracování selhání bez opakování zprávy, která selhala.
- Pokud je hodnota *MsgRetryExit* neprázdná, atribut **MsgRetryCount** neovlivní agenta MCA; místo toho je to uživatelská procedura pro opakování zpráv, která určuje, kolikrát se pokus o zopakování provede a v jakých intervalech; uživatelská procedura se vyvolá, i když je atribut **MsgRetryCount** nulový.

Atribut **MsgRetryCount** je zpřístupněn uživatelské proceduře ve struktuře MQCD, ale uživatelská procedura, kterou není třeba respektovat, pokračuje bez omezení, dokud uživatelská procedura nevrátí hodnotu MQXCC_SUPPRESS_FUNCTION v poli *ExitResponse* MQCXP.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR.

Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_3.

MsgRetryKonec (MQCHARn)
Toto pole určuje název uživatelské procedury pro opakování zpráv kanálu.

Uživatelská procedura opakování zpráv je uživatelská procedura vyvolaná agentem MCA, když agent MCA obdrží kód dokončení MQCC_FAILED z volání MQOPEN nebo MQPUT. Účelem uživatelské procedury je určit časový interval, po který agent MCA čeká, než se znovu pokusí o operaci MQOPEN nebo MQPUT. Alternativně lze uživatelskou proceduru nastavit tak, aby nezkoušel operaci znovu.

Uživatelská procedura je vyvolána pro všechny kódy příčiny, které mají kód dokončení MQCC_FAILED- nastavení uživatelské procedury určuje, které kódy příčiny má agent MCA zkusit znovu, pro kolik pokusů a v jakých časových intervalech.

Nemá-li být operace dále zkoušena, provede agent MCA normální zpracování selhání; toto zpracování zahrnuje generování zprávy o výjimce (je-li určeno odesilatelem) a buď umístění původní zprávy do fronty nedoručených zpráv, nebo vyřazení zprávy (podle toho, zda odesilatel uvedl MQRO_DEAD_LETTER_Q nebo MQRO_DISCARD_MSG). Selhání týkající se fronty nedoručených zpráv (například zaplnění fronty nedoručených zpráv) nezpůsobí vyvolání uživatelské procedury opakování zprávy.

Není-li název uživatelské procedury prázdný, je uživatelská procedura volána v následujících časech:

- Bezprostředně před provedením čekání před opětovným pokusem o doručení zprávy
- Při inicializaci a ukončení kanálu

Popis obsahu tohoto pole v různých prostředích viz “MQCD-Definice kanálu” na stránce 1471 .

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR.

Délka tohoto pole je dána hodnotou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro dané prostředí.

Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_3.

MsgRetryInterval (MQLONG)
Toto pole určuje minimální interval v milisekundách, po kterém je operace otevření nebo vložení zopakována.

Účinek tohoto atributu závisí na tom, zda je hodnota *MsgRetryExit* prázdná nebo neprázdná:

- Pokud je hodnota *MsgRetryExit* prázdná, atribut **MsgRetryInterval** určuje minimální dobu, po kterou agent MCA čeká před zopakováním zprávy, pokud první operace MQOPEN nebo MQPUT selže s kódem dokončení MQCC_FAILED. Hodnota nula znamená, že opakování bude provedeno co nejdříve po předchozím pokusu. Opakování se provádí pouze v případě, že hodnota *MsgRetryCount* je větší než nula.

  Tento atribut se také používá jako doba čekání, pokud uživatelská procedura opakování zprávy vrátí neplatnou hodnotu v poli *MsgRetryInterval* v MQCXP.
- Není-li parametr *MsgRetryExit* prázdný, atribut **MsgRetryInterval** neovlivní agenta MCA; místo toho je to uživatelská procedura opakování zprávy, která určuje, jak dlouho agent MCA čeká. Atribut **MsgRetryInterval** je zpřístupněn uživatelské proceduře ve struktuře MQCD, ale uživatelská procedura jej nemusí respektovat.

Hodnota je v rozsahu 0 až 999 999 999 999.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR.

Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_3.

Následující pole v této struktuře nejsou přítomna, pokud je hodnota *Version* menší než MQCD_VERSION_4.

MsgRetryUserData (MQCHAR32)
Toto pole určuje uživatelská data uživatelské procedury pro opakování zpráv kanálu.

Tato data jsou předána uživatelské proceduře pro opakování zpráv kanálu v poli *ExitData* parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na počátku obsahuje data, která byla nastavena v definici kanálu. Během doby životnosti této instance MCA jsou však veškeré změny provedené v obsahu tohoto pole uživatelskou procedurou libovolného typu zachovány a viditelné pro následná vyvolání uživatelských procedur (bez ohledu na typ) pro tuto instanci MCA. Takové změny neovlivňují definici kanálu používanou jinými instancemi MCA. Lze použít libovolné znaky (včetně binárních dat).

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR.

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH. Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_3.

Toto pole není v produktu IBM MQ for IBM irelevantní.

MsgUserData (MQCHAR32)
Toto pole určuje uživatelská data uživatelské procedury pro zprávy kanálu.

Tato data jsou předána uživatelské proceduře pro zprávy kanálu v poli *ExitData* parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na počátku obsahuje data, která byla nastavena v definici kanálu. Během doby životnosti této instance MCA jsou však veškeré změny provedené v obsahu tohoto pole uživatelskou procedurou libovolného typu zachovány a viditelné pro následná vyvolání uživatelských procedur (bez ohledu na typ) pro tuto instanci MCA. Takové změny neovlivňují definici kanálu používanou jinými instancemi MCA. Lze použít libovolné znaky (včetně binárních dat).

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto pole není v produktu IBM MQ for IBM irelevantní.

MsgUserDataPtr (MQPTR)
Toto pole určuje adresu prvního pole *MsgUserData* .

Je-li hodnota *MsgExitsDefined* větší než nula, je tato adresa adresou seznamu položek uživatelských dat pro každou uživatelskou proceduru pro zprávu kanálu v řetězci.

Každá datová položka uživatele je v poli délky *ExitDataLength*, vyplněné vpravo mezerami. Vedle sebe se nacházejí pole *MsgExitsDefined* -jedno pro každou uživatelskou proceduru. Je-li počet definovaných položek uživatelských dat menší než počet názvů uživatelských procedur, jsou nedefinované položky uživatelských dat nastaveny na mezery. Naopak, je-li počet definovaných položek uživatelských dat větší než počet názvů uživatelských procedur, přebytečné položky uživatelských dat jsou ignorovány a nejsou prezentovány uživatelské proceduře.

Veškeré změny provedené v těchto hodnotách uživatelskou procedurou jsou zachovány. To umožňuje jedné uživatelské proceduře předávat informace jiné uživatelské proceduře. U změn se neprovádí žádné ověření, takže například binární data lze v případě potřeby zapsat do těchto polí.

Pokud je *MsgExitsDefined* nula, toto pole je ukazatel null.

Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

NetworkPriority (MQLONG)
Toto pole určuje prioritu síťového připojení pro daný kanál.

Je-li k dispozici více cest ke konkrétnímu cíli, je vybrána cesta s nejvyšší prioritou. Hodnota je v rozsahu 0 až 9; 0 je nejnižší priorita.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_5.

Následující pole v této struktuře nejsou přítomna, pokud je hodnota *Version* menší než MQCD_VERSION_6.

NonPersistentMsgSpeed (MQLONG)
Toto pole určuje rychlost, jakou přechodné zprávy procházejí kanálem.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_RECEIVER, MQCHT_REQUESTER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Hodnota je jedna z následujících:

MQNPMS_NORMAL normální
Normální rychlost.

Je-li kanál definován jako MQNPMS_NORMAL, přechodné zprávy procházejí kanálem normální rychlostí. To má tu výhodu, že tyto zprávy nejsou ztraceny, pokud dojde k selhání kanálu. Trvalé a přechodné zprávy ve stejné přenosové frontě také udržují své vzájemné pořadí.

MQNPMS_FAST
Rychlá rychlost.

Je-li kanál definován jako MQNPMS_FAST, přechodné zprávy procházejí kanálem vysokou rychlostí. To zlepšuje propustnost kanálu, ale znamená to, že přechodné zprávy jsou ztraceny, pokud dojde k selhání kanálu. Také je možné, aby přechodné zprávy přeskakovaly před trvalými zprávami čekajícími ve stejné přenosové frontě, to znamená, že pořadí přechodných zpráv není udržováno vzhledem k trvalým zprávám. Pořadí dočasných zpráv, které jsou vzájemně relativní, je však zachováno. Podobně je udržováno pořadí trvalých zpráv relativně k sobě navzájem.

Heslo (MQCHAR12)
Toto pole určuje heslo používané agentem kanálu zpráv při pokusu o zahájení zabezpečené relace SNA s agentem vzdáleného kanálu zpráv.

Toto pole může být neprázdné pouze v systému AIX, Linux, and Windowsa je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER nebo MQCHT_CLNTCONN. V systému z/OSnení toto pole relevantní.

Délka tohoto pole je dána hodnotou MQ_PASSWORD_LENGTH. Použije se však pouze prvních 10 znaků.

Toto pole není k dispozici, pokud je hodnota *Version* menší než hodnota MQCD_VERSION_2.

PropertyControl (MQLONG)
Toto pole určuje, co se stane s vlastnostmi zpráv v případě, že má být zpráva odeslána správci front V6 nebo předchozímu správci front (správci front, který nerozumí konceptu deskriptoru vlastností).

Hodnota může být některá z následujících:

MQPROP_KOMPATIBILITA
Pokud zpráva obsahuje vlastnost s předponou **mcd.**, **jms.**, **usr.** nebo **mqext.**, jsou všechny vlastnosti zprávy doručeny aplikaci v záhlaví MQRFH2 . Jinak jsou všechny vlastnosti zprávy, s výjimkou vlastností obsažených v deskriptoru zprávy (nebo rozšíření), vyřazeny a pro aplikaci již nejsou přístupné.

Tato hodnota je výchozí hodnota. Umožňuje aplikacím, které očekávají, že vlastnosti související s produktem JMSbudou v záhlaví MQRFH2 v datech zprávy, pokračovat v práci bez úprav.

MQPROP_NONE
Před odesláním zprávy vzdálenému správci front budou ze zprávy odebrány všechny vlastnosti zprávy kromě vlastností nacházejících se v deskriptoru zprávy (či rozšíření).

MQPROP_ALL
Všechny vlastnosti zprávy jsou zahrnuty do zprávy při jejím odeslání vzdálenému správci front. Vlastnosti, s výjimkou vlastností obsažených v deskriptoru (či rozšíření) zprávy, budou umístěny v jednom nebo několika záhlavích v datech zprávy.

Tento atribut lze použít pro kanály odesilatele, serveru, odesilatele klastru a příjemce klastru.

"MQIA_ * (Celočíselné selektory atributů)" na stránce 129
"MQPROP_ * (řídicí hodnoty vlastností fronty a kanálu a maximální délka vlastností)" na stránce 169

PutAuthority (MQLONG)
Toto pole uvádí, zda se identifikátor uživatele v informacích o kontextu přidružených ke zprávě použije k zavedení oprávnění pro vložení zprávy do cílové fronty.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR. Jedná se o jednu z následujících položek:

VÝCHOZÍ
Použije se výchozí identifikátor uživatele.

MQPA_CONTEXT
Použije se identifikátor uživatele kontextu.

MQPA_ALTERNATE_OR_MCA
Použije se ID uživatele z pole UserIdentifier deskriptoru zprávy. Není použito žádné ID uživatele přijaté ze sítě. Tato hodnota je podporována pouze v systému z/OS.

MQPA_ONLY_MCA
Použije se výchozí ID uživatele. Není použito žádné ID uživatele přijaté ze sítě. Tato hodnota je podporována pouze v systému z/OS.

QMgrName (MQCHAR48)
Toto pole určuje název správce front, ke kterému se může uživatelská procedura připojit.

Pro kanály s jinou hodnotou *ChannelType* než MQCHT_CLNTCONN je toto pole název správce front, ke kterému se může uživatelská procedura připojit, což v systému AIX, Linux, and Windowsje vždy neprázdné.

Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

ReceiveExit (MQCHARn)
Toto pole určuje název uživatelské procedury pro příjem kanálu.

Není-li tento název prázdný, je uživatelská procedura volána v následujících časech:

- Bezprostředně před zpracováním přijatých síťových dat.

  Uživatelské proceduře je poskytnuta úplná přenosová vyrovnávací paměť jako přijatá. Obsah vyrovnávací paměti lze upravit podle potřeby.
- Při inicializaci a ukončení kanálu.

Popis obsahu tohoto pole v různých prostředích viz "MQCD-Definice kanálu" na stránce 1471 .

Délka tohoto pole je dána hodnotou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro dané prostředí.

ReceiveExitPtr (MQPTR)
Toto pole určuje adresu prvního pole *ReceiveExit* .

Je-li hodnota *ReceiveExitsDefined* větší než nula, je tato adresa adresou seznamu názvů jednotlivých uživatelských procedur pro příjem kanálu v řetězci.

Každý název je v poli o délce *ExitNameLength*, zprava vyplněný mezerami. Vedle sebe se nacházejí pole *ReceiveExitsDefined* -jedno pro každou uživatelskou proceduru.

Všechny změny provedené v těchto názvech uživatelskou procedurou jsou zachovány, ačkoli uživatelská procedura kanálu zpráv neprovádí žádnou explicitní akci-nemění, které uživatelské procedury jsou vyvolány.

Pokud je *ReceiveExitsDefined* nula, toto pole je ukazatel null.

Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

ReceiveExitsDefinováno (MQLONG)
Toto pole určuje počet uživatelských procedur pro příjem kanálu definovaných v řetězci.

Je větší než nebo rovno nule.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

Data ReceiveUser(MQCHAR32)
Tento kanál určuje uživatelská data uživatelské procedury pro příjem kanálu.

Tato data jsou předána uživatelské proceduře pro příjem kanálu v poli *ExitData* parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na počátku obsahuje data, která byla nastavena v definici kanálu. Během doby životnosti této instance MCA jsou však veškeré změny provedené v obsahu tohoto pole uživatelskou procedurou libovolného typu zachovány a viditelné pro následná vyvolání uživatelských procedur (bez ohledu na typ) pro tuto instanci MCA. To platí pro východy na různých konverzacích. Takové změny neovlivňují definici kanálu používanou jinými instancemi MCA. Lze použít libovolné znaky (včetně binárních dat).

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto pole není v produktu IBM MQ for IBM irelevantní.

Následující pole v této struktuře nejsou přítomna, pokud je hodnota *Version* menší než MQCD_VERSION_2.

ReceiveUserDataPtr (MQPTR)
Toto pole určuje adresu prvního pole *ReceiveUserData* .

Je-li hodnota *ReceiveExitsDefined* větší než nula, je tato adresa adresou seznamu položek uživatelských dat pro každou uživatelskou proceduru pro příjem kanálu v řetězci.

Každá datová položka uživatele je v poli délky *ExitDataLength*, vyplněné vpravo mezerami. Vedle sebe se nacházejí pole *ReceiveExitsDefined* -jedno pro každou uživatelskou proceduru. Je-li počet definovaných položek uživatelských dat menší než počet názvů uživatelských procedur, jsou nedefinované položky uživatelských dat nastaveny na mezery. Naopak, je-li počet definovaných položek uživatelských dat větší než počet názvů uživatelských procedur, přebytečné položky uživatelských dat jsou ignorovány a nejsou prezentovány uživatelské proceduře.

Veškeré změny provedené v těchto hodnotách uživatelskou procedurou jsou zachovány. To umožňuje jedné uživatelské proceduře předávat informace jiné uživatelské proceduře. U změn se neprovádí žádné ověření, takže například binární data lze v případě potřeby zapsat do těchto polí.

Pokud je *ReceiveExitsDefined* nula, toto pole je ukazatel null.

Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

Následující pole v této struktuře nejsou přítomna, pokud je hodnota *Version* menší než MQCD_VERSION_5.

RemotePassword (MQCHAR12)
Toto pole určuje heslo od partnera.

Toto pole obsahuje platné informace pouze v případě, že *ChannelType* je MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

- Pro uživatelskou proceduru zabezpečení zprávy v kanálu MQCHT_CLNTCONN je toto heslo heslo, které bylo získáno z prostředí. Uživatelská procedura může zvolit odeslání do uživatelské procedury zabezpečení na serveru.
- V případě uživatelské procedury zabezpečení v kanálu MQCHT_SVRCONN může toto pole obsahovat heslo, které bylo získáno z prostředí klienta, pokud neexistuje žádná uživatelská procedura zabezpečení klienta. Uživatelská procedura může použít toto heslo k ověření identifikátoru uživatele v souboru *RemoteUserIdentifier*.

  Existuje-li v klientu uživatelská procedura zabezpečení, lze tyto informace získat v toku zabezpečení z klienta.

Délka tohoto pole je dána hodnotou MQ_PASSWORD_LENGTH. Toto pole není k dispozici, pokud je hodnota *Version* menší než hodnota MQCD_VERSION_2.

RemoteSecurityID (MQBYTE40)
Toto pole určuje identifikátor zabezpečení pro vzdáleného uživatele.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

Následující speciální hodnota označuje, že neexistuje žádný identifikátor zabezpečení:

MQSID_NONE
Nebyl uveden žádný identifikátor zabezpečení.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQSID_NONE_ARRAY; tato konstanta má stejnou hodnotu jako MQSID_NONE, ale je polem znaků místo řetězce.

Toto je vstupní pole pro ukončení. Délka tohoto pole je dána hodnotou MQ_SECURITY_ID_LENGTH. Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_6.

Následující pole v této struktuře nejsou přítomna, pokud je hodnota *Version* menší než MQCD_VERSION_7.

Identifikátor RemoteUser(MQCHAR12)
Toto pole uvádí prvních 12 bajtů identifikátoru uživatele od partnera.

Existují dvě pole, která obsahují identifikátor vzdáleného uživatele:

- *RemoteUserIdentifier* obsahuje prvních 12 bajtů identifikátoru vzdáleného uživatele a je vyplněn mezerami, pokud je identifikátor kratší než 12 bajtů. *RemoteUserIdentifier* může být prázdné.
- *LongRemoteUserIdPtr* ukazuje na úplný identifikátor vzdáleného uživatele, který může být delší než 12 bajtů. Jeho délka je dána *LongRemoteUserIdLength*. Úplný identifikátor neobsahuje žádné koncové mezery a není ukončen s hodnotou null. Pokud je identifikátor prázdný, *LongRemoteUserIdLength* je nula a hodnota *LongRemoteUserIdPtr* není definována.

  Parametr *LongRemoteUserIdPtr* není přítomen, pokud je hodnota *Version* menší než hodnota MQCD_VERSION_6.

Identifikátor vzdáleného uživatele je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

- Pro uživatelskou proceduru pro zabezpečení zprávy v kanálu MQCHT_CLNTCONN je tato hodnota identifikátor uživatele, který byl získán z prostředí. Uživatelská procedura může zvolit odeslání do uživatelské procedury zabezpečení na serveru.
- V případě uživatelské procedury zabezpečení v kanálu MQCHT_SVRCONN může toto pole obsahovat identifikátor uživatele, který byl získán z prostředí klienta, pokud neexistuje žádná uživatelská procedura zabezpečení klienta. Uživatelská procedura může ověřit toto ID uživatele (pravděpodobně s heslem v souboru *RemotePassword*) a aktualizovat hodnotu v souboru *MCAUserIdentifier*.

  Existuje-li v klientu uživatelská procedura zabezpečení, lze tyto informace získat v toku zabezpečení z klienta.

Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Toto pole není k dispozici, pokud je hodnota *Version* menší než hodnota MQCD_VERSION_2.

SecurityExit (MQCHARn)
Toto pole určuje název uživatelské procedury pro zabezpečení zprávy kanálu.

Není-li tento název prázdný, je uživatelská procedura volána v následujících časech:

- Okamžitě po zavedení kanálu.

  Před přenosem jakékoli zprávy je ukončení poskytnuta možnost podnítit toky zabezpečení k potvrzení autorizace připojení.
- Po přijetí odezvy na tok zpráv zabezpečení.

  Všechny toky zpráv zabezpečení přijaté ze vzdáleného procesoru na vzdáleném počítači jsou předávány uživatelské proceduře.
- Při inicializaci a ukončení kanálu.

Popis obsahu tohoto pole v různých prostředích viz “MQCD-Definice kanálu” na stránce 1471 .

Délka tohoto pole je dána hodnotou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro dané prostředí.

Data SecurityUser(MQCHAR32)
Tento kanál určuje uživatelská data uživatelské procedury zabezpečení kanálu.

Tato data jsou předána uživatelské proceduře pro zabezpečení zprávy kanálu v poli *ExitData* parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na počátku obsahuje data, která byla nastavena v definici kanálu. Během doby životnosti této instance MCA jsou však veškeré změny provedené v obsahu tohoto pole uživatelskou procedurou libovolného typu zachovány a viditelné pro následná vyvolání uživatelských procedur (bez ohledu na typ) pro tuto instanci MCA. To platí pro východy na různých konverzacích. Tyto změny nemají vliv na definici kanálu používanou jinými instancemi MCA. Lze použít libovolné znaky (včetně binárních dat).

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto pole není v produktu IBM MQ for IBM irelevantní.

SendExit (MQCHARn)
Toto pole určuje název uživatelské procedury pro odesílání kanálu.

Není-li tento název prázdný, je uživatelská procedura volána v následujících časech:

- Bezprostředně před odesláním dat do sítě.

  Uživatelské proceduře je před přenosem předána úplná přenosová vyrovnávací paměť. Obsah vyrovnávací paměti lze upravit podle potřeby.
- Při inicializaci a ukončení kanálu.

Popis obsahu tohoto pole v různých prostředích viz “MQCD-Definice kanálu” na stránce 1471 .

Délka tohoto pole je dána hodnotou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro dané prostředí.

SendExitPtr (MQPTR)
Toto pole určuje adresu prvního pole *SendExit* .

Je-li hodnota *SendExitsDefined* větší než nula, je tato adresa adresou seznamu názvů jednotlivých uživatelských procedur pro odeslání zprávy kanálu v řetězci.

Každý název je v poli o délce *ExitNameLength*, zprava vyplněný mezerami. Vedle sebe se nacházejí pole *SendExitsDefined* -jedno pro každou uživatelskou proceduru.

Všechny změny provedené v těchto názvech uživatelskou procedurou jsou zachovány, přestože uživatelská procedura pro odesílání zpráv neprovádí žádnou explicitní akci-nemění, které uživatelské procedury jsou vyvolány.

Pokud je *SendExitsDefined* nula, toto pole je ukazatel null.

Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

SendExitsDefinováno (MQLONG)
Toto pole určuje počet uživatelských procedur pro odeslání kanálu definovaných v řetězci.

Je větší než nebo rovno nule.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

Data SendUser(MQCHAR32)
Toto pole určuje uživatelská data uživatelské procedury odeslání zprávy kanálu.

Tato data jsou předána uživatelské proceduře kanálu v poli *ExitData* parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na počátku obsahuje data, která byla nastavena v definici kanálu. Během doby životnosti této instance MCA jsou však veškeré změny provedené v obsahu tohoto pole uživatelskou procedurou libovolného typu zachovány a viditelné pro následná vyvolání uživatelských procedur (bez ohledu na typ) pro tuto instanci MCA. To platí pro východy na různých konverzacích. Takové změny neovlivňují definici kanálu používanou jinými instancemi MCA. Lze použít libovolné znaky (včetně binárních dat).

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto pole není v produktu IBM MQ for IBM irelevantní.

SendUserDataPtr (MQPTR)
Toto pole uvádí adresu pole *SendUserData* .

Je-li hodnota *SendExitsDefined* větší než nula, je tato adresa adresou seznamu položek uživatelských dat pro každou uživatelskou proceduru pro zprávu kanálu v řetězci.

Každá datová položka uživatele je v poli délky *ExitDataLength*, vyplněné vpravo mezerami. Vedle sebe se nacházejí pole *MsgExitsDefined* -jedno pro každou uživatelskou proceduru. Je-li počet definovaných položek uživatelských dat menší než počet názvů uživatelských procedur, jsou nedefinované položky uživatelských dat nastaveny na mezery. Naopak, je-li počet definovaných položek uživatelských dat větší než počet názvů uživatelských procedur, přebytečné položky uživatelských dat jsou ignorovány a nejsou prezentovány uživatelské proceduře.

Veškeré změny provedené v těchto hodnotách uživatelskou procedurou jsou zachovány. To umožňuje jedné uživatelské proceduře předávat informace jiné uživatelské proceduře. U změn se neprovádí žádné ověření, takže například binární data lze v případě potřeby zapsat do těchto polí.

Pokud je *SendExitsDefined* nula, toto pole je ukazatel null.

Na platformách, kde programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec odpovídající délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

SeqNumberZalamovat (MQLONG)
Toto pole určuje nejvyšší přípustné pořadové číslo zprávy.

Když je tato hodnota dosažena, pořadová čísla se zalomí, aby začala znovu od 1.

Tato hodnota je nepřevoditelná a musí se shodovat v definici lokálního i vzdáleného kanálu.

Toto pole není relevantní pro kanály s hodnotou *ChannelType* MQCHT_SVRCONN nebo MQCHT_CLNTCONN.

SharingConversations (MQLONG)
Toto pole určuje maximální počet konverzací, které mohou sdílet instanci kanálu přidruženou k tomuto kanálu.

Toto pole se používá u kanálů připojení klienta a připojení serveru.

Hodnota 0 znamená, že kanál pracuje stejně jako ve verzích starších než IBM WebSphere MQ 7.0 s ohledem na následující atributy:

- Sdílení konverzace
- Dopředné čtení
- STOP CHANNEL(*channelname*) MODE(QUIESCE)
- Synchronizační signály
- Asynchronní spotřeba klienta

Hodnota 1 je minimální hodnota pro chování IBM MQ . Ačkoli je v instanci kanálu povolena pouze jedna konverzace, je k dispozici funkce dopředného čtení, asynchronní spotřeby a chování synchronizačního a klidového zastavení kanálu CLNTCONN-SVRCONN .

Toto je vstupní pole pro ukončení. Není přítomen, pokud je hodnota *Version* menší než MQCD_VERSION_9.

Výchozí hodnota tohoto pole je 10.

Poznámka: Omezení *MaxInstances* a *MaxInstancesPerClient* použitá na kanál omezují počet instancí kanálu, nikoli počet konverzací, které mohou tyto instance sdílet.

ShortConnectionNázev (MQCHAR20)
Toto pole určuje prvních 20 bajtů názvu připojení.

Pokud je pole *Version* MQCD_VERSION_1, *ShortConnectionName* obsahuje úplný název připojení.

Je-li pole *Version* MQCD_VERSION_2 nebo vyšší, obsahuje *ShortConnectionName* prvních 20 znaků názvu připojení. Úplný název připojení je dán polem *ConnectionName* ; *ShortConnectionName* a prvních 20 znaků *ConnectionName* jsou identické.

Podrobnosti o obsahu tohoto pole viz *ConnectionName* .

Poznámka: Název tohoto pole byl změněn pro MQCD_VERSION_2 a následné verze MQCD; pole bylo dříve nazýváno *ConnectionName*.

Délka tohoto pole je dána hodnotou MQ_SHORT_CONN_NAME_LENGTH.

Počet ShortRetry(MQLONG)
Toto pole určuje maximální počet pokusů o připojení ke vzdálenému počítači.

Toto pole je maximální počet pokusů o připojení ke vzdálenému počítači v intervalech určených parametrem *ShortRetryInterval*, než se použijí (obvykle delší) hodnoty *LongRetryCount* a *LongRetryInterval* .

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

ShortRetry(MQLONG)
Toto pole určuje maximální počet sekund čekání před opětovným pokusem o připojení ke vzdálenému počítači.

Interval mezi opakovanými pokusy může být prodloužen, pokud má kanál čekat na aktivaci.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

z/OS *SPLProtection (MQLONG)*
Toto pole určuje hodnotu ochrany zásad zabezpečení AMS .

Hodnota je jedna z následujících:

MQSPL_PASSTHRU
Projděte, beze změny, všechny zprávy odeslané nebo přijaté agentem MCA pro tento kanál.

Tato hodnota je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_RECEIVER nebo MQCHT_REQUESTER a je výchozí hodnotou.

MQSPL_REMOVE
Odeberte jakoukoli ochranu systému AMS ze zpráv načtených z přenosové fronty agentem MCA a odešlete zprávy partnerovi.

Tato hodnota je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER nebo MQCHT_SERVER.

MQSPL_ASPOLICY
Na základě zásady definované pro cílovou frontu se uplatní ochrana AMS na příchozí zprávy před jejich vložením do cílové fronty.

Tato hodnota je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_RECEIVER nebo MQCHT_REQUESTER.

Toto je vstupní pole pro ukončení. Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_12.

SSLCipherSpec (MQCHAR32)
Toto pole uvádí specifikaci šifrování, která se používá při použití TLS.

Je-li volba SSLCipherSpec prázdná, kanál nepoužívá protokol TLS. Není-li pole prázdné, obsahuje toto pole řetězec určující používanou specifikaci CipherSpec .

Tento parametr je platný pro všechny typy kanálů. Je podporován na následujících platformách:

- AIX AIX
- IBM i IBM i
- Linux Linux
- Windows Windows
- z/OS z/OS

Je platný pouze pro typy kanálů typu transportu (TRPTYPE) protokolu TCP.

Toto je vstupní pole pro ukončení. Délka tohoto pole je dána hodnotou MQ_SSL_CIPHER_SPEC_LENGTH. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_7.

SSLClientAuth (MQLONG)
Toto pole určuje, zda je vyžadováno ověření klienta TLS.

Toto pole je relevantní pouze pro definice kanálu SVRCONN.

Jedná se o jednu z následujících hodnot:

MQSCA_REQUIRED
Je vyžadováno ověření klienta.

MQSCA_OPTIONAL
Ověření klienta je volitelné.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_7.

SSLPeerNameDélka (MQLONG)
Toto pole určuje délku názvu partnera TLS v bajtech, na který odkazuje *SSLPeerNamePtr*.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_7.

SSLPeerNamePtr (MQPTR)
Toto pole určuje adresu názvu partnera TLS.

Když je během úspěšného navázání komunikace TLS přijat certifikát, je rozlišující název subjektu certifikátu zkopírován do pole MQCD, ke kterému přistupuje SSLPeerNamePtr na konci kanálu, který přijímá certifikát. Přepíše hodnotu SSLPeerName pro kanál, pokud je tato hodnota přítomna v definici kanálu lokálního uživatele. Je-li na tomto konci kanálu zadána uživatelská procedura pro zabezpečení zprávy, obdrží rozlišující název z certifikátu typu peer v tabulce MQCD.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_7.

Poznámka: Aplikace uživatelské procedury zabezpečení vytvořené před vydáním produktu IBM WebSphere MQ 7.1 mohou vyžadovat aktualizaci. Další informace naleznete v tématu Programy uživatelských procedur pro zabezpečení kanálu.

StrucLength (MQLONG)
Toto pole určuje délku struktury MQCD v bajtech.

Délka nezahrnuje žádný z řetězců adresovaných poli ukazatele obsaženými v rámci struktury. Hodnota je jedna z následujících:

MQCD_LENGTH_4
Délka struktury definice kanálu version-4 .

MQCD_LENGTH_5
Délka struktury definice kanálu version-5 .

MQCD_LENGTH_6
Délka struktury definice kanálu version-6 .

MQCD_LENGTH_7
Délka struktury definice kanálu version-7 .

MQCD_LENGTH_8
Délka struktury definice kanálu version-8 .

MQCD_LENGTH_9
Délka struktury definice kanálu version-9 .

MQCD_LENGTH_10
Délka struktury definice kanálu version-10 .

MQCD_LENGTH_11
Délka struktury definice kanálu version-11 .

z/OS **MQCD_LENGTH_12**
Délka struktury definice kanálu version-12 .

Následující konstanta určuje délku aktuální verze:

MQCD_CURRENT_LENGTH
Délka aktuální verze struktury definice kanálu.

Poznámka: Tyto konstanty mají hodnoty specifické pro dané prostředí.

Pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_4.

TpName (MQCHAR64)
Toto pole určuje název transakčního programu LU 6.2 .

Toto pole je relevantní pouze v případě, že přenosový protokol (*TransportType*) je MQXPT_LU62a *ChannelType* není MQCHT_SVRCONN nebo MQCHT_RECEIVER.

Toto pole je vždy prázdné na platformách, na kterých jsou informace obsaženy v objektu Side komunikace.

Délka tohoto pole je dána hodnotou MQ_TP_NAME_LENGTH.

TransportType (MQLONG)
Toto pole uvádí přenosový protokol, který se má použít.

Hodnota není kontrolována, pokud byl kanál iniciován z druhého konce.

Jedná se o jednu z následujících hodnot:

MQXPT_LU62
Přenosový protokol LU 6.2 .

MQXPT_TCP
Přenosový protokol TCP/IP.

MQXPT_NETBIOS
Přenosový protokol NetBIOS .

Tato hodnota je podporována v následujících prostředích: Windows.

MQXPT_SPX
Přenosový protokol SPX.

Tato hodnota je podporována v následujících prostředích: Windowsplus IBM MQ klienti připojení k těmto systémům.

UseDLQ (MQLONG)
Toto pole uvádí, zda se použije fronta nedoručených zpráv (nebo nedoručená fronta zpráv), když zprávy nemohou být doručeny kanály.

Může obsahovat jednu z následujících hodnot:

MQUSEDLQ_NO
Zprávy, které nelze doručit prostřednictvím kanálu, jsou považovány za selhání. Kanál buď zruší zprávu, nebo ukončí kanál v souladu s nastavením NPMSPEED.

MQUSEDLQ_YES
Když atribut správce front DEADQ poskytuje název fronty nedoručených zpráv, použije se, jinak je chování stejné jako pro NO. Hodnota YES je výchozí hodnota.

UserIdentifier (MQCHAR12)
Toto pole určuje identifikátor uživatele, který používá agent kanálu zpráv při pokusu o zahájení zabezpečené relace SNA se vzdáleným agentem kanálu zpráv.

Toto pole může být neprázdné pouze v systému AIX, Linux, and Windowsa je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER nebo MQCHT_CLNTCONN. V systému z/OSnení toto pole relevantní.

Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Použije se však pouze prvních 10 znaků.

Toto pole není přítomno, pokud je hodnota *Version* menší než MQCD_VERSION_2.

Verze (MQLONG)
Pole Version uvádí nejvyšší číslo verze, kterou můžete nastavit pro strukturu.

Hodnota závisí na prostředí:

MQCD _VERSION_1
Struktura definice kanálu verze 1.

MQCD _VERSION_2
Struktura definice kanálu verze 2.

MQCD _VERSION_3
Struktura definice kanálu verze 3.

MQCD _VERSION_4
Struktura definice kanálu verze 4.

MQCD _VERSION_5
Struktura definice kanálu verze 5.

MQCD _VERSION_6
Struktura definice kanálu verze 6.

MQCD _VERSION_7
Struktura definice kanálu verze 7.

MQCD _VERSION_8
Struktura definice kanálu verze 8.

MQCD _VERSION_9
Struktura definice kanálu verze 9.

MQCD _VERSION_10
Struktura definice kanálu verze 10.

MQCD _VERSION_11
Struktura definice kanálu verze 11.

Verze 11 je nejvyšší, kterou můžete nastavit pole na IBM MQ 8.0 na všech platformách.

z/OS **MQCD _VERSION_12**
Struktura definice kanálu verze 12.

Verze 12 je nejvyšší, kterou můžete nastavit pole na hodnotu IBM MQ 9.1.3.

Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MQCD_CURRENT_VERSION
Hodnota nastavená v souboru MQCD_CURRENT_VERSION je aktuální verze použité struktury definice kanálu.

Hodnota MQCD_CURRENT_VERSION závisí na prostředí. Obsahuje nejvyšší hodnotu podporovanou platformou.

MQCD_CURRENT_VERSION se nepoužívá k inicializaci výchozích struktur poskytnutých v záhlaví, kopírování a zahrnutí souborů poskytnutých pro různé programovací jazyky. Výchozí inicializace produktu Version závisí na platformě a vydání.

Deklarace MQCD v záhlaví, kopii a souborech začlenění jsou inicializovány na hodnotu MQCD_VERSION_6. Chcete-li použít další pole MQCD , aplikace musí nastavit číslo verze na

MQCD_CURRENT_VERSION. Pokud píšete aplikaci, která je přenosná mezi několika prostředími, musíte zvolit verzi, která je podporována ve všech prostředích.

Tip: Když je zavedena nová verze struktury MQCD , rozvržení existující části se nezmění. Uživatelská procedura musí zkontrolovat číslo verze. Musí být větší nebo rovna nejnižší verzi, která obsahuje pole, která uživatelská procedura potřebuje použít.

XmitQName (MQCHAR48)
Toto pole uvádí název přenosové fronty, ze které se načítají zprávy.

Toto pole je relevantní pouze pro kanály s hodnotou *ChannelType* MQCHT_SENDER nebo MQCHT_SERVER.

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH.

C prohlášení

Tato deklarace je deklarací jazyka C pro strukturu 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];      /* TLS CipherSpec */
  MQPTR     SSLPeerNamePtr;         /* Address of TLS peer name */
  MQLONG    SSLPeerNameLength;      /* Length of TLS peer name */
  MQLONG    SSLClientAuth;          /* Whether TLS client */
                                    /* authentication is required */
  MQLONG    KeepAliveInterval;      /* Keepalive interval */
  MQCHAR    LocalAddress[48];       /* Local communications */
                                    /* address */
  MQLONG    BatchHeartbeat;         /* Batch heartbeat interval */
  /* Ver:7 */
  MQLONG    HdrCompList[2];         /* Header data compression */
                                    /* list */
  MQLONG    MsgCompList[16];        /* Message data compression */
                                    /* list */
  MQLONG    CLWLChannelRank;        /* Channel rank */
  MQLONG    CLWLChannelPriority;    /* Channel priority */
  MQLONG    CLWLChannelWeight;      /* Channel weight */
  MQLONG    ChannelMonitoring;      /* Channel monitoring */
  MQLONG    ChannelStatistics;      /* Channel statistics */
  /* Ver:8 */
  MQLONG    SharingConversations;   /* Limit on sharing */
                                    /* conversations */
  MQLONG    PropertyControl;        /* Message property control */
  MQLONG    MaxInstances;           /* Limit on SVRCONN channel */
                                    /* instances */
  MQLONG    MaxInstancesPerClient;  /* Limit on SVRCONN channel */
                                    /* instances per client */
  MQLONG    ClientChannelWeight;    /* Client channel weight */
  MQLONG    ConnectionAffinity;     /* Connection affinity */
```

```
   /* Ver:9 */
   MQLONG    BatchDataLimit;           /* Batch data limit */
   MQLONG    UseDLQ;                   /* Use Dead Letter Queue */
   MQLONG    DefReconnect;             /* Default client reconnect */
                                       /* option */
   /* Ver:10 */
   MQCHAR64  CertificateLabel;         /* Certificate label */
 /* Ver:11 */
   MQLONG    SPLProtection             /* AMS Security policy protection */
 /* Ver:12 */
};
```

Deklarace jazyka COBOL

Tato deklarace je deklarací jazyka COBOL pro strukturu 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 **
      ** TLS CipherSpec
       15  MQCD-SSLCIPHERSPEC PIC X(32).
      ** Address of TLS peer name
       15  MQCD-SSLPEERNAMEPTR POINTER.
      ** Length of TLS peer name
       15  MQCD-SSLPEERNAMELENGTH PIC S9(9) BINARY.
      ** Whether TLS client authentication is required
```

```
       15  MQCD-SSLCLIENTAUTH PIC S9(9) BINARY.
      ** Keepalive interval
       15  MQCD-KEEPALIVEINTERVAL PIC S9(9) BINARY.
      ** Local communications address
       15  MQCD-LOCALADDRESS PIC X(48).
      ** Batch heartbeat interval
       15  MQCD-BATCHHEARTBEAT PIC S9(9) BINARY.
      ** Ver:7 **
      ** Header data compression list
       15  MQCD-HDRCOMPLIST PIC S9(9) BINARY.
      ** Message data compression list
       15  MQCD-MSGCOMPLIST PIC S9(9) BINARY.
      ** Channel rank
       15  MQCD-CLWLCHANNELRANK PIC S9(9) BINARY.
      ** Channel priority
       15  MQCD-CLWLCHANNELPRIORITY PIC S9(9) BINARY.
      ** Channel weight
       15  MQCD-CLWLCHANNELWEIGHT PIC S9(9) BINARY.
      ** Channel monitoring
       15  MQCD-CHANNELMONITORING PIC S9(9) BINARY.
      ** Channel statistics
       15  MQCD-CHANNELSTATISTICS PIC S9(9) BINARY.
      ** Ver:8 **
      ** Limit on sharing conversations
       15  MQCD-SHARINGCONVERSATIONS PIC S9(9) BINARY.
      ** Message property control
       15  MQCD-PROPERTYCONTROL PIC S9(9) BINARY.
      ** Limit on SVRCONN channel instances
       15  MQCD-MAXINSTANCES PIC S9(9) BINARY.
      ** Limit on SVRCONN channel instances per client
       15  MQCD-MAXINSTANCESPERCLIENT PIC S9(9) BINARY.
      ** Client channel weight
       15  MQCD-CLIENTCHANNELWEIGHT PIC S9(9) BINARY.
      ** Connection affinity
       15  MQCD-CONNECTIONAFFINITY PIC S9(9) BINARY.
      ** Ver:9 **
      ** Batch data limit
       15  MQCD-BATCHDATALIMIT PIC S9(9) BINARY.
      ** Use Dead Letter Queue
       15  MQCD-USEDLQ PIC S9(9) BINARY.
      ** Default client reconnect option
       15  MQCD-DEFRECONNECT PIC S9(9) BINARY.
      ** Ver:10 **
      ** Certificate Label
       15 MQCD-CERTLABL PIC X (64)
      ** Ver:11 **
      ** AMS Security policy protection
       15 MQCD-SPLPROTECTION PIC S9(9) BINARY
      ** Ver:12 **
```

Deklarace RPG (ILE)

Tato deklarace je deklarací RPG pro strukturu 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* TLS CipherSpec
D  CDSCS                1217   1248
D* Address of TLS peer name
D  CDSPN                1249   1264*
D* Length of TLS peer name
D  CDSPL                1265   1268I 0
D* Whether TLS client authentication is required
D  CDSCA                1269   1272I 0
D* Keepalive interval
D  CDKAI                1273   1276I 0
D* Local communications address
D  CDLOA                1277   1324
D* Batch heartbeat interval
D  CDBHB                1325   1328I 0
D* Ver:7 **
D* Header data compression list
D  CDHCL0
D  CDHCL1               1329   1332I 0
D  CDHCL2               1333   1336I 0
D  CDHCL                         10I 0 DIM(2) OVERLAY(CDHCL0)
D* Message data compression list
D  CDMCL0
D  CDMCL1               1337   1340I 0
D  CDMCL2               1341   1344I 0
D  CDMCL3               1345   1348I 0
D  CDMCL4               1349   1352I 0
D  CDMCL5               1353   1356I 0
D  CDMCL6               1357   1360I 0
D  CDMCL7               1361   1364I 0
D  CDMCL8               1365   1368I 0
D  CDMCL9               1369   1372I 0
D  CDMCL10              1373   1376I 0
D  CDMCL11              1377   1380I 0
D  CDMCL12              1381   1384I 0
D  CDMCL13              1385   1388I 0
D  CDMCL14              1389   1392I 0
D  CDMCL15              1393   1396I 0
D  CDMCL16              1397   1400I 0
D  CDMCL                         10I 0 DIM(16) OVERLAY(CDMCL0)
D* Channel rank
D  CDCWCR               1401   1404I 0
D* Channel priority
D  CDCWCP               1405   1408I 0
D* Channel weight
D  CDCWCW               1409   1412I 0
D* Channel monitoring
D  CDCHLMON             1413   1416I 0
D* Channel statistics
D  CDCHLST              1417   1420I 0
D* Ver:8 **
```

```
     D* Limit on sharing conversations
     D  CDSHC               1421   1424I 0
     D* Message property control
     D  CDPRC               1425   1428I 0
     D* Limit on SVRCONN channel instances
     D  CDMXIN              1429   1432I 0
     D* Limit on SVRCONN channel instances per client
     D  CDMXIC              1433   1436I 0
     D* Client channel weight
     D  CDCLNCHLW           1437   1440I 0
     D* Connection affinity
     D  CDCONNAFF           1441   1444I 0
     D* Ver:9 **
     D* Batch data limit
     D  CDBDL               1445   1448I 0
     D* Use Dead Letter Queue
     D  CDUDLQ              1449   1452I 0
     D* Default client reconnect option
     D  CDDRCN              1453   1456I 0
     D* Ver:10 **
```

System/390 deklarace modulu asembleru

Tato deklarace je deklarací modulu sestavení System/390 pro strukturu 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   TLS CipherSpec
MQCD_SSLPEERNAMEPTR          DS   F      Address of TLS peer name
MQCD_SSLPEERNAMELENGTH       DS   F      Length of TLS peer name
MQCD_SSLCLIENTAUTH           DS   F      Whether TLS client
*                                        authentication is required
MQCD_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_CERTLABL                DS   F      Certificate label
MQCD_SPLPROTECTION           DS   F      AMS Security policy protection
MQCD_LENGTH                  EQU  *-MQCD
                             ORG  MQCD
MQCD_AREA                    DS   CL(MQCD_LENGTH)
```

Vizuální základní deklarace

Tato deklarace je deklarací jazyka Visual Basic struktury MQCD.

V jazyku Visual Basic lze strukturu MQCD použít se strukturou MQCNO ve volání 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  'TLS CipherSpec'
  SSLPeerNamePtr         As MQPTR      'Address of TLS peer name'
  SSLPeerNameLength      As Long       'Length of TLS peer name'
  SSLClientAuth          As Long       'Whether TLS client'
                                       'authentication is required'
  KeepAliveInterval      As Long       'Keepalive interval'
  LocalAddress           As String*48  'Local communications address'
  BatchHeartbeat         As Long       'Batch heartbeat interval'
  HdrCompList(0 to 1)    As Long2      'Header data compression list'
  MsgCompList(0 To 15)   As Long16     'Message data compression list'
  CLWLChannelRank        As Long       'Channel Rank'
  CLWLChannelPriority    As Long       'Channel priority'
  CLWLChannelWeight      As Long       'Channel Weight'
  ChannelMonitoring      As Long       'Channel Monitoring control'
  ChannelStatistics      As Long       'Channel Statistics'
End Type
```

Změna polí MQCD v uživatelské proceduře kanálu

Uživatelská procedura kanálu může změnit pole v MQCD. Tyto změny se však obvykle neprovádějí, s výjimkou uvedených okolností.

Pokud program uživatelské procedury kanálu změní pole v datové struktuře MQCD, proces kanálu IBM MQ obvykle novou hodnotu ignoruje. Nová hodnota však zůstává v MQCD a je předána všem zbývajícím uživatelským procedurám v řetězci uživatelských procedur a všem konverzím sdílejících instanci kanálu.

Je-li vlastnost SharingConversations ve struktuře MQCXP nastavena na hodnotu FALSE, lze provádět změny určitých polí v závislosti na typu uživatelského programu, typu kanálu a kódu příčiny ukončení. V následující tabulce jsou uvedena pole, která lze změnit a ovlivnit chování kanálu a za jakých okolností. Pokud uživatelský program změní jedno z těchto polí za jakýchkoli jiných okolností nebo jakékoli pole, které není uvedeno v seznamu, proces kanálu novou hodnotu ignoruje. Nová hodnota zůstává v MQCD a je předána všem zbývajícím uživatelským procedurám v řetězci uživatelských procedur a všem konverzací sdílejících instanci kanálu.

Libovolný typ uživatelského programu při volání pro inicializaci (MQXR_INIT) může změnit pole ChannelName libovolného typu kanálu, pokud je parametr MQCXP SharingConverstions nastaven na hodnotu FALSE. Pole MCAUserIdentifier může změnit pouze uživatelská procedura zabezpečení bez ohledu na hodnotu MQCXP SharingConverstions.

Tabulka 823. Pole, která lze změnit a ovlivnit chování kanálu

Pole	**Kód příčiny ukončení**	**Typ ukončení**	**Typ kanálu**
ChannelName	MQXR_INIT	Vše	Vše
TransportType	MQXR_INIT	Vše	Vše
XmitQName	MQXR_INIT	Vše	SDR, RCVR
ModeName	MQXR_INIT	Vše	Vše
TpName	MQXR_INIT	Vše	Vše
BatchSize	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
DiscInterval	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR

Tabulka 823. Pole, která lze změnit a ovlivnit chování kanálu (pokračování)

Pole	Kód příčiny ukončení	Typ ukončení	Typ kanálu
Počet ShortRetry	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Interval ShortRetry	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Počet LongRetry	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
Interval LongRetry	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
SeqNumberZalamovat	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
MaxMsgLength	MQXR_INIT	Vše	Vše
PutAuthority	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
DataConversion	MQXR_INIT	Vše	Vše
MCAUserIdentifier	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	Zabezpečení	RCVR, RQSTR, SVRCONN, CLUSRCVR
ConnectionName	MQXR_INIT	Vše	SDR, SVR, RQSTR, CLNTCONN, CLUSSDR, CLUSRCVR
MsgRetryUserData	MQXR_INIT	Vše	RCVR, RQSTR, CLUSRCVR
Počet MsgRetry	MQXR_INIT	Vše	RCVR, RQSTR, CLUSRCVR

Tabulka 823. Pole, která lze změnit a ovlivnit chování kanálu (pokračování)

Pole	Kód příčiny ukončení	Typ ukončení	Typ kanálu
Interval MsgRetry	MQXR_INIT	Vše	RCVR, RQSTR, CLUSRCVR
HeartbeatInterval	MQXR_INIT	Vše	Vše
BatchInterval	MQXR_INIT	Vše	SDR, SVR, CLUSSDR, CLUSRCVR
NonPersistentMsgSpeed	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
MCASecurityId	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	Zabezpečení	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSSDR, CLUSRCVR
SSLCipherSpec	MQXR_INIT	Vše	Vše
SSLPeerNamePtr	MQXR_INIT	Vše	Vše
Délka parametru SSLPeerName	MQXR_INIT	Vše	Vše
SSLClientAuth	MQXR_INIT	Vše	SVR, RCVR, RQSTR, SVRCONN, CLUSRCVR
KeepAliveInterval	MQXR_INIT	Vše	Vše
LocalAddress	MQXR_INIT	Vše	SDR, SVR, RQSTR, CLNTCONN, CLUSSDR, CLUSRCVR
BatchHeartbeat	MQXR_INIT	Vše	SDR, SVR, CLUSSDR, CLUSRCVR
Seznam HdrComp	MQXR_INIT	Vše	Vše
Seznam MsgComp	MQXR_INIT	Vše	Vše
ChannelMonitoring	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSSDR, CLUSRCVR

Tabulka 823. Pole, která lze změnit a ovlivnit chování kanálu (pokračování)			
Pole	**Kód příčiny ukončení**	**Typ ukončení**	**Typ kanálu**
ChannelStatistics	MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
SharingConversations	MQXR_INIT	Vše	SVRCONN, CLNTCONN
PropertyControl	MQXR_INIT	Vše	SDR, SVR, CLUSSDR, CLUSRCVR

MQCXP-Parametr uživatelské procedury kanálu

Struktura MQCXP je předána každému typu uživatelské procedury volané agentem MCA (Message Channel Agent), kanálem připojení klienta nebo kanálem připojení serveru.

Viz MQ_CHANNEL_EXIT.

Pole popsaná jako "vstup do uživatelské procedury" v popisech, které následují, jsou kanálem ignorovány, když uživatelská procedura vrací řízení do kanálu. Vstupní pole, která uživatelská procedura změní v bloku parametrů uživatelské procedury kanálu, nebudou zachována pro další vyvolání. Změny provedené ve vstupních/výstupních polích (například v poli *ExitUserArea*) jsou zachovány pouze pro vyvolání této instance uživatelské procedury. Tyto změny nelze použít k předávání dat mezi různými uživatelskými procedurami definovanými na stejném kanálu nebo mezi stejnými uživatelskými procedurami definovanými na různých kanálech.

Související odkazy

“Pole” na stránce 1512
Toto téma obsahuje seznam všech polí ve struktuře MQCXP a popisuje jednotlivá pole.

“C prohlášení” na stránce 1524
Tato deklarace je deklarací jazyka C pro strukturu MQCXP.

“Deklarace jazyka COBOL” na stránce 1524
Tato deklarace je deklarací jazyka COBOL pro strukturu MQCXP.

“Deklarace RPG (ILE)” na stránce 1525
Tato deklarace je deklarací RPG pro strukturu MQCXP.

“System/390 deklarace modulu asembleru” na stránce 1526
Tato deklarace je deklarací modulu sestavení System/390 pro strukturu MQCXP.

Pole

Toto téma obsahuje seznam všech polí ve struktuře MQCXP a popisuje jednotlivá pole.

StrucId (MQCHAR4)
Toto pole uvádí identifikátor struktury.

Hodnota musí být:

MQCXP_STRUC_ID
Identifikátor pro strukturu parametrů uživatelské procedury kanálu.

Pro programovací jazyk C je definována také konstanta MQCXP_STRUC_ID_ARRAY; tato konstanta má stejnou hodnotu jako MQCXP_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro ukončení.

Verze (MQLONG)
Toto pole určuje číslo verze struktury.

Hodnota závisí na prostředí:

MQCXP_VERSION_1
Struktura parametrů uživatelské procedury kanálu Version-1 .

MQCXP_VERSION_2
Struktura parametrů uživatelské procedury kanálu Version-2 .

MQCXP_VERSION_3
Struktura parametrů uživatelské procedury kanálu Version-3 .

Linux AIX Pole má tuto hodnotu v systémech AIX and Linux , které nejsou uvedeny jinde.

MQCXP_VERSION_4
Struktura parametrů uživatelské procedury kanálu Version-4 .

MQCXP_VERSION_5
Struktura parametrů uživatelské procedury kanálu Version-5 .

MQCXP_VERSION_6
Struktura parametrů uživatelské procedury kanálu Version-6 .

MQCXP_VERSION_8
Struktura parametrů uživatelské procedury kanálu Version-8 .

z/OS Pole má tuto hodnotu v souboru z/OS.

MQCXP_VERSION_9
Struktura parametrů uživatelské procedury kanálu Version-9 .

Pole má tuto hodnotu v následujících prostředích:

- AIX AIX
- IBM i IBM i
- Linux Linux
- Windows Windows
- z/OS z/OS

Pole, která existují pouze v novějších verzích struktury, jsou jako taková identifikována v popisech polí. Následující konstanta určuje číslo verze aktuální verze:

MQCXP_CURRENT_VERSION
Aktuální verze struktury parametrů uživatelské procedury kanálu.

Hodnota závisí na prostředí.

Poznámka: Při zavedení nové verze struktury MQCXP se nezmění rozvržení existující části. Uživatelská procedura proto musí zkontrolovat, zda je číslo verze stejné nebo větší než nejnižší verze, která obsahuje pole, která uživatelská procedura potřebuje použít.

Toto je vstupní pole pro ukončení.

ExitId (MQLONG)
Toto pole určuje typ volané uživatelské procedury, která je nastavena při vstupu do uživatelské procedury.

Možné jsou následující hodnoty:

MQXT_CHANNEL_SEC_EXIT
Uživatelská procedura zabezpečení kanálu.

MQXT_CHANNEL_MSG_EXIT
Uživatelská procedura pro zprávy kanálu.

MQXT_CHANNEL_SEND_EXIT
Uživatelská procedura odeslání kanálu.

MQXT_CHANNEL_RCV_EXIT
Uživatelská procedura pro příjem kanálu.

MQXT_CHANNEL_MSG_RETRY_EXIT
Zpráva kanálu-uživatelská procedura opakování.

MQXT_CHANNEL_AUTO_DEF_EXIT
Uživatelská procedura automatické definice kanálu.

V systému z/OSje tento typ uživatelské procedury podporován pouze pro kanály typu MQCHT_CLUSSDR a MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení.

ExitReason (MQLONG)
Toto pole určuje příčinu, proč je uživatelská procedura volána a nastavena při vstupu do uživatelské procedury.

Není používán uživatelskou procedurou automatické definice. Možné jsou následující hodnoty:

MQXR_INIT
Inicializace ukončení.

Tato hodnota označuje, že uživatelská procedura je vyvolávána poprvé. Umožňuje uživatelské proceduře získat a inicializovat všechny prostředky, které potřebuje (například paměť).

MQXR_TERM
Ukončení ukončení.

Tato hodnota označuje, že uživatelská procedura bude ukončena. Uživatelská procedura by měla uvolnit všechny prostředky, které získala od své inicializace (například: paměť).

MQXR_MSG
Zpracujte zprávu.

Tato hodnota označuje, že uživatelská procedura je vyvolávána pro zpracování zprávy. Tato hodnota se vyskytuje pouze pro uživatelské procedury zpráv kanálu.

MQXR_XMIT
Zpracování přenosu.

Tato hodnota se vyskytuje pouze pro uživatelské procedury odeslání a přijetí kanálu.

MQXR_SEC_MSG
Byla přijata zpráva zabezpečení.

Tato hodnota se vyskytuje pouze pro uživatelské procedury zabezpečení kanálu.

MQXR_INIT_SEC
Zahajte bezpečnostní výměnu.

Tato hodnota se vyskytuje pouze pro uživatelské procedury zabezpečení kanálu.

Uživatelská procedura zabezpečení příjemce je vždy vyvolána s tímto důvodem ihned po vyvolání s MQXR_INIT, aby měla možnost zahájit výměnu zabezpečení. Pokud odmítne příležitost (vrácením MQXCC_OK namísto MQXCC_SEND_SEC_MSG nebo MQXCC_SEND_AND_REQUEST_SEC_MSG), uživatelská procedura zabezpečení odesilatele se vyvolá s MQXR_INIT_SEC.

Pokud uživatelská procedura zabezpečení příjemce zahájí výměnu zabezpečení (vrácením MQXCC_SEND_SEC_MSG nebo MQXCC_SEND_AND_REQUEST_SEC_MSG), uživatelská procedura zabezpečení odesilatele není nikdy vyvolána pomocí MQXR_INIT_SEC; místo toho je vyvolána pomocí MQXR_SEC_MSG ke zpracování zprávy příjemce. (V obou případech je nejprve vyvolán pomocí MQXR_INIT.)

Pokud některá z procedur zabezpečení nepožaduje ukončení kanálu (nastavením parametru *ExitResponse* na hodnotu MQXCC_SUPPRESS_FUNCTION nebo MQXCC_CLOSE_CHANNEL), musí být výměna zabezpečení dokončena na straně, která zahájila výměnu. Proto, pokud je uživatelská procedura zabezpečení vyvolána s MQXR_INIT_SEC a iniciuje výměnu, bude při příštím vyvolání uživatelské procedury s MQXR_SEC_MSG. K tomu dochází bez ohledu na to, zda existuje zpráva zabezpečení, kterou má uživatelská procedura zpracovat či nikoli. Pokud partner vrátí MQXCC_SEND_SEC_MSG nebo MQXCC_SEND_AND_REQUEST_SEC_MSG, objeví se zpráva zabezpečení, ale ne, pokud partner vrátí MQXCC_OK, nebo u partnera neexistuje žádná uživatelská procedura zabezpečení. Pokud není k dispozici žádná zpráva zabezpečení ke zpracování, uživatelská procedura zabezpečení na zahajovacím konci je znovu vyvolána s hodnotou *DataLength* nula.

MQXR_RETRY
Zopakujte zprávu.

Tato hodnota se vyskytuje pouze pro uživatelské procedury opakování zprávy.

MQXR_AUTO_CLUSSDR
Automatická definice odesílacího kanálu klastru.

Tato hodnota se vyskytuje pouze pro uživatelské procedury automatické definice kanálu.

MQXR_AUTO_RECEIVER
Automatická definice přijímacího kanálu.

Tato hodnota se vyskytuje pouze pro uživatelské procedury automatické definice kanálu.

MQXR_AUTO_SVRCONN
Automatická definice kanálu připojení serveru.

Tato hodnota se vyskytuje pouze pro uživatelské procedury automatické definice kanálu.

MQXR_AUTO_CLUSRCVR
Automatická definice přijímacího kanálu klastru.

Tato hodnota se vyskytuje pouze pro uživatelské procedury automatické definice kanálu.

MQXR_SEC_PARMS
Parametry zabezpečení

Tato hodnota se vztahuje pouze na uživatelské procedury zabezpečení a označuje, že uživatelské proceduře je předána struktura MQCSP. Další informace viz "MQCSP-parametry zabezpečení" na stránce 337

Poznámka:

1. Máte-li pro kanál nadefinovanou více než jednu uživatelskou proceduru, budou při inicializaci agenta MCA vyvolány pomocí MQXR_INIT. Kromě toho jsou všechny vyvolány pomocí MQXR_TERM při ukončení agenta MCA.
2. Pro uživatelskou proceduru automatické definice kanálu není hodnota *ExitReason* nastavena, pokud je hodnota *Version* menší než hodnota MQCXP_VERSION_4. Hodnota MQXR_AUTO_SVRCONN je v tomto případě odvozena.

Toto je vstupní pole pro ukončení.

ExitResponse (MQLONG)
Toto pole určuje odezvu z ukončení.

Toto pole je nastaveno uživatelskou procedurou pro komunikaci s agentem MCA. Musí to být jedna z následujících hodnot:

MQXCC_OK
Ukončení bylo úspěšně dokončeno.

- Pro uživatelskou proceduru zabezpečení kanálu tato hodnota označuje, že přenos zpráv může nyní pokračovat normálně.

- Pro uživatelskou proceduru opakování zpráv kanálu tato hodnota označuje, že agent MCA musí čekat na časový interval vrácený uživatelskou procedurou v poli *MsgRetryInterval* v prostředí MQCXP, a pak zprávu zopakujte.

Pole *ExitResponse2* může obsahovat další informace.

MQXCC_SUPPRESS_FUNCTION
Potlačit funkci.

- Pro uživatelskou proceduru pro zabezpečení zprávy kanálu tato hodnota označuje, že kanál musí být ukončen.
- V případě uživatelské procedury pro zprávy kanálu tato hodnota označuje, že zpráva nemá pokračovat dále směrem k cíli. Místo toho agent MCA vygeneruje zprávu sestavy výjimek (pokud byla vyžádána odesilatelem původní zprávy) a umístí zprávu obsaženou v původní vyrovnávací paměti do fronty nedoručených zpráv (pokud odesilatel zadal parametr MQRO_DEAD_LETTER_Q) nebo ji zruší (pokud odesilatel zadal parametr MQRO_DISCARD_MSG).

  V případě trvalých zpráv platí, že pokud odesílatel zadal hodnotu MQRO_DEAD_LETTER_Q, ale vložení do fronty nedoručených zpráv se nezdaří nebo fronta nedoručených zpráv neexistuje, původní zpráva zůstane v přenosové frontě a zpráva sestavy nebude generována. Původní zpráva je také ponechána v přenosové frontě, pokud zprávu sestavy nelze úspěšně vygenerovat.

  Pole *Feedback* ve struktuře MQDLH na začátku zprávy ve frontě nedoručených zpráv označuje, proč byla zpráva vložena do fronty nedoručených zpráv. Tento kód zpětné vazby je také použit v deskriptoru zprávy zprávy o výjimce (pokud ji požadoval odesilatel).
- Pro uživatelskou proceduru opakování zpráv kanálu tato hodnota označuje, že agent MCA nečeká a zprávu zopakuje; místo toho agent MCA pokračuje okamžitě se svým běžným zpracováním selhání (zpráva je umístěna do fronty nedoručených zpráv nebo vyřazena, jak je uvedeno odesilatelem zprávy).
- Pro uživatelskou proceduru automatické definice kanálu musí být zadána hodnota MQXCC_OK nebo MQXCC_SUPPRESS_FUNCTION. Není-li zadána žádná z těchto hodnot, předpokládá se standardně hodnota MQXCC_SUPPRESS_FUNCTION a automatická definice je opuštěna.

Tato odezva není podporována pro uživatelské procedury odeslání a přijetí kanálu.

MQXCC_SEND_SEC_MSG
Odeslat zprávu zabezpečení.

Tuto hodnotu může nastavit pouze uživatelská procedura zabezpečení kanálu. Označuje, že uživatelská procedura poskytla zprávu zabezpečení, která musí být předána partnerovi.

MQXCC_SEND_AND_REQUEST_SEC_MSG
Odeslat zprávu zabezpečení, která vyžaduje odpověď.

Tuto hodnotu může nastavit pouze uživatelská procedura zabezpečení kanálu. Označuje, že

- že výstup poskytl bezpečnostní zprávu, kterou lze předat partnerovi, a
- že ukončení vyžaduje odpověď od partnera. Není-li přijata žádná odpověď, musí být kanál ukončen, protože uživatelská procedura ještě nerozhodla, zda může komunikace pokračovat.

MQXCC_SUPPRESS_EXIT
Potlačit ukončení.

- Tuto hodnotu mohou nastavit všechny typy uživatelské procedury kanálu kromě uživatelské procedury pro zabezpečení nebo uživatelské procedury pro automatickou definici. Potlačuje jakékoli další vyvolání této uživatelské procedury (jako by její název byl v definici kanálu prázdný) až do ukončení kanálu, kdy je uživatelská procedura znovu vyvolána s hodnotou *ExitReason* MQXR_TERM.
- Pokud uživatelská procedura pro opakování zpráv vrátí tuto hodnotu, opakované pokusy pro následné zprávy jsou řízeny atributy kanálu *MsgRetryCount* a *MsgRetryInterval* jako obvykle. Pro aktuální zprávu provede agent MCA počet neprovedených opakování v intervalech určených atributem kanálu *MsgRetryInterval* , ale pouze v případě, že kód příčiny je takový, který by agent

MCA normálně zopakoval (viz pole *MsgRetryCount* popsané v části “MQCD-Definice kanálu” na stránce 1471). Počet neprovedených opakovaných pokusů je hodnota atributu **MsgRetryCount** minus počet, kolikrát uživatelská procedura vrátila hodnotu MQXCC_OK pro aktuální zprávu; je-li tento počet záporný, MCA neprovede pro aktuální zprávu žádné další opakování.

MQXCC_UZAVŘENÝ_KANÁL
Zavřít kanál.

Tuto hodnotu lze nastavit libovolným typem uživatelské procedury kanálu kromě uživatelské procedury s automatickou definicí.

Není-li sdílení konverzací povoleno, tato hodnota kanál zavře.

Je-li povoleno sdílení konverzací, tato hodnota ukončí konverzaci. Pokud je tato konverzace jedinou konverzací na kanálu, kanál se také zavře.

Toto pole je vstupní/výstupní pole z ukončení.

ExitResponse2 (MQLONG)
Toto pole určuje sekundární odpověď z uživatelské procedury.

Toto pole je při vstupu do uživatelské procedury nastaveno na nulu. Uživatelská procedura ji může nastavit tak, aby poskytovala další informace o funkcích kanálu IBM MQ . Není používán uživatelskou procedurou automatické definice.

Uživatelská procedura může nastavit jednu nebo více následujících hodnot. Pokud se požaduje více než jedna, hodnoty se přidají. Kombinace, které nejsou platné, jsou uvedeny; jiné kombinace jsou povoleny.

MQXR2_PUT_WITH_DEF_ACTION
Vložit s výchozí akcí.

Tato hodnota je nastavena uživatelskou procedurou kanálu pro zprávy příjemce. Označuje, že zpráva má být vložena s výchozí akcí agenta MCA, což je buď výchozí ID uživatele agenta MCA, nebo kontext *UserIdentifier* v deskriptoru zprávy MQMD (deskriptor zprávy).

Hodnota je nula, což odpovídá počáteční hodnotě nastavené při vyvolání ukončení. Konstanta je poskytována pro dokumentační účely.

MQXR2_PUT_WITH_DEF_USERID
Vložit s výchozím identifikátorem uživatele.

Tuto hodnotu může nastavit pouze uživatelská procedura pro zprávy kanálu příjemce. Označuje, že zpráva má být vložena s výchozím identifikátorem uživatele MCA.

MQXR2_PUT_WITH_MSG_USERID
Vložit s identifikátorem uživatele zprávy.

Tuto hodnotu může nastavit pouze uživatelská procedura pro zprávy kanálu příjemce. Označuje, že zpráva má být vložena s kontextem *UserIdentifier* do deskriptoru zprávy MQMD (deskriptor zprávy) zprávy (ta mohla být upravena uživatelskou procedurou).

Měl by být nastaven pouze jeden z parametrů MQXR2_PUT_WITH_DEF_ACTION, MQXR2_PUT_WITH_DEF_USERIDa MQXR2_PUT_WITH_MSG_USERID .

MQXR2_USE_AGENT_BUFFER
Použít vyrovnávací paměť agenta.

Tato hodnota označuje, že data, která mají být předána, jsou v *AgentBuffer*, ne *ExitBufferAddr*.

Hodnota je nula, což odpovídá počáteční hodnotě nastavené při vyvolání ukončení. Konstanta je poskytována pro dokumentační účely.

MQXR2_USE_EXIT_BUFFER
Použít výstupní vyrovnávací paměť.

Tato hodnota označuje, že data, která mají být předána, jsou v *ExitBufferAddr*, ne *AgentBuffer*.

Měl by být nastaven pouze jeden z parametrů MQXR2_USE_AGENT_BUFFER a MQXR2_USE_EXIT_BUFFER .

MQXR2_DEFAULT_CONTINUATION
Výchozí pokračování.

Pokračování s další uživatelskou procedurou v řetězci závisí na odezvě z poslední vyvolané uživatelské procedury:

- Jsou-li vráceny hodnoty MQXCC_SUPPRESS_FUNCTION nebo MQXCC_CLOSE_CHANNEL, nejsou volány žádné další uživatelské procedury v řetězu.
- Jinak je vyvolána další uživatelská procedura v řetězci.

MQXR2_CONTINUE_CHAIN
Pokračujte dalším ukončením.

MQXR2_SUPPRESS_CHAIN
Přeskočit zbývající východy v řetězci.

Toto je vstupní/výstupní pole pro ukončení.

Zpětná vazba (MQLONG)
Toto pole určuje kód zpětné vazby.

Toto pole je nastaveno na hodnotu MQFB_NONE při vstupu do uživatelské procedury.

Pokud uživatelská procedura pro zprávy kanálu nastaví pole *ExitResponse* na hodnotu MQXCC_SUPPRESS_FUNCTION, pole *Feedback* určuje kód zpětné vazby, který identifikuje, proč byla zpráva vložena do fronty nedoručených zpráv (nedoručených zpráv), a také se používá k odeslání zprávy o výjimce, pokud byla požadována. V tomto případě, pokud je pole *Feedback* MQFB_NONE, použije se následující kód zpětné vazby:

MQFB_STOPPED_BY_MSG_EXIT
Zpráva byla zastavena uživatelskou procedurou pro zprávy kanálu.

Hodnota vrácená v tomto poli pomocí uživatelských procedur zabezpečení kanálu, odeslání, přijetí a opakování zpráv není použita agentem MCA.

Hodnota vrácená v tomto poli pomocí uživatelských procedur automatické definice se nepoužije, pokud je *ExitResponse* MQXCC_OK, ale jinak se použije pro parametr *AuxErrorDataInt1* ve zprávě události.

Jedná se o vstupní/výstupní pole z ukončení.

MaxSegmentDélka (MQLONG)
Toto pole uvádí maximální délku v bajtech, kterou lze odeslat v jednom přenosu.

Není používán uživatelskou procedurou automatické definice. Je zajímavé pro uživatelskou proceduru odeslání kanálu, protože tato uživatelská procedura musí zajistit, aby nezvýšila velikost přenosového segmentu na hodnotu větší než *MaxSegmentLength*. Délka zahrnuje počáteční 8 bajtů, které nesmí uživatelská procedura změnit. Hodnota je vyjednána mezi funkcemi kanálu IBM MQ při inicializaci kanálu. Další informace o délkách segmentů naleznete v tématu Psaní programů channel-exit .

Hodnota v tomto poli není smysluplná, pokud je *ExitReason* MQXR_INIT.

Toto je vstupní pole pro ukončení.

Oblast ExitUser(MQBYTE16)
Toto pole určuje uživatelskou oblast ukončení-pole, které je k dispozici pro uživatelskou proceduru, která má být použita.

Je inicializován na binární nulu před prvním vyvoláním uživatelské procedury (která má hodnotu *ExitReason* nastavenou na MQXR_INIT) a poté jsou všechny změny provedené touto uživatelskou procedurou v tomto poli zachovány při všech voláních uživatelské procedury.

Je definována následující hodnota:

MQXUA_NONE
Žádné informace o uživateli.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je definována také konstanta MQXUA_NONE_ARRAY; tato konstanta má stejnou hodnotu jako MQXUA_NONE, ale je polem znaků místo řetězce.

Délka tohoto pole je dána hodnotou MQ_EXIT_USER_AREA_LENGTH. Toto je vstupní/výstupní pole pro ukončení.

ExitData (MQCHAR32)
Toto pole uvádí data ukončení.

Toto pole je při vstupu do uživatelské procedury nastaveno na informace, které funkce kanálu IBM MQ převzaly z definice kanálu. Pokud takové informace nejsou k dispozici, toto pole je prázdné.

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto je vstupní pole pro ukončení.

Následující pole v této struktuře nejsou přítomna, pokud je hodnota *Version* menší než MQCXP_VERSION_2.

Počet MsgRetry(MQLONG)
Toto pole určuje, kolikrát byla zpráva zopakována.

Při prvním vyvolání uživatelské procedury pro konkrétní zprávu má toto pole hodnotu nula (dosud se nezkoušely žádné opakované pokusy). Při každém následném vyvolání uživatelské procedury pro tuto zprávu se hodnota zvýší o jedničku o hodnotu MCA.

Toto je vstupní pole pro ukončení. Hodnota v tomto poli není smysluplná, pokud je *ExitReason* MQXR_INIT. Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_2.

MsgRetryInterval (MQLONG)
Toto pole určuje minimální interval v milisekundách, po jehož uplynutí se zopakuje operace vložení.

Při prvním vyvolání uživatelské procedury pro konkrétní zprávu toto pole obsahuje hodnotu atributu kanálu *MsgRetryInterval* . Uživatelská procedura může ponechat hodnotu beze změny nebo ji upravit tak, aby určovala jiný časový interval v milisekundách. Pokud uživatelská procedura vrátí MQXCC_OK v souboru *ExitResponse*, MCA před zopakováním operace MQOPEN nebo MQPUT počká alespoň tento časový interval. Uvedený časový interval musí být nula nebo větší.

Druhý a následující okamžik, kdy je pro tuto zprávu vyvolána uživatelská procedura, obsahuje toto pole hodnotu vrácenou předchozím vyvoláním uživatelské procedury.

Je-li hodnota vrácená v poli *MsgRetryInterval* menší než nula nebo větší než 999 999 999 a *ExitResponse* je MQXCC_OK, agent MCA ignoruje pole *MsgRetryInterval* v MQCXP a místo toho čeká na interval určený atributem kanálu *MsgRetryInterval* .

Toto je vstupní/výstupní pole pro ukončení. Hodnota v tomto poli není smysluplná, pokud je *ExitReason* MQXR_INIT. Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_2.

MsgRetryPříčina (MQLONG)
Toto pole určuje kód příčiny z předchozího pokusu o vložení zprávy.

Toto pole je kód příčiny z předchozího pokusu o vložení zprávy; jedná se o jednu z hodnot MQRC_ *.

Toto je vstupní pole pro ukončení. Hodnota v tomto poli není smysluplná, pokud je *ExitReason* MQXR_INIT. Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_2.

Následující pole v této struktuře nejsou přítomna, pokud je hodnota *Version* menší než MQCXP_VERSION_3.

HeaderLength (MQLONG)
Toto pole určuje délku informací záhlaví.

Toto pole je relevantní pouze pro uživatelskou proceduru zprávy a uživatelskou proceduru opakování zprávy. Hodnota je délka struktur záhlaví směrování na začátku dat zprávy; jedná se o strukturu MQXQH, záhlaví MQMDE (záhlaví rozšíření popisu zprávy) a (pro zprávu distribučního seznamu) strukturu MQDH a pole záznamů MQOR a MQPMR, které následují za strukturou MQXQH.

Uživatelská procedura zprávy může tyto informace záhlaví zkontrolovat a v případě potřeby je upravit, ale data, která uživatelská procedura vrátí, musí být stále ve správném formátu. Uživatelská procedura nesmí například šifrovat nebo komprimovat data záhlaví na odesílajícím konci, a to ani v případě, že uživatelská procedura zprávy na přijímajícím konci provede kompenzaci změn.

Pokud uživatelská procedura zprávy upraví informace záhlaví tak, aby se změnila jejich délka (například přidáním dalšího místa určení do zprávy rozdělovníku), musí před návratem odpovídajícím způsobem změnit hodnotu *HeaderLength* .

Toto je vstupní/výstupní pole pro ukončení. Hodnota v tomto poli není smysluplná, pokud je *ExitReason* MQXR_INIT. Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_3.

PartnerName (MQCHAR48)
Toto pole určuje název partnera.

Název partnera, jak je uvedeno:

- Pro kanály SVRCONN se jedná o ID přihlášeného uživatele v klientovi.
- Pro všechny ostatní typy kanálů se jedná o název správce front partnera.

Při inicializaci uživatelské procedury je toto pole prázdné, protože správce front nezná název partnera, dokud nedojde k počátečnímu vyjednávání.

Toto je vstupní pole pro ukončení. Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_3.

FAPLevel (MQLONG)
Úroveň vyjednaných formátů a protokolů.

Toto je vstupní pole pro ukončení. Změny tohoto pole by měly být provedeny pouze pod vedením služby IBM . Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_3.

CapabilityFlags (MQLONG)
Příznak schopnosti můžete nastavit na hodnotu MQCF_NONE nebo MQCF_DIST_LISTS.

Můžete nastavit jeden z následujících příznaků schopnosti:

MQCF_NONE
Žádné příznaky.

MQCF_DIST_LISTS
Distribuční seznamy jsou podporovány.

Toto je vstupní pole pro ukončení. Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_3.

ExitNumber (MQLONG)
Toto pole určuje pořadové číslo ukončení.

Pořadové číslo ukončení v rámci typu definovaného v souboru *ExitId*. Pokud je například vyvolávaná uživatelská procedura třetí definovanou uživatelskou procedurou pro zprávy, toto pole obsahuje hodnotu 3. Je-li typ uživatelské procedury takový, pro který nelze definovat seznam uživatelských procedur (například uživatelská procedura pro zabezpečení zprávy), má toto pole hodnotu 1.

Toto je vstupní pole pro ukončení. Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_3.

Následující pole v této struktuře nejsou přítomna, pokud je hodnota *Version* menší než MQCXP_VERSION_5.

ExitSpace (MQLONG)
Toto pole určuje počet bajtů v přenosové vyrovnávací paměti vyhrazené pro uživatelskou proceduru, která má být použita.

Toto pole je relevantní pouze pro uživatelskou proceduru odeslání. Určuje velikost prostoru v bajtech, který funkce kanálu IBM MQ rezervují ve vyrovnávací paměti pro přenos, kterou má uživatelská procedura použít. Toto pole umožňuje uživatelské proceduře přidat do přenosové vyrovnávací paměti malé množství dat (obvykle nepřesahující několik set bajtů) pro použití doplňkovou uživatelskou procedurou pro příjem na druhém konci. Data přidaná uživatelskou procedurou pro odesílání musí být odebrána uživatelskou procedurou pro příjem.

Hodnota je v systému z/OSvždy nula.

Poznámka: Tento prostředek nesmí být používán k odesílání velkého množství dat, protože by mohl snížit výkon nebo dokonce blokovat činnost kanálu.

Nastavením parametru *ExitSpace* je zaručeno, že v přenosové vyrovnávací paměti bude vždy k dispozici alespoň tento počet bajtů, které má uživatelská procedura používat. Uživatelská procedura však může použít méně než vyhrazené množství, nebo více než vyhrazené množství, pokud je v přenosové vyrovnávací paměti k dispozici místo. Výstupní prostor ve vyrovnávací paměti je poskytován po existujících datech.

Příkaz *ExitSpace* může být nastaven uživatelskou procedurou pouze v případě, že má parametr *ExitReason* hodnotu MQXR_INIT; ve všech ostatních případech je hodnota vrácená uživatelskou procedurou ignorována. Na vstupu do uživatelské procedury je hodnota *ExitSpace* nula pro volání MQXR_INIT a hodnota vrácená voláním MQXR_INIT v jiných případech.

Je-li hodnota vrácená voláním MQXR_INIT záporná nebo je-li ve vyrovnávací paměti přenosu pro data zpráv k dispozici méně než 1024 bajtů po vyhrazení požadovaného prostoru uživatelské procedury pro všechny uživatelské procedury odeslání v řetězci, vypíše agent MCA chybovou zprávu a zavře kanál. Podobně, pokud během přenosu dat uživatelské procedury v řetězci uživatelských procedur pro odesílání alokují více uživatelského prostoru, než kolik jich vyhrazilo, takže ve vyrovnávací paměti pro přenos dat zpráv zůstane méně než 1024 bajtů, vypíše agent MCA chybovou zprávu a zavře kanál. Limit 1024 umožňuje, aby řídicí a administrativní toky kanálu byly zpracovány řetězcem uživatelských procedur pro odesílání bez nutnosti segmentovat toky.

Jedná se o vstupní/výstupní pole pro ukončení, pokud je *ExitReason* MQXR_INIT, a vstupní pole ve všech ostatních případech. Pole není k dispozici, pokud je hodnota *Version* menší než MQCXP_VERSION_5.

SSLCertUserID (MQCHAR12)
Toto pole uvádí UserId přidružené ke vzdálenému certifikátu.

Na všech platformách kromě platformy z/OS je prázdná.

Toto je vstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCXP_VERSION_6.

SSLRemCertIssName(MQLONG)
Toto pole určuje délku (v bajtech) úplného rozlišujícího názvu vydavatele vzdáleného certifikátu, na který odkazuje příkaz Ptr SSLCertRemoteIssuerName.

Toto je vstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCXP_VERSION_6. Hodnota je nula, pokud se nejedná o kanál TLS.

SSLRemCertIssNamePtr (PMQVOID)
Toto pole určuje adresu úplného rozlišujícího názvu vydavatele vzdáleného certifikátu.

Jeho hodnota je ukazatel Null, pokud se nejedná o kanál TLS.

Toto je vstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCXP_VERSION_6.

Poznámka: Chování uživatelských procedur zabezpečení kanálu při určování rozlišujícího názvu subjektu a rozlišujícího názvu vydavatele se změnilo z IBM WebSphere MQ 7.1. Další informace naleznete v tématu Programy uživatelských procedur pro zabezpečení kanálu.

SecurityParms (PMQCSP)
Toto pole určuje adresu struktury MQCSP, která se používá k určení ověřovacích pověření.

Počáteční hodnota tohoto pole je ukazatel Null.

Toto je vstupní/výstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCXP_VERSION_6.

Hodnota, kterou uživatelská procedura vrátí v tomto poli, musí být použitelná pro IBM MQ , dokud nebude MQXR_TERM.

CurHdrKomprese (MQLONG)
Toto pole uvádí, která technika se momentálně používá ke kompresi dat záhlaví.

Je nastaven na jednu z následujících hodnot:

MQCOMPRESS_NONE
Neprovádí se žádná komprese dat hlavičky.

MQCOMPRESS_SYSTEM
Provádí se komprese dat hlavičky.

Hodnotu lze změnit uživatelskou procedurou pro zprávy odesílajícího kanálu na jednu z vyjednaných podporovaných hodnot přístupných z pole HdrCompv seznamu MQCD. To umožňuje techniku použitou ke kompresi dat záhlaví, která mají být zvolena pro každou zprávu na základě obsahu zprávy. Změněná hodnota se použije pouze pro aktuální zprávu. Kanál se ukončí, pokud je atribut změněn na nepodporovanou hodnotu. Hodnota se ignoruje, pokud se změní mimo uživatelskou proceduru pro zprávu odesílajícího kanálu.

Toto je vstupní/výstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCXP_VERSION_6.

CurMsgKomprese (MQLONG)
Toto pole uvádí, která technika se momentálně používá ke kompresi dat zprávy.

Je nastaven na jednu z následujících hodnot:

MQCOMPRESS_NONE
Neprovádí se žádná komprese dat hlavičky.

MQCOMPRESS_RLE
Komprese dat zprávy se provádí pomocí kódování délky spuštění.

MQCOMPRESS_ZLIBFAST
Komprese dat zprávy se provádí pomocí techniky komprese zlib. Preferuje se rychlá komprese.

MQCOMPRESS_ZLIBHIGH
Komprese dat zprávy se provádí pomocí techniky komprese zlib. Preferuje se vysoká úroveň komprese.

Hodnotu lze změnit uživatelskou procedurou pro zprávy odesílajícího kanálu na jednu z vyjednaných podporovaných hodnot, k nimž lze přistupovat z pole MsgCompList MQCD. To umožňuje, aby technika použitá ke komprimaci dat zprávy byla rozhodnuta pro každou zprávu na základě obsahu zprávy. Změněná hodnota se použije pouze pro aktuální zprávu. Kanál se ukončí, pokud je atribut změněn na nepodporovanou hodnotu. Hodnota se ignoruje, pokud se změní mimo uživatelskou proceduru pro zprávu odesílajícího kanálu.

Toto je vstupní/výstupní pole pro ukončení. Pole není k dispozici, je-li hodnota *Version* menší než hodnota MQCXP_VERSION_6.

Hconn (MQHCONN)
Toto pole určuje manipulátor připojení, který uživatelská procedura používá v případě, že potřebuje provést volání MQI v rámci uživatelské procedury.

Toto pole není relevantní pro ukončení běžící na kanálech připojení klienta, kde obsahuje hodnotu MQHC_UNUSABLE_HCONN (-1).

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCXP_VERSION_7.

SharingConversations (MQBOOL)
Toto pole určuje, zda je konverzace jedinou konverzací, která může být v současné době spuštěna na této instanci kanálu, nebo zda může být v této instanci kanálu spuštěna více konverzací.

Také označuje, zda je uživatelský program vystaven riziku, že MQCD bude pozměněn jiným uživatelským programem spuštěným ve stejnou dobu.

Toto pole je relevantní pouze pro uživatelské programy spuštěné na kanálech připojení klienta nebo připojení serveru.

Je nastaven na jednu z následujících hodnot:

NEPRAVDA
Instance uživatelské procedury je jedinou instancí uživatelské procedury, která může být v současné době spuštěna na této instanci kanálu. To umožňuje uživatelské proceduře bezpečně aktualizovat pole MQCD bez soupeření z jiných uživatelských procedur spuštěných na jiných instancích kanálu. Zda jsou změny polí MQCD provedeny kanálem, je definováno tabulkou polí MQCD v souboru "Změna polí MQCD v uživatelské proceduře kanálu" na stránce 1509.

PRAVDA
Instance uživatelské procedury není jedinou instancí uživatelské procedury, která může být aktuálně spuštěna na této instanci kanálu. Všechny změny provedené v MQCD nejsou ovlivněny kanálem, s výjimkou změn uvedených v tabulce polí MQCD v části "Změna polí MQCD v uživatelské proceduře kanálu" na stránce 1509 pro příčiny ukončení jiné než MQXR_INIT. Pokud tato uživatelská procedura aktualizuje pole MQCD, zajistěte, aby nedocházelo k soupeření o další uživatelské procedury spuštěné v jiných konverzacích současně tím, že zajistíte serializaci mezi uživatelskými procedurami spuštěnými v této instanci kanálu.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud je hodnota *Version* menší než MQCXP_VERSION_7.

MCAUserSource (MQLONG)
Toto pole určuje zdroj poskytnutého ID uživatele MCA.

Může obsahovat jednu z následujících hodnot:

MQUSRC_MAP
ID uživatele je uvedeno v atributu MCAUSER.

MQUSRC_CHANNEL
ID uživatele pochází z příchozího partnera nebo je zadáno v poli MCAUSER definovaném v objektu kanálu.

Toto je vstupní pole pro ukončení. Pole není k dispozici, pokud je verze nižší než MQCXP_VERSION_8.

Body pEntry(PMQIEP)
Toto pole určuje adresu vstupního bodu rozhraní pro volání MQI nebo DCI.

Pole není k dispozici, pokud je hodnota *Verze* menší než hodnota MQCXP_VERSION_8.

RemoteProduct (MQCHAR4)
Toto pole určuje název vzdáleného produktu.

Toto pole identifikuje vzdálený produkt klienta, například C nebo Java, jak je zobrazeno v poli **RPRODUCT** DISPLAY CHSATUS.

Pole není k dispozici, pokud je hodnota *Verze* menší než hodnota MQCXP_VERSION_9.

RemoteVersion (MQCHAR8)
Toto pole určuje název vzdálené verze.

Toto pole identifikuje verzi knihoven klienta, jak je zobrazeno v poli **RVERSION** DISPLAY CHSTATUS.

Pole není k dispozici, pokud je hodnota *Verze* menší než hodnota MQCXP_VERSION_9.

C prohlášení

Tato deklarace je deklarací jazyka C pro strukturu 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 TLS certificate */
  MQLONG    SSLRemCertIssNameLength;   /* Length of
                                  distinguished name of issuer
                                  of remote TLS certificate */
  MQPTR     SSLRemCertIssNamePtr;      /* Address of
                                  distinguished name of issuer
                                  of remote TLS certificate */
  PMQVOID   SecurityParms;     /* Security parameters */
  MQLONG    CurHdrCompression; /* Header data compression
                                  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 */
  MQCHAR4   RemoteProduct;     /* The identifier for the remote product */
  MQCHAR8   RemoteVersion;     /* The version of the remote product */
  /* Ver:9 */
};
```

Deklarace jazyka COBOL

Tato deklarace je deklarací jazyka COBOL pro strukturu 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 TLS
** certificate
   15  MQCXP-SSLREMCERTISSNAMELENGTH PIC S9(9) BINARY.
** Address of distinguished name of issuer of remote TLS
** certificate
   15  MQCXP-SSLREMCERTISSNAMEPTR    POINTER.
** Security parameters
   15  MQCXP-SECURITYPARMS           PIC S9(18) BINARY.
** Header data compression used for current message
   15  MQCXP-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.
** Identifier of the remote product
   15 MQCXP-RPRODUCT                 PIC X(4).
** Identifier of the remote version
   15 MQCXP-RVERSION                 PIC X(8).
```

Deklarace RPG (ILE)

Tato deklarace je deklarací RPG pro strukturu 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 TLS certificate
     D  CXSSLCU              161    172
     D* Length of distinguished name of issuer of remote TLS certificate
     D  CXSRCINL             173    176I 0
     D* Address of distinguished name of issuer of remote TLS certificate
     D  CXSRCINP             177    192*
     D* Security parameters
     D  CXSECP               193    208*
     D* Header data compression used for current message
     D  CXCHC                209    212I 0
     D* Message data compression used for current message
     D  CXCMC                213    216I 0
     D* Connection handle
     D  CXHCONN               217    220I   0
     D* Multiple conversations possible on channel instance?
     D  CXSHARECONV           221    224I   0
     D* Source of the provided MCA user ID
     D  MCAUSERSOURCE         225    228I   0
     D* Identifer of the remote product
     D  CXRPRO                229    232I   0
     D* Identifier of the remote version
     D  CXRVER                233    240I   0
```

System/390 deklarace modulu asembleru

Tato deklarace je deklarací modulu sestavení System/390 pro strukturu 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 TLS certificate
```

```
MQCXP_SSLREMCERTISSNAMELENGTH   DS   F     Length of distinguished name
*                                 of issuer of remote TLS certificate
MQCXP_SSLREMCERTISSNAMEPTR      DS   F     Address of distinguished name
*                                 of issuer of remote TLS certificate
MQCXP_SECURITYPARMS     DS   F    Address of security parameters
MQCXP_CURHDRCOMPRESSION         DS   F     Header data compression used for
*                                 current message
MQCXP_CURMSGCOMPRESSION         DS   F      Message data compression used for
*                                 current message
MQCXP_HCONN             DS   F Connection handle
MQCXP_SHARINGCONVERSATIONS      DS   F Multiple conversations possible on
*                                 channel inst?
MQCXP_MCAUSERSOURCE     DS   F    Source of the provided MCA user ID
MQCXP_RPRODUCT          DS   CL4  Identifer of the remote product
MQCXP_RVERSION          DS   CL8  Identifer of the remote version

MQCXP_LENGTH            EQU  *-MQCXP
                        ORG  MQCXP
MQCXP_AREA              DS   CL(MQCXP_LENGTH)
```

MQXWD-deskriptor čekání na ukončení

Struktura MQXWD je vstupní/výstupní parametr volání MQXWAIT.

Tato struktura je podporována pouze na systému z/OS.

Související odkazy

"Pole" na stránce 1527
Toto téma obsahuje seznam všech polí ve struktuře MQXWD a popisuje jednotlivá pole.

"C prohlášení" na stránce 1528
Tato deklarace je deklarací jazyka C pro strukturu MQXWD.

"System/390 deklarace modulu asembleru" na stránce 1528
Tato deklarace je deklarací modulu sestavení System/390 pro strukturu MQXWD.

Pole

Toto téma obsahuje seznam všech polí ve struktuře MQXWD a popisuje jednotlivá pole.

StrucId (MQCHAR4)
Toto pole uvádí identifikátor struktury.

Hodnota musí být:

MQXWD_STRUC_ID
Identifikátor pro strukturu deskriptoru čekání na ukončení.

Pro programovací jazyk C je definována také konstanta MQXWD_STRUC_ID_ARRAY; tato konstanta má stejnou hodnotu jako MQXWD_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQXWD_STRUC_ID.

Verze (MQLONG)
Toto pole určuje číslo verze struktury.

Hodnota musí být:

MQXWD_VERSION_1
Číslo verze pro strukturu deskriptoru čekání na ukončení.

Počáteční hodnota tohoto pole je MQXWD_VERSION_1.

Reserved1 (MQLONG)
Toto pole je vyhrazeno. Jeho hodnota musí být nula.

Toto je vstupní pole.

Reserved2 (MQLONG)
Toto pole je vyhrazeno. Jeho hodnota musí být nula.

Toto je vstupní pole.

Reserved3 (MQLONG)
Toto pole je vyhrazeno. Jeho hodnota musí být nula.

Toto je vstupní pole.

ECB (MQLONG)
Toto pole určuje řídicí blok události, na který se má čekat.

Toto pole je řídicí blok události (ECB), na který se má čekat. Před vydáním volání MQXWAIT musí být nastavena na nulu; po úspěšném dokončení obsahuje poštovní kód.

Toto pole je vstupní/výstupní pole.

C prohlášení
Tato deklarace je deklarací jazyka C pro strukturu 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 */
};
```

System/390 deklarace modulu asembleru
Tato deklarace je deklarací modulu sestavení System/390 pro strukturu 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)
```

Volání ukončení pracovní zátěže klastru a datové struktury

Tento oddíl poskytuje referenční informace pro uživatelskou proceduru pracovní zátěže klastru a datové struktury. Jedná se o obecné informace o programovacím rozhraní.

Uživatelské procedury pracovní zátěže klastru můžete zapsat v následujících programovacích jazycích:

- C
- System/390 assembleru (IBM MQ for z/OS)

Volání je popsáno v:

- “MQ_CLUSTER_WORKLOAD_EXIT -Popis volání” na stránce 1529

Datové typy struktury používané uživatelskou procedurou jsou popsány v:

- “MQXCLWLN -Procházet záznamy pracovní zátěže klastru” na stránce 1531
- “MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru” na stránce 1534
- “MQWDR-Struktura záznamu cíle pracovní zátěže klastru” na stránce 1543
- “MQWQR -Struktura záznamu fronty pracovní zátěže klastru” na stránce 1547
- “MQWCR -Struktura záznamu klastru pracovní zátěže klastru” na stránce 1552

- z/OS Asynchronní chování příkazů CLUSTER v systému z/OS

V rámci této sekce jsou atributy správce front a atributy front zobrazeny v plném rozsahu. Ekvivalentní názvy použité v příkazech MQSC jsou uvedeny níže. Podrobnosti o příkazech MQSC naleznete v tématu Příkazy MQSC.

Tabulka 824. Atributy správce front	
Celé jméno	**Název použitý v MQSC**
ClusterWorkloadData	CLWLDATA
ClusterWorkloadExit	CLWLEXIT
ClusterWorkloadLength	CLWLLEN

Tabulka 825. Atributy fronty	
Celé jméno	**Název použitý v MQSC**
DefBind	DEFBIND
DefPersistence	DEFPSIST
DefPriority	DEFPRTY
InhibitPut	PUT
QDesc	DESCR

Související úlohy
Zápis a kompilace uživatelských procedur pracovní zátěže klastru

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání

Uživatelská procedura pracovní zátěže klastru je volána správcem front za účelem směrování zprávy do dostupného správce front.

Poznámka: Správce front neposkytl žádný vstupní bod s názvem `MQ_CLUSTER_WORKLOAD_EXIT` . Místo toho je název uživatelské procedury pracovní zátěže klastru definován atributem správce front `ClusterWorkloadExit` .

Uživatelská procedura `MQ_CLUSTER_WORKLOAD_EXIT` je podporována na všech platformách.

Syntaxe

```
MQ_CLUSTER_WORKLOAD_EXIT (ExitParms)
```

Související odkazy
MQXCLWLN -Procházet záznamy pracovní zátěže klastru
Volání `MQXCLWLN` se používá k navigaci v řetězcích záznamů `MQWDR`, `MQWQR`a `MQWCR` uložených v mezipaměti klastru.

MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole v `MQWXP` -Struktura parametrů uživatelské procedury pracovní zátěže klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu cíle pracovní zátěže klastru `MQWDR` .

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu fronty pracovní zátěže klastru `MQWQR` .

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu pracovní zátěže klastru MQWCR .

Parametry pro MQ_CLUSTER_WORKLOAD_EXIT

Popis parametrů ve volání MQ_CLUSTER_WORKLOAD_EXIT .

***ExitParms* (MQWXP) -vstupní/výstupní**
Blok parametrů ukončení.

- Uživatelská procedura nastaví informace v systému MQWXP , aby označila, jak spravovat pracovní zátěž.

Související odkazy
Poznámky k použití
Funkce prováděná uživatelskou procedurou pracovní zátěže klastru je definována poskytovatelem uživatelské procedury. Uživatelská procedura však musí být v souladu s pravidly definovanými v přidruženém řídicím bloku MQWXP.

Jazyková vyvolání pro MQ_CLUSTER_WORKLOAD_EXIT
Produkt MQ_CLUSTER_WORKLOAD_EXIT podporuje dva jazyky, C a High Level Assembler.

Poznámky k použití

Funkce prováděná uživatelskou procedurou pracovní zátěže klastru je definována poskytovatelem uživatelské procedury. Uživatelská procedura však musí být v souladu s pravidly definovanými v přidruženém řídicím bloku MQWXP.

Správce front neposkytl žádný vstupní bod s názvem MQ_CLUSTER_WORKLOAD_EXIT . Pro název MQ_CLUSTER_WORKLOAD_EXIT v programovacím jazyce C je však k dispozici položka typedef . Pomocí funkce typedef deklarujte uživatelskou proceduru zapsanou uživatelem, abyste se ujistili, že jsou parametry správné.

Související odkazy
Parametry pro MQ_CLUSTER_WORKLOAD_EXIT
Popis parametrů ve volání MQ_CLUSTER_WORKLOAD_EXIT .

Jazyková vyvolání pro MQ_CLUSTER_WORKLOAD_EXIT
Produkt MQ_CLUSTER_WORKLOAD_EXIT podporuje dva jazyky, C a High Level Assembler.

Jazyková vyvolání pro MQ_CLUSTER_WORKLOAD_EXIT

Produkt MQ_CLUSTER_WORKLOAD_EXIT podporuje dva jazyky, C a High Level Assembler.

Vyvolání jazyka C

```
MQ_CLUSTER_WORKLOAD_EXIT (&ExitParms);
```

Proměnnou *MQ_CLUSTER_WORKLOAD_EXIT* nahraďte názvem funkce uživatelské procedury pracovní zátěže klastru.

Deklarujte parametry **MQ_CLUSTER_WORKLOAD_EXIT** takto:

```
MQWXP ExitParms; /* Exit parameter block */
```

Vyvolání High Level Assembler

```
CALL EXITNAME,(EXITPARMS)
```

Deklarujte parametry následujícím způsobem:

```
EXITPARMS      CMQWXPA       Exit parameter block
```

Související odkazy

Parametry pro MQ_CLUSTER_WORKLOAD_EXIT
Popis parametrů ve volání MQ_CLUSTER_WORKLOAD_EXIT .

Poznámky k použití
Funkce prováděná uživatelskou procedurou pracovní zátěže klastru je definována poskytovatelem uživatelské procedury. Uživatelská procedura však musí být v souladu s pravidly definovanými v přidruženém řídicím bloku MQWXP.

MQXCLWLN -Procházet záznamy pracovní zátěže klastru

Volání MQXCLWLN se používá k navigaci v řetězcích záznamů MQWDR, MQWQRa MQWCR uložených v mezipaměti klastru.

Mezipaměť klastru je oblast hlavního úložiště, která se používá k ukládání informací souvisejících s klastrem.

Pokud je mezipaměť klastru statická, má pevnou velikost. Nastavíte-li ji na dynamickou, mezipaměť klastru se může podle potřeby rozbalit.

Pomocí systémového parametru nebo makra nastavte typ mezipaměti klastru na hodnotu STATIC nebo DYNAMIC .

- Multi Použijte systémový parametr ClusterCacheTyp na systému Multiplatforms.
- z/OS Použijte parametr CLCACHE v makru CSQ6SYSP na z/OS.

Syntaxe

```
MQXCLWLN (ExitParms, CurrentRecord, NextOffset, NextRecord, Compcode, Reason)
```

Související odkazy

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání
Uživatelská procedura pracovní zátěže klastru je volána správcem front za účelem směrování zprávy do dostupného správce front.

MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu cíle pracovní zátěže klastru MQWDR .

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu fronty pracovní zátěže klastru MQWQR .

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu pracovní zátěže klastru MQWCR .

Parametry pro MQXCLWLN -Procházet záznamy pracovní zátěže klastru

Popis parametrů ve volání MQXCLWLN .

ExitParms (MQWXP) -vstupní/výstupní
Blok parametrů ukončení.

Tato struktura obsahuje informace týkající se vyvolání uživatelské procedury. Uživatelská procedura nastaví informace v této struktuře, aby označila, jak spravovat pracovní zátěž.

CurrentRecord (MQPTR) -vstupní
Adresa aktuálního záznamu.

Tato struktura obsahuje informace týkající se adresy záznamu, který je momentálně kontrolován výstupem. Záznam musí být jedním z následujících typů:

- Záznam cíle pracovní zátěže klastru (MQWDR)
- Záznam fronty pracovní zátěže klastru (MQWQR)
- Záznam klastru pracovní zátěže klastru (MQWCR)

NextOffset (MQLONG) -vstupní
Posunutí dalšího záznamu.

Tato struktura obsahuje informace týkající se posunutí dalšího záznamu nebo struktury. *NextOffset* je hodnota odpovídajícího pole offsetu v aktuálním záznamu a musí se jednat o jedno z následujících polí:

- ChannelDefOffset v poli MQWDR
- ClusterRecOffset v tabulce MQWDR
- ClusterRecClusterRec v poli MQWQR
- ClusterRecClusterRec v poli MQWCR

NextRecord (MQPTR) -výstup
Adresa dalšího záznamu nebo struktury.

Tato struktura obsahuje informace týkající se adresy dalšího záznamu nebo struktury. Je-li *CurrentRecord* adresou MQWDRa *NextOffset* je hodnotou pole ChannelDefOffset , *NextRecord* je adresou struktury definice kanálu (MQCD).

Pokud neexistuje další záznam nebo struktura, správce front nastaví ukazatel *NextRecord* na hodnotu null a volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_NO_RECORD_AVAILABLE.

CompCode (MQLONG) -výstup
Kód dokončení.

Kód dokončení má jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_WARNING
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG) -výstup
Kvalifikující kód příčiny CompCode

Je-li CompCode MQCC_OK:

MQRC_NONE
(0, X'0000')
Není důvod se hlásit.

Pokud je *CompCode* MQCC_WARNING:

MQRC_NO_RECORD_AVAILABLE
(2359, X'0937')
Není k dispozici žádný záznam. Bylo vydáno volání MQXCLWLN z uživatelské procedury pracovní zátěže klastru pro získání adresy dalšího záznamu v řetězci. Aktuální záznam je posledním záznamem v řetězci. Nápravná akce: Žádná.

Pokud je *CompCode* MQCC_FAILED:

MQRC_CURRENT_RECORD_ERROR
(2357, X'0935')
Parametr **CurrentRecord** není platný. Bylo vydáno volání MQXCLWLN z uživatelské procedury pracovní zátěže klastru pro získání adresy dalšího záznamu v řetězci. Adresa uvedená parametrem **CurrentRecord** není adresou platného záznamu.

CurrentRecord musí být adresa cílového záznamu, MQWDR, záznamu fronty (MQWQR) nebo záznamu klastru (MQWCR) které se nacházejí v mezipaměti klastru. Nápravná akce: Zkontrolujte, zda uživatelská procedura pracovní zátěže klastru předává adresu platného záznamu umístěného v mezipaměti klastru.

MQRC_ENVIRONMENT_ERROR
(2012, X'07DC')
Volání není platné v prostředí. Bylo vydáno volání MQXCLWLN , nikoli však z uživatelské procedury pracovní zátěže klastru.

MQRC_NEXT_OFFSET_ERROR
(2358, X'0936')
Parametr **NextOffset** není platný. Bylo vydáno volání MQXCLWLN z uživatelské procedury pracovní zátěže klastru pro získání adresy dalšího záznamu v řetězci. Posunutí zadané parametrem **NextOffset** není platné. **NextOffset** musí být hodnota jednoho z následujících polí:

- ChannelDefOffset v poli MQWDR
- ClusterRecOffset v tabulce MQWDR
- ClusterRecClusterRec v poli MQWQR
- ClusterRecClusterRec v poli MQWCR

Nápravná akce: Ujistěte se, že hodnota uvedená pro parametr **NextOffset** je hodnota jednoho z polí uvedených dříve.

MQRC_NEXT_RECORD_ERROR
(2361, X'0939')
Parametr **NextRecord** není platný.

MQRC_WXP_ERROR
(2356, X'0934')
Struktura parametrů uživatelské procedury pracovní zátěže není platná. Bylo vydáno volání MQXCLWLN z uživatelské procedury pracovní zátěže klastru pro získání adresy dalšího záznamu v řetězci. Struktura parametru uživatelské procedury pracovní zátěže **ExitParms** není platná z jednoho z následujících důvodů:

- Ukazatel parametru není platný. Není vždy možné zjistit ukazatele parametrů, které nejsou platné; pokud nejsou zjištěny, dojde k nepředvídatelným výsledkům.
- Pole StrucId není MQWXP_STRUC_ID.
- Pole Verze není MQWXP_VERSION_2.
- Pole Kontext neobsahuje hodnotu předanou uživatelské proceduře správcem front.

Nápravná akce: Ujistěte se, že parametr uvedený pro **ExitParms** je struktura MQWXP , která byla předána uživatelské proceduře při vyvolání uživatelské procedury.

Související odkazy
Poznámky k použití pro MQXCLWLN-Procházet záznamy pracovní zátěže klastru
Pomocí funkce MQXCLWLN můžete procházet záznamy klastru, i když je mezipaměť statická.

Vyvolání jazyka MQXCLWLN
Produkt MQXCLWLN podporuje dva jazyky: C a High Level Assembler.

Poznámky k použití pro MQXCLWLN-Procházet záznamy pracovní zátěže klastru
Pomocí funkce MQXCLWLN můžete procházet záznamy klastru, i když je mezipaměť statická.

Pokud je mezipaměť klastru dynamická, musí se k navigaci v záznamech použít volání MQXCLWLN . Ukončení skončí abnormálně, pokud se k navigaci v záznamech použije jednoduchá aritmetika ukazatele-a-posunutí.

Pokud je mezipaměť klastru statická, MQXCLWLN nemusí být použita pro navigaci v záznamech. Obvykle se používá MQXCLWLN , i když je mezipaměť statická. Poté můžete mezipaměť klastru změnit na dynamickou, aniž byste museli měnit uživatelskou proceduru pracovní zátěže.

Související odkazy

Parametry pro MQXCLWLN -Procházet záznamy pracovní zátěže klastru
Popis parametrů ve volání MQXCLWLN .

Vyvolání jazyka MQXCLWLN
Produkt MQXCLWLN podporuje dva jazyky: C a High Level Assembler.

Vyvolání jazyka MQXCLWLN

Produkt MQXCLWLN podporuje dva jazyky: C a High Level Assembler.

Vyvolání jazyka C

```
MQXCLWLN (&ExitParms, CurrentRecord, NextOffset, &NextRecord, &CompCode, &Reason) ;
```

Deklarujte parametry následujícím způsobem:

```
Typedef struct tagMQXCLWLN {
MQWXP   ExitParms;      /* Exit parameter block */
MQPTR   CurrentRecord;  /* Address of current record*/
MQLONG  NextOffset;     /* Offset of next record */
MQPTR   NextRecord;     /* Address of next record or structure */
MQLONG  CompCode;       /* Completion code */
MQLONG  Reason;         /* Reason code qualifying CompCode */
```

Vyvolání High Level Assembler

```
CALL MQXCLWLN,(CLWLEXITPARMS,CURRENTRECORD,NEXTOFFSET,NEXTRECORD,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
CLWLEXITPARMS CMQWXPA, Cluster workload exit parameter block
CURRENTRECORD CMQWDRA, Current record
NEXTOFFSET    DS F     Next offset
NEXTRECORD    DS F     Next record
COMPCODE      DS F     Completion code
REASON        DS F     Reason code qualifying COMPCODE
```

Související odkazy

Parametry pro MQXCLWLN -Procházet záznamy pracovní zátěže klastru
Popis parametrů ve volání MQXCLWLN .

Poznámky k použití pro MQXCLWLN-Procházet záznamy pracovní zátěže klastru
Pomocí funkce MQXCLWLN můžete procházet záznamy klastru, i když je mezipaměť statická.

MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru

Následující tabulka shrnuje pole v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru.

Tabulka 826. Pole v systému MQWXP		
Pole	**Popis**	**Stránka**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
ExitId	Typ ukončení	ExitId
ExitReason	Důvod vyvolání uživatelské procedury	ExitReason
ExitResponse	Odezva z ukončení	ExitResponse
ExitResponse2	Sekundární odezva z ukončení	ExitResponse2
Feedback	Kód zpětné vazby	Zpětná vazba
Flags	Hodnoty příznaků. Tyto bitové příznaky se používají k označení informací o vkládané zprávě	Příznaky
ExitUserArea	Ukončit uživatelskou oblast	ExitUserOblast
ExitData	Data uživatelské procedury	ExitData
MsgDescPtr	Adresa deskriptoru zprávy (MQMD)	MsgDescPtr
MsgBufferPtr	Adresa vyrovnávací paměti obsahující některá nebo všechna data zprávy	MsgBufferPtr
MsgBufferLength	Délka vyrovnávací paměti obsahující data zprávy	MsgBuffer
MsgLength	Délka úplné zprávy	MsgLength
QName	Název fronty	QName
QMgrName	Název lokálního správce front	QMgrName
DestinationCount	Počet možných cílů	DestinationCount
DestinationChosen	Zvolený cíl	DestinationChosen
DestinationArrayPtr	Adresa pole ukazatelů na cílové záznamy (MQWDR)	DestinationArrayPtr
QArrayPtr	Adresa pole ukazatelů na záznamy fronty (MQWQR)	QArrayPtr
Poznámka: Zbývající pole jsou ignorována, pokud je verze menší než MQWXP_VERSION_2.		
CacheContext	Informace o kontextu	CacheContext
CacheType	Typ mezipaměti klastru	CacheType
Poznámka: Zbývající pole jsou ignorována, pokud je verze menší než MQWXP_VERSION_3.		
CLWLMRUChannels	Maximální počet povolených aktivních odchozích kanálů klastru	CLWLMRUChannels
Poznámka: Zbývající pole jsou ignorována, pokud je verze menší než MQWXP_VERSION_4.		
pEntryPoints	Adresa struktury MQIEP pro povolení volání MQI a DCI	pEntryBody

Struktura parametrů uživatelské procedury pracovní zátěže klastru popisuje informace, které jsou předány uživatelské proceduře pracovní zátěže klastru.

Struktura parametrů uživatelské procedury pracovní zátěže klastru je podporována na všech platformách.

Dále jsou k dispozici struktury MQWXP1, MQWXP2 a MQWXP3 pro zpětnou kompatibilitu.

Související odkazy

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání
Uživatelská procedura pracovní zátěže klastru je volána správcem front za účelem směrování zprávy do dostupného správce front.

MQXCLWLN -Procházet záznamy pracovní zátěže klastru
Volání MQXCLWLN se používá k navigaci v řetězcích záznamů MQWDR, MQWQRa MQWCR uložených v mezipaměti klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu cíle pracovní zátěže klastru MQWDR .

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu fronty pracovní zátěže klastru MQWQR .

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu pracovní zátěže klastru MQWCR .

Pole v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru

Popis polí v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru

StrucId (MQCHAR4)-vstup
Identifikátor struktury pro strukturu parametru uživatelské procedury pracovní zátěže klastru.

- Hodnota StrucId je MQWXP_STRUC_ID.
- Pro programovací jazyk C je definována také konstanta MQWXP_STRUC_ID_ARRAY . Má stejnou hodnotu jako MQWXP_STRUC_ID. Jedná se o pole znaků namísto řetězce.

Verze (MQLONG)-vstup
Označuje číslo verze struktury. Verze má jednu z následujících hodnot:

MQWXP_VERSION_1
Struktura parametrů uživatelské procedury pracovní zátěže klastru Version-1 .

Produkt MQWXP_VERSION_1 je podporován ve všech prostředích.

MQWXP_VERSION_2
Struktura parametrů uživatelské procedury pracovní zátěže klastru Version-2 .

Produkt MQWXP_VERSION_2 je podporován v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

MQWXP_VERSION_3
Struktura parametrů uživatelské procedury pracovní zátěže klastru Version-3 .

Produkt MQWXP_VERSION_3 je podporován v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

MQWXP_VERSION_4
Struktura parametrů uživatelské procedury pracovní zátěže klastru Version-4 .

Produkt MQWXP_VERSION_4 je podporován v následujících prostředích:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows

MQWXP_CURRENT_VERSION
Aktuální verze struktury parametrů uživatelské procedury pracovní zátěže klastru.

ExitId (MQLONG)-vstup
Označuje typ volaného ukončení. Uživatelská procedura pracovní zátěže klastru je jedinou podporovanou uživatelskou procedurou.

- Hodnota `ExitId` musí být `MQXT_CLUSTER_WORKLOAD_EXIT`

ExitReason (MQLONG)-vstup
Označuje příčinu vyvolání uživatelské procedury pracovní zátěže klastru. `ExitReason` má jednu z následujících hodnot:

MQXR_INIT
Označuje, že uživatelská procedura je vyvolávána poprvé.

Získejte a inicializujte všechny prostředky, které může uživatelská procedura potřebovat, například hlavní paměť.

MQXR_TERM
Označuje, že uživatelská procedura bude ukončena.

Uvolněte všechny prostředky, které uživatelská procedura získala od své inicializace, například hlavní paměť.

MQXR_CLWL_OPEN
Voláno pomocí `MQOPEN`.

MQXR_CLWL_PUT
Voláno pomocí `MQPUT` nebo `MQPUT1`.

MQXR_CLWL_MOVE
Voláno agentem MCA při změně stavu kanálu.

MQXR_CLWL_REPOS
Voláno `MQPUT` nebo `MQPUT1` pro zprávu PCF správce úložiště.

MQXR_CLWL_REPOS_MOVE
Voláno agentem MCA pro zprávu PCF správce úložiště, pokud se stav kanálu změnil.

ExitResponse (MQLONG)-výstup
Nastavte `ExitResponse` , abyste označili, zda zpracování zprávy pokračuje. Musí to být jedna z následujících hodnot:

MQXCC_OK
Pokračujte v normálním zpracování zprávy.

- `DestinationChosen` identifikuje místo určení, do kterého má být zpráva odeslána.

MQXCC_SUPPRESS_FUNCTION
Ukončit zpracování zprávy.

- Akce provedené správcem front závisí na příčině vyvolání uživatelské procedury:

Tabulka 827. Akce prováděné správcem front	
ExitReason	**Přijatá opatření**
– MQXR_CLWL_OPEN – MQXR_CLWL_REPOS – MQXR_CLWL_PUT	Volání MQOPEN, MQPUTnebo MQPUT1 se nezdařilo s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_STOPPED_BY_CLUSTER_EXIT.
– MQXR_CLWL_MOVE – MQXR_CLWL_REPOS_MOVE	Zpráva je umístěna do fronty nedoručených zpráv.

MQXCC_SUPPRESS_EXIT
Pokračovat v běžném zpracování aktuální zprávy. Nevyvolávejte uživatelskou proceduru znovu, dokud se správce front nevypne.

Správce front zpracovává následné zprávy, jako by byl atribut správce front ClusterWorkloadExit prázdný. DestinationChosen identifikuje místo určení, kam je odeslána aktuální zpráva.

Jakákoli jiná hodnota
Zpracujte zprávu, jako by byl zadán parametr MQXCC_SUPPRESS_FUNCTION .

ExitResponse2 (MQLONG)-vstupní/výstupní
Nastavte volbu ExitResponse2 , chcete-li správci front poskytnout další informace.

- MQXR2_STATIC_CACHE je výchozí hodnota a je nastavena při vstupu do uživatelské procedury.
- Když má ExitReason hodnotu MQXR_INIT, uživatelská procedura může nastavit jednu z následujících hodnot v ExitResponse2:

  MQXR2_STATIC_CACHE
  Uživatelská procedura vyžaduje statickou mezipaměť klastru.

  – Pokud je mezipaměť klastru statická, uživatelská procedura nemusí používat volání MQXCLWLN k navigaci řetězy záznamů v mezipaměti klastru.

  – Pokud je mezipaměť klastru dynamická, uživatelská procedura nemůže správně procházet záznamy v mezipaměti.

  Poznámka: Správce front zpracuje návrat z volání MQXR_INIT , jako by uživatelská procedura vrátila hodnotu MQXCC_SUPPRESS_EXIT v poli ExitResponse .

  MQXR2_DYNAMIC_CACHE
  Uživatelská procedura může pracovat se statickou nebo dynamickou mezipamětí.

  – Pokud uživatelská procedura vrátí tuto hodnotu, musí uživatelská procedura použít volání MQXCLWLN k navigaci řetězy záznamů v mezipaměti klastru.

Feedback (MQLONG)-vstup
Vyhrazené pole. Hodnota je nula.

Příznaky (MQLONG)-vstup
Označuje informace o vkládané zprávě.

- Hodnota Příznaky je MQWXP_PUT_BY_CLUSTER_CHL. Zpráva pochází z kanálu klastru, nikoli lokálně nebo z kanálu mimo klastr. Jinými slovy, zpráva pochází od jiného správce front klastru.

Vyhrazeno (MQLONG)-vstup
Vyhrazené pole. Hodnota je nula.

ExitUserOblast (MQBYTE16)-vstupní/výstupní
Nastavte oblast ExitUser tak, aby komunikovala mezi voláními do uživatelské procedury.

- ExitUserOblast je inicializována na binární nulu před prvním vyvoláním uživatelské procedury. Veškeré změny provedené v tomto poli uživatelskou procedurou jsou zachovány v rámci vyvolání

uživatelské procedury, které se vyskytnou mezi voláním MQCONN a odpovídajícím voláním funkce MQDISC . Když se vyskytne volání MQDISC , pole se resetuje na binární nulu.

- První vyvolání uživatelské procedury je označeno polem ExitReason s hodnotou MQXR_INIT.
- Jsou definovány tyto konstanty:

  MQXUA_NONE -řetězec
  MQXUA_NONE_ARRAY -znakové pole
  Žádné informace o uživateli. Obě konstanty jsou binární nula pro délku pole.

  MQ_EXIT_USER_AREA_LENGTH
  Délka oblasti ExitUser.

ExitData (MQCHAR32)-vstup
Hodnota atributu správce front ClusterWorkloadData . Pokud pro příslušný atribut nebyla definována žádná hodnota, bude toto pole obsahovat pouze prázdné znaky.

- Délka řetězce ExitData je dána MQ_EXIT_DATA_LENGTH.

MsgDescPtr (PMQMD)-vstup
Adresa kopie deskriptoru zprávy (MQMD) pro zpracovávanou zprávu.

- Všechny změny provedené v deskriptoru zpráv uživatelskou procedurou jsou správcem front ignorovány.
- Pokud má parametr ExitReason jednu z následujících hodnot, MsgDescPtr je nastaven na ukazatel Null a do uživatelské procedury není předán žádný deskriptor zprávy:
  - MQXR_INIT
  - MQXR_TERM
  - MQXR_CLWL_OPEN

MsgBufferPtr (PMQVOID)-vstup
Adresa vyrovnávací paměti obsahující kopii prvních MsgBufferDélka bajtů dat zprávy.

- Veškeré změny dat zprávy provedené uživatelskou procedurou jsou správcem front ignorovány.
- Žádná data zprávy nejsou předána uživatelské proceduře, když:
  - MsgDescPtr je ukazatel s hodnotou Null.
  - Zpráva neobsahuje žádná data.
  - Atribut správce front ClusterWorkloadLength je nulový.

  V těchto případech je MsgBufferPtr ukazatel Null.

MsgBufferDélka (MQLONG)-vstup
Délka vyrovnávací paměti obsahující data zprávy předaná uživatelské proceduře.

- Délka je řízena atributem správce front ClusterWorkloadLength .
- Délka může být menší než délka celé zprávy, viz MsgLength.

MsgLength (MQLONG)-vstup
Délka úplné zprávy předané uživatelské proceduře.

- MsgBuffervyrovnávací paměti zpráv může být menší než délka celé zprávy.
- MsgLength je nula, pokud ExitReason je MQXR_INIT, MQXR_TERMnebo MQXR_CLWL_OPEN.

QName (MQCHAR48)-vstup
Název cílové fronty. Fronta je fronta klastru.

- Délka QName je MQ_Q_NAME_LENGTH.

QMgrName (MQCHAR48)-vstup
Název lokálního správce front, který vyvolal uživatelskou proceduru pracovní zátěže klastru.

- Délka položky QMgrName je MQ_Q_MGR_NAME_LENGTH.

DestinationCount (MQLONG)-vstup
Počet možných cílů. Cíle jsou instance cílové fronty a jsou popsány cílovými záznamy.

- Cílový záznam je struktura MQWDR . Pro každou možnou trasu ke každé instanci fronty existuje jedna struktura.
- Struktury MQWDR jsou adresovány polem ukazatelů, viz DestinationArrayPtr.

DestinationChosen (MQLONG)-vstup/výstup
Zvolený cíl.

- Číslo struktury MQWDR , která identifikuje trasu a instanci fronty, kam se má zpráva odeslat.
- Hodnota je v rozsahu 1- DestinationCount.
- Na vstupu do uživatelské procedury DestinationChosen označuje trasu a instanci fronty, kterou vybral správce front. Uživatelská procedura může přijmout tuto volbu, nebo zvolit jinou přenosovou cestu a instanci fronty.
- Hodnota nastavená uživatelskou procedurou musí být v rozsahu 1- DestinationCount. Pokud je vrácena jakákoli jiná hodnota, správce front použije hodnotu DestinationChosen na vstupu do uživatelské procedury.

DestinationArrayPtr (PPMQWDR)-vstup
Adresa pole ukazatelů na cílové záznamy (MQWDR).

- Existují cílové záznamy DestinationCount .

QArrayPtr (PPMQWQR)-vstup
Adresa pole ukazatelů na záznamy fronty (MQWQR).

- Jsou-li k dispozici záznamy fronty, je jich DestinationCount .
- Pokud nejsou k dispozici žádné záznamy fronty, QArrayPtr je ukazatel Null.

  Poznámka: QArrayPtr může být ukazatel null, i když je DestinationCount větší než nula.

CacheContext (MQPTR): Verze 2-vstup
Pole CacheContext je vyhrazeno pro použití správcem front. Uživatelská procedura nesmí měnit hodnotu tohoto pole.

CacheType (MQLONG): Verze 2-vstup
Mezipaměť klastru má jeden z následujících typů:

MQCLCT_STATIC
Mezipaměť je statická.

- Velikost mezipaměti je pevná a při provozu správce front ji nelze zvětšit.
- K navigaci v záznamech v tomto typu mezipaměti není třeba používat volání MQXCLWLN .

MQCLCT_DYNAMIC
Mezipaměť je dynamická.

- Velikost mezipaměti se může zvýšit, aby vyhovovala různým informacím o klastru.
- Chcete-li procházet záznamy v tomto typu mezipaměti, musíte použít volání MQXCLWLN .

CLWLMRUChannels (MQLONG): Verze 3-vstup
Označuje maximální počet aktivních odchozích kanálů klastru, které mají být brány v úvahu pro použití algoritmem výběru pracovní zátěže klastru.

- CLWLMRUChannels je hodnota 1-999 999 999.

pEntryPoints (PMQIEP): Verze 4
Adresa struktury MQIEP, jejímž prostřednictvím lze provádět volání MQI a DCI.

Související odkazy
Počáteční hodnoty a deklarace jazyka pro MQWXP
Počáteční hodnoty a deklarace jazyka C a High Level Assembler pro MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru.

Počáteční hodnoty a deklarace jazyka pro MQWXP

Počáteční hodnoty a deklarace jazyka C a High Level Assembler pro MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru.

Tabulka 828. Pole v systému MQWXP

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQWXP_STRUC_ID	'WXP¬'
Version	MQWXP_VERSION_2	2
ExitId	Není	0
ExitReason	MQXCC_OK	0
ExitResponse	Není	0
ExitResponse2	Není	0
Flags	Není	0
ExitUserArea	{MQXUA_NONE_ARRAY}	0
ExitData	Není	""
MsgDescPtr	Není	NULL
MsgBufferPtr	Není	NULL
MsgBufferLength	Není	0
MsgBufferPtr	Není	0
QName	Není	""
QMgrName	Není	""
DestinationCount	Není	0
DestinationChosen	Není	0
DestinationArrayPtr	Není	NULL
QArrayPtr	Není	NULL
CacheContext	Není	NULL
CacheType	MQCLCT_DYNAMIC	1
CLWLMRUChannels	Není	0
pEntryPoints	Není	NULL

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C obsahuje proměnná makra MQWXP_DEFAULT výchozí hodnoty. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQWDR MyWXP = {MQWXP_DEFAULT};
```

C prohlášení

```
typedef struct tagMQWXP {
```

```
  MQCHAR4   StrucId;              /* Structure identifier */
  MQLONG    Version;              /* Structure version number */
  MQLONG    ExitId;               /* Type of exit */
  MQLONG    ExitReason;           /* Reason for invoking exit */
  MQLONG    ExitResponse;         /* Response from exit */
  MQLONG    ExitResponse2;        /* Reserved */
  MQLONG    Feedback;             /* Reserved */
  MQLONG    Flags;                /* Flags */
  MQBYTE16  ExitUserArea;         /* Exit user area */
  MQCHAR32  ExitData;             /* Exit data */
  PMQMD     MsgDescPtr;           /* Address of message descriptor */
  PMQVOID   MsgBufferPtr;         /* Address of buffer containing some
                                     or all of the message data */
  MQLONG    MsgBufferLength;      /* Length of buffer containing message
                                     data */
  MQLONG    MsgLength;            /* Length of complete message */
  MQCHAR48  QName;                /* Queue name */
  MQCHAR48  QMgrName;             /* Name of local queue manager */
  MQLONG    DestinationCount;     /* Number of possible destinations */
  MQLONG    DestinationChosen;    /* Destination chosen */
  PPMQWDR   DestinationArrayPtr;  /* Address of an array of pointers to
                                     destination records */
  PPMQWQR   QArrayPtr;            /* Address of an array of pointers to
                                     queue records */

  /* version 1 */
  MQPTR     CacheContext;         /* Context information */
  MQLONG    CacheType;            /* Type of cluster cache */
  /* version 2 */
  MQLONG    CLWLMRUChannnels;     /* Maximum number of most recently
                                     used cluster channels */
  /* version 3 */
  PMQIEP    pEntryPoints;         /* Address of the MQIEP structure */
  /* version 4 */
};
```

High Level Assembler

```
MQWXP                          DSECT
MQWXP_STRUCID                  DS   CL4      Structure identifier
MQWXP_VERSION                  DS   F        Structure version number
MQWXP_EXITID                   DS   F        Type of exit
MQWXP_EXITREASON               DS   F        Reason for invoking exit
MQWXP_EXITRESPONSE             DS   F        Response from exit
MQWXP_EXITRESPONSE2            DS   F        Reserved
MQWXP_FEEDBACK                 DS   F        Reserved
MQWXP_RESERVED                 DS   F        Reserved
MQWXP_EXITUSERAREA             DS   XL16     Exit user area
MQWXP_EXITDATA                 DS   CL32     Exit data
MQWXP_MSGDESCPTR               DS   F        Address of message
*                                            descriptor
MQWXP_MSGBUFFERPTR             DS   F        Address of buffer containing
*                                            some or all of the message
*                                            data
MQWXP_MSGBUFFERLENGTH          DS   F        Length of buffer containing
*                                            message data
MQWXP_MSGLENGTH                DS   F        Length of complete message
MQWXP_QNAME                    DS   CL48     Queue name
MQWXP_QMGRNAME                 DS   CL48     Name of local queue manager
MQWXP_DESTINATIONCOUNT         DS   F        Number of possible
*                                            destinations
MQWXP_DESTINATIONCHOSEN        DS   F        Destination chosen
MQWXP_DESTINATIONARRAYPTR      DS   F        Address of an array of
*                                            pointers to destination
*                                            records
MQWXP_QARRAYPTR                DS   F        Address of an array of
*                                            pointers to queue records
MQWXP_CACHECONTEXT             DS   F        Context information
MQWXP_CACHETYPE                DS   F        Type of cluster cache
MQWXP_CLWLMRUCHANNELS          DS   F        Number of most recently used
*                                            channels for workload balancing

MQWXP_LENGTH                   EQU  *-MQWXP  Length of structure
                               ORG  MQWXP
MQWXP_AREA                     DS   CL(MQWXP_LENGTH)
```

Související odkazy

Pole v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru

MQWDR-Struktura záznamu cíle pracovní zátěže klastru

Následující tabulka shrnuje pole ve struktuře záznamu cíle pracovní zátěže klastru MQWDR .

Tabulka 829. Pole v tabulce MQWDR		
Pole	**Popis**	**Stránka**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQWDR	StrucLength
QMgrFlags	Příznaky správce front	QMgrFlags
QMgrIdentifier	Identifikátor správce front	QMgrIdentifier
QMgrName	Název správce front	QMgrName
ClusterRecOffset	Logické posunutí prvního záznamu klastru (MQWCR)	ClusterRecClusterRec
ChannelState	Stav kanálu	ChannelState
ChannelDefOffset	Logické posunutí struktury definice kanálu (MQCD)	ChannelDefOdsazení
Poznámka: Zbývající pole jsou ignorována, pokud je verze menší než MQWDR_VERSION_2.		
DestSeqNumber	Pořadové číslo cíle kanálu	DestSeq
DestSeqFactor	Faktor posloupnosti místa určení kanálu pro vážení	DestSeqFaktor

Struktura cílového záznamu pracovní zátěže klastru obsahuje informace týkající se jednoho z možných míst určení pro zprávu. Pro každou instanci cílové fronty existuje jedna struktura záznamu cíle pracovní zátěže klastru.

Cílová struktura záznamu pracovní zátěže klastru je podporována ve všech prostředích.

Kromě toho jsou pro zpětnou kompatibilitu k dispozici struktury MQWDR1 a MQWDR2 .

Související odkazy

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání
Uživatelská procedura pracovní zátěže klastru je volána správcem front za účelem směrování zprávy do dostupného správce front.

MQXCLWLN -Procházet záznamy pracovní zátěže klastru
Volání MQXCLWLN se používá k navigaci v řetězcích záznamů MQWDR, MQWQRa MQWCR uložených v mezipaměti klastru.

MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru.

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu fronty pracovní zátěže klastru MQWQR .

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu pracovní zátěže klastru MQWCR .

Pole v MQWDR-Struktura záznamu cíle pracovní zátěže klastru

Popis parametrů ve struktuře záznamu cíle pracovní zátěže klastru MQWDR .

StrucId (MQCHAR4) -vstupní
Identifikátor struktury pro strukturu záznamu cíle pracovní zátěže klastru.

- Hodnota StrucId je MQWDR_STRUC_ID.
- Pro programovací jazyk C je definována také konstanta MQWDR_STRUC_ID_ARRAY . Má stejnou hodnotu jako MQWDR_STRUC_ID. Jedná se o pole znaků namísto řetězce.

Verze (MQLONG) -vstupní
Číslo verze struktury. Verze má jednu z následujících hodnot:

MQWDR_VERSION_1
Záznam cíle pracovní zátěže klastru Version-1 .

MQWDR_VERSION_2
Version-2 záznam cíle pracovní zátěže klastru.

MQWDR_CURRENT_VERSION
Aktuální verze záznamu cíle pracovní zátěže klastru.

StrucLength (MQLONG) -vstupní
Délka struktury MQWDR . StrucLength má jednu z následujících hodnot:

MQWDR_LENGTH_1
Délka záznamu cíle pracovní zátěže klastru version-1 .

MQWDR_LENGTH_2
Délka záznamu cíle pracovní zátěže klastru version-2 .

MQWDR_CURRENT_LENGTH
Délka aktuální verze záznamu cíle pracovní zátěže klastru.

QMgrFlags (MQLONG) -vstupní
Příznaky správce front označující vlastnosti správce front, který je hostitelem instance cílové fronty popsané ve struktuře MQWDR . Jsou definovány následující příznaky:

MQQMF_REPOSITORY_Q_MGR
Cíl je správce front úplného úložiště.

MQQMF_CLUSSDR_USER_DEFINED
Odesílací kanál klastru byl definován ručně.

MQQMF_CLUSSDR_AUTO_DEFINED
Odesílací kanál klastru byl definován automaticky.

MQQMF_AVAILABLE
Správce cílové fronty je k dispozici pro příjem zpráv.

Ostatní hodnoty
Další příznaky v poli mohou být nastaveny správcem front pro interní účely.

QMgrIdentifier (MQCHAR48) -vstupní
Identifikátor správce front je jedinečný identifikátor pro správce front, který je hostitelem instance cílové fronty popsané ve struktuře MQWDR .

- Identifikátor je generován správcem front.
- Délka položky QMgrIdentifier je MQ_Q_MGR_IDENTIFIER_LENGTH.

QMgrName (MQCHAR48) -vstupní
Název správce front, který je hostitelem instance cílové fronty popsané strukturou MQWDR .

- QMgrName může být název lokálního správce front i jiného správce front v klastru.
- Délka položky QMgrName je MQ_Q_MGR_NAME_LENGTH.

ClusterRecClusterRec (MQLONG) -vstupní
Logické posunutí první struktury MQWCR , která patří do struktury MQWDR .

- V případě statických mezipamětí představuje ClusterRecOffset posun první struktury MQWCR , která patří do struktury MQWDR .
- Posun je měřen v bajtech od začátku struktury MQWDR .
- Nepoužívejte logické posunutí pro aritmetiku ukazatele s dynamickými mezipaměťmi. Chcete-li získat adresu dalšího záznamu, musíte použít volání MQXCLWLN .

ChannelState (MQLONG) -vstupní
Stav kanálu, který propojuje lokálního správce front se správcem front identifikovaným strukturou MQWDR . Možné jsou následující hodnoty:

MQCHS_BINDING
Kanál vyjednává s partnerem.

MQCHS_INACTIVE
Kanál není aktivní.

MQCHS_INITIALIZING
Probíhá inicializace kanálu.

MQCHS_PAUSED
Kanál byl pozastaven.

MQCHS_REQUESTING
Žadatelský kanál požaduje připojení.

MQCHS_RETRYING
Kanál se znovu pokouší navázat připojení.

MQCHS_RUNNING
Kanál přenáší nebo čeká na zprávy.

MQCHS_STARTING
Kanál čeká na aktivaci.

MQCHS_STOPPING
Kanál se zastavuje.

MQCHS_STOPPED
Kanál byl zastaven.

ChannelDefOffset (MQLONG) -vstupní
Logické posunutí definice kanálu (MQCD) pro kanál, který propojuje lokálního správce front se správcem front identifikovaným strukturou MQWDR .

- ChannelDefOffset je jako ClusterRecOffset
- Logické posunutí nelze použít v aritmetice ukazatele. Chcete-li získat adresu dalšího záznamu, musíte použít volání MQXCLWLN .

DestSeqFaktor (MQLONG) -vstupní
Cílový faktor posloupnosti, který umožňuje výběr kanálu na základě váhy.

- DestSeqFaktor se používá před tím, než jej správce front změní.
- Správce pracovní zátěže zvyšuje hodnotu DestSeqFactor způsobem, který zajišťuje, že jsou zprávy distribuovány mimo kanály podle jejich váhy.

DestSeqČíslo (MQLONG) -vstupní
Hodnota místa určení kanálu klastru před tím, než ji správce front změní.

- Správce pracovní zátěže zvýší hodnotu DestSeqPočet při každém vložení zprávy do tohoto kanálu.
- Uživatelské procedury pracovní zátěže mohou pomocí volby DestSeqČíslo rozhodnout, který kanál má zprávu vypnout.

Související odkazy
Počáteční hodnoty a deklarace jazyka pro MQWDR
Počáteční hodnoty a deklarace jazyka C a High Level Assembler pro MQWDR -záznam cíle pracovní zátěže klastru.

Počáteční hodnoty a deklarace jazyka pro MQWDR
Počáteční hodnoty a deklarace jazyka C a High Level Assembler pro MQWDR -záznam cíle pracovní zátěže klastru.

Tabulka 830. Pole v tabulce MQWDR		
Název pole	**Název konstanty**	**Hodnota konstanty**
StrucId	MQWDR_STRUC_ID	'WDR¬'
Version	MQWDR_VERSION_1	1
StrucLength	MQWDR_CURRENT_LENGTH [3]	136
QMgrFlags	MQWDR_NONE	0
QMgrIdentifier	Není	""
QMgrName	Není	""
ClusterRecOffset	Není	0
ChannelState	Není	0
ChannelDefOffset	Není	0
DestSeqNumber	Není	0
DestSeqFactor	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C obsahuje proměnná makra MQWDR_DEFAULT výchozí hodnoty. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

   ```
   MQWDR MyWDR = {MQWDR_DEFAULT};
   ```

3. Počáteční hodnoty záměrně nastavují délku struktury na délku aktuální verze, a nikoli na verzi 1 struktury.

High Level Assembler

```
MQWDR                           DSECT
MQWDR_STRUCID                   DS   CL4      Structure identifier
MQWDR_VERSION                   DS   F        Structure version number
MQWDR_STRUCLENGTH               DS   F        Length of MQWDR structure
MQWDR_QMGRFLAGS                 DS   F        Queue manager flags
MQWDR_QMGRIDENTIFIER            DS   CL48     Queue manager identifier
MQWDR_QMGRNAME                  DS   CL48     Queue manager name
MQWDR_CLUSTERRECOFFSET          DS   F        Offset of first cluster
*                                            record
MQWDR_CHANNELSTATE              DS   F        Channel state
MQWDR_CHANNELDEFOFFSET          DS   F        Offset of channel definition
*                                            structure
MQWDR_LENGTH                    EQU  *-MQWDR  Length of structure
                                ORG  MQWDR
MQWDR_AREA                      DS   CL(MQWDR_LENGTH)
```

C prohlášení

```
typedef struct tagMQWDR {
  MQCHAR4   StrucId;           /* Structure identifier */
  MQLONG    Version;           /* Structure version number */
  MQLONG    StrucLength;       /* Length of MQWDR structure */
  MQLONG    QMgrFlags;         /* Queue managerflags */
  MQCHAR48  QMgrIdentifier;    /* Queue manageridentifier */
  MQCHAR48  QMgrName;          /* Queue manager name */
  MQLONG    ClusterRecOffset;  /* Offset of first cluster record */
  MQLONG    ChannelState;      /* Channel state */
  MQLONG    ChannelDefOffset;  /* Offset of channel definition structure */
```

```
  /* Ver:1 */
  MQLONG    DestSeqNumber;     /* Cluster channel destination sequence number */
  MQINT64   DestSeqFactor;     /* Cluster channel factor sequence number */
  /* Ver:2 */
};
```

Související odkazy

Pole v MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Popis parametrů ve struktuře záznamu cíle pracovní zátěže klastru MQWDR .

MQWQR -Struktura záznamu fronty pracovní zátěže klastru

Následující tabulka shrnuje pole ve struktuře záznamu fronty pracovní zátěže klastru MQWQR .

Tabulka 831. Pole v produktu MQWQR

Pole	Popis	Stránka
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQWQR	StrucLength
QFlags	Příznaky fronty	QFlags
QName	Název fronty	QName
QMgrIdentifier	Identifikátor správce front	QMgrIdentifier
ClusterRecOffset	Posun prvního záznamu klastru (MQWCR)	ClusterRecClusterRec
QType	Typ fronty	QTYPE
QDesc	Popis fronty	QDesc
DefBind	Výchozí vazba	DefBind
DefPersistence	Výchozí trvalost zpráv	DefPersistence
DefPriority	Výchozí priorita zpráv	DefPriority
InhibitPut	Zda jsou povoleny operace vložení do fronty	InhibitPut
Poznámka: Zbývající pole jsou ignorována, pokud je verze menší než MQWQR_VERSION_2.		
CWLQueuePriority	Hodnota 0-9 představující prioritu fronty	CLWLQueuePriority
CLWLQueueRank	Hodnota 0-9 představující ohodnocení důležitosti fronty	CLWLQueueRank
Poznámka: Zbývající pole jsou ignorována, pokud je verze menší než MQWQR_VERSION_3.		
DefPutResponse	Odezva výchozího umístění	DefPutOdezva

Struktura záznamu fronty pracovní zátěže klastru obsahuje informace související s jedním z možných cílů pro zprávu. Pro každou instanci cílové fronty existuje jedna struktura záznamu fronty pracovní zátěže klastru.

Struktura záznamu fronty pracovní zátěže klastru je podporována ve všech prostředích.

Dále jsou k dispozici struktury MQWQR1 a MQWQR2 pro zpětnou kompatibilitu.

Související odkazy

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání
Uživatelská procedura pracovní zátěže klastru je volána správcem front za účelem směrování zprávy do dostupného správce front.

MQXCLWLN -Procházet záznamy pracovní zátěže klastru
Volání MQXCLWLN se používá k navigaci v řetězcích záznamů MQWDR, MQWQRa MQWCR uložených v mezipaměti klastru.

MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu cíle pracovní zátěže klastru MQWDR .

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu pracovní zátěže klastru MQWCR .

Pole v MQWQR -Struktura záznamu fronty pracovní zátěže klastru

Popis polí ve struktuře záznamu fronty pracovní zátěže klastru MQWQR .

StrucId (MQCHAR4) -vstupní
Identifikátor struktury pro strukturu záznamu fronty pracovní zátěže klastru.

- Hodnota StrucId je MQWQR_STRUC_ID.
- Pro programovací jazyk C je definována také konstanta MQWQR_STRUC_ID_ARRAY . Má stejnou hodnotu jako MQWQR_STRUC_ID. Jedná se o pole znaků namísto řetězce.

Verze (MQLONG) -vstupní
Číslo verze struktury. Verze má jednu z následujících hodnot:

MQWQR_VERSION_1
Záznam fronty pracovní zátěže klastru Version-1 .

MQWQR_VERSION_2
Version-2 -záznam fronty pracovní zátěže klastru.

MQWQR_VERSION_3
Version-3 -záznam fronty pracovní zátěže klastru.

MQWQR_CURRENT_VERSION
Aktuální verze záznamu fronty pracovní zátěže klastru.

StrucLength (MQLONG) -vstupní
Délka struktury MQWQR . StrucLength má jednu z následujících hodnot:

MQWQR_LENGTH_1
Délka záznamu fronty pracovní zátěže klastru version-1 .

MQWQR_LENGTH_2
Délka záznamu fronty pracovní zátěže klastru version-2 .

MQWQR_LENGTH_3
Délka záznamu fronty pracovní zátěže klastru version-3 .

MQWQR_CURRENT_LENGTH
Délka aktuální verze záznamu fronty pracovní zátěže klastru.

QFlags (MQLONG) -vstupní
Příznaky fronty označují vlastnosti fronty. Jsou definovány následující příznaky:

MQQF_LOCAL_Q
Cíl je lokální fronta.

MQQF_CLWL_USEQ_ANY
Povolit použití lokálních a vzdálených front ve vložení.

MQQF_CLWL_USEQ_LOCAL
Povolit pouze vložení do lokální fronty.

Ostatní hodnoty
Další příznaky v poli mohou být nastaveny správcem front pro interní účely.

Název fronty (MQCHAR48) -vstupní
Název fronty, která je jedním z možných cílů zprávy.

- Délka QName je MQ_Q_NAME_LENGTH.

QMgrIdentifier (MQCHAR48) -vstupní
Identifikátor správce front je jedinečný identifikátor pro správce front, který je hostitelem instance fronty popsané ve struktuře MQWQR .

- Identifikátor je generován správcem front.
- Délka položky QMgrIdentifier je MQ_Q_MGR_IDENTIFIER_LENGTH.

ClusterRecClusterRec (MQLONG) -vstupní
Logické posunutí první struktury MQWCR , která patří do struktury MQWQR .

- V případě statických mezipamětí představuje ClusterRecOffset posun první struktury MQWCR , která patří do struktury MQWQR .
- Posun se měří v bajtech od začátku struktury MQWQR .
- Nepoužívejte logické posunutí pro aritmetiku ukazatele s dynamickými mezipaměťmi. Chcete-li získat adresu dalšího záznamu, musíte použít volání MQXCLWLN .

QType (MQLONG) -vstupní
Typ fronty cílové fronty. Možné jsou následující hodnoty:

MQCQT_LOCAL_Q
Lokální fronta.

MQCQT_ALIAS_Q
Fronta alias.

MQCQT_REMOTE_ Q
Vzdálená fronta.

MQCQT_Q_MGR_ALIAS
Alias správce front.

QDesc (MQCHAR64) -vstupní
Atribut fronty popisu fronty definovaný ve správci front, který je hostitelem instance cílové fronty popsané strukturou MQWQR .

- Délka parametru QDesc je MQ_Q_DESC_LENGTH.

DefBind (MQLONG) -vstupní
Výchozí atribut fronty vazby definovaný ve správci front, který je hostitelem instance cílové fronty popsané ve struktuře MQWQR . Při použití skupin s klastry musí být zadána hodnota MQBND_BIND_ON_OPEN nebo MQBND_BIND_ON_GROUP . Jsou možné následující hodnoty:

MQBND_BIND_ON_OPEN
Vazba byla opravena voláním funkce MQOPEN .

MQBND_BIND_NOT_FIXED
Vazba není opravena.

MQBND_BIND_ON_GROUP
Umožňuje aplikaci požadovat, aby byla skupina zpráv přidělena ke stejné cílové instanci.

DefPersistence (MQLONG) -vstupní
Výchozí atribut fronty perzistence zpráv definovaný ve správci front, který je hostitelem instance cílové fronty popsané strukturou MQWQR . Možné jsou následující hodnoty:

MQPER_PERSISTENT
Zpráva je trvalá.

MQPER_NOT_PERSISTENT
Zpráva není trvalá.

DefPriority (MQLONG) -vstupní

Výchozí atribut fronty priority zpráv definovaný ve správci front, který je hostitelem instance cílové fronty popsané ve struktuře MQWQR . Rozsah priorit je 0- MaxPriority.

- 0 je nejnižší priorita.
- MaxPriority je atribut správce front správce front, který je hostitelem této instance cílové fronty.

InhibitPut (MQLONG) -vstupní

Atribut fronty s blokovaným vkládáním definovaný ve správci front, který je hostitelem instance cílové fronty popsané strukturou MQWQR . Možné jsou následující hodnoty:

MQQA_PUT_INHIBITED
Operace vložení jsou zablokovány.

MQQA_PUT_ALLOWED
Operace vložení jsou povoleny.

CLWLQueuePriority (MQLONG) -vstupní

Atribut priority fronty pracovní zátěže klastru definovaný ve správci front, který je hostitelem instance cílové fronty popsané ve struktuře MQWQR .

CLWLQueueRank (MQLONG) -vstupní

Ohodnocení důležitosti fronty pracovní zátěže klastru definované ve správci front, který je hostitelem instance cílové fronty popsané ve struktuře MQWQR .

DefPutodezva (MQLONG) -vstupní

Výchozí atribut fronty odezvy vložení definovaný ve správci front, který je hostitelem instance cílové fronty popsané strukturou MQWQR . Možné jsou následující hodnoty:

MQPRT_SYNC_RESPONSE
Synchronní odezva na volání MQPUT nebo MQPUT1 .

MQPRT_ASYNC_RESPONSE
Asynchronní odezva na volání MQPUT nebo MQPUT1 .

Související odkazy

Počáteční hodnoty a deklarace jazyka pro MQWQR
Počáteční hodnoty a deklarace jazyka C a High Level Assembler pro MQWQR -záznam fronty pracovní zátěže klastru.

Počáteční hodnoty a deklarace jazyka pro MQWQR

Počáteční hodnoty a deklarace jazyka C a High Level Assembler pro MQWQR -záznam fronty pracovní zátěže klastru.

Tabulka 832. Pole v produktu MQWQR

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQWQR_STRUC_ID_ARRAY	'WQR¬'
Version	MQWQR_VERSION_1	1
StrucLength	MQWQR_CURRENT_LENGTH [3]	212
QFlags	Není	0
QName	Není	""
QMgrIdentifier	Není	""
ClusterRecOffset	Není	0
QType	Není	0
QDesc	Není	""
DefBind	Není	0

Tabulka 832. Pole v produktu MQWQR (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
DefPersistence	Není	0
DefPriority	Není	0
InhibitPut	Není	0
CLWLQueuePriority	Není	0
CLWLQueueRank	Není	0
DefPutResponse	Není	1

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyku C obsahuje proměnná makra MQWQR_DEFAULT výchozí hodnoty. Použijte jej následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

   ```
   MQWQR MyWQR = {MQWQR_DEFAULT};
   ```

3. Počáteční hodnoty záměrně nastavují délku struktury na délku aktuální verze, a nikoli na verzi 1 struktury.

C prohlášení

```
typedef struct tagMQWQR {
  MQCHAR4   StrucId;           /* Structure identifier */
  MQLONG    Version;           /* Structure version number */
  MQLONG    StrucLength;       /* Length of MQWQR structure */
  MQLONG    QFlags;            /* Queue flags */
  MQCHAR48  QName;             /* Queue name */
  MQCHAR48  QMgrIdentifier;    /* Queue manageridentifier */
  MQLONG    ClusterRecOffset;  /* Offset of first cluster record */
  MQLONG    QType;             /* Queue type */
  MQCHAR64  QDesc;             /* Queue description */
  MQLONG    DefBind;           /* Default binding */
  MQLONG    DefPersistence;    /* Default message persistence */
  MQLONG    DefPriority;       /* Default message priority */
  MQLONG    InhibitPut;        /* Whether put operations on the queue
                                  are allowed */
  /* version 2 */
  MQLONG    CLWLQueuePriority; /* Queue priority */
  MQLONG    CLWLQueueRank;     /* Queue rank */
  /* version 3 */
  MQLONG    DefPutResponse;    /* Default put response */
};
```

High Level Assembler

```
MQWQR                     DSECT
MQWQR_STRUCID             DS   CL4      Structure identifier
MQWQR_VERSION             DS   F        Structure version number
MQWQR_STRUCLENGTH         DS   F        Length of MQWQR structure
MQWQR_QFLAGS              DS   F        Queue flags
MQWQR_QNAME               DS   CL48     Queue name
MQWQR_QMGRIDENTIFIER      DS   CL48     Queue manager identifier
MQWQR_CLUSTERRECOFFSET    DS   F        Offset of first cluster
*                                       record
MQWQR_QTYPE               DS   F        Queue type
MQWQR_QDESC               DS   CL64     Queue description
MQWQR_DEFBIND             DS   F        Default binding
MQWQR_DEFPERSISTENCE      DS   F        Default message persistence
MQWQR_DEFPRIORITY         DS   F        Default message priority
MQWQR_INHIBITPUT          DS   F        Whether put operations on
```

```
*                                        the queue are allowed
MQWQR_DEFPUTRESPONSE           DS   F    Default put response
MQWQR_LENGTH                   EQU  *-MQWQR  Length of structure
                               ORG  MQWQR
MQWQR_AREA                     DS   CL(MQWQR_LENGTH)
```

Související odkazy

Pole v MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Popis polí ve struktuře záznamu fronty pracovní zátěže klastru MQWQR .

MQWCR -Struktura záznamu klastru pracovní zátěže klastru

Následující tabulka shrnuje pole ve struktuře záznamu pracovní zátěže klastru MQWCR .

Tabulka 833. Pole v produktu MQWCR

Pole	Popis	Stránka
ClusterName	Název klastru	ClusterName
ClusterRecOffset	Posun dalšího záznamu klastru (MQWCR)	ClusterRecClusterRec
ClusterFlags	Příznaky klastru	ClusterFlags

Struktura záznamu klastru pracovní zátěže klastru obsahuje informace o klastru. Pro každý klastr, do kterého patří cílová fronta, existuje jedna struktura záznamu klastru pracovní zátěže klastru.

Struktura záznamu klastru pracovní zátěže klastru je podporována ve všech prostředích.

Související odkazy

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání
Uživatelská procedura pracovní zátěže klastru je volána správcem front za účelem směrování zprávy do dostupného správce front.

MQXCLWLN -Procházet záznamy pracovní zátěže klastru
Volání MQXCLWLN se používá k navigaci v řetězcích záznamů MQWDR, MQWQRa MQWCR uložených v mezipaměti klastru.

MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole v MQWXP -Struktura parametrů uživatelské procedury pracovní zátěže klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu cíle pracovní zátěže klastru MQWDR .

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře záznamu fronty pracovní zátěže klastru MQWQR .

Pole v MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

Popis polí v MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

ClusterName (MQCHAR48) -vstupní
Název klastru, ke kterému náleží instance cílové fronty, která vlastní strukturu MQWCR . Instance cílové fronty je popsána strukturou MQWDR .

- Délka ClusterName je MQ_CLUSTER_NAME_LENGTH.

ClusterRecClusterRec (MQLONG) -vstupní
Logické posunutí další struktury MQWCR .

- Pokud již nejsou k dispozici žádné další struktury MQWCR , je hodnota ClusterRecOffset nula.
- Posun je měřen v bajtech od začátku struktury MQWCR .

ClusterFlags (MQLONG) -vstupní
Příznaky klastru označují vlastnosti správce front identifikovaného strukturou MQWCR . Jsou definovány následující příznaky:

MQQMF_REPOSITORY_Q_MGR
Cíl je správce front úplného úložiště.

MQQMF_CLUSSDR_USER_DEFINED
Odesílací kanál klastru byl definován ručně.

MQQMF_CLUSSDR_AUTO_DEFINED
Odesílací kanál klastru byl definován automaticky.

MQQMF_AVAILABLE
Správce cílové fronty je k dispozici pro příjem zpráv.

Ostatní hodnoty
Další příznaky v poli mohou být nastaveny správcem front pro interní účely.

Související odkazy

Počáteční hodnoty a deklarace jazyka pro MQWCR
Počáteční hodnoty a deklarace jazyka C a High Level Assembler pro MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

Počáteční hodnoty a deklarace jazyka pro MQWCR

Počáteční hodnoty a deklarace jazyka C a High Level Assembler pro MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

Tabulka 834. Pole v produktu MQWCR

Název pole	Název konstanty	Hodnota konstanty
ClusterName	Není	" "
ClusterRecOffset	Není	0
ClusterFlags	Není	0

C prohlášení

```
typedef struct tagMQWCR {
  MQCHAR48  ClusterName;       /* Cluster name */
  MQLONG    ClusterRecOffset;  /* Offset of next cluster record */
  MQLONG    ClusterFlags;      /* Cluster flags */
};
```

High Level Assembler

```
MQWCR                          DSECT
MQWCR_CLUSTERNAME              DS   CL48     Cluster name
MQWCR_CLUSTERRECOFFSET         DS   F        Offset of next cluster
*                                           record
MQWCR_CLUSTERFLAGS             DS   F        Cluster flags
MQWCR_LENGTH                   EQU  *-MQWCR  Length of structure
                               ORG  MQWCR
MQWCR_AREA                     DS   CL(MQWCR_LENGTH)
```

Související odkazy

Pole v MQWCR -Struktura záznamu klastru pracovní zátěže klastru.
Popis polí v MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

Odkaz uživatelské procedury rozhraní API

Tento oddíl poskytuje referenční informace, které jsou zajímavé především pro programátora, který píše uživatelské procedury rozhraní API.

Obecné poznámky k použití

poznámky:

1. Všechny funkce uživatelské procedury mohou vyvolat volání MQXEP. Toto volání je navrženo speciálně pro použití z funkcí uživatelské procedury rozhraní API.
2. Funkce MQ_INIT_EXIT nemůže vydat žádná jiná volání produktu MQ než MQXEP.
3. Pro aktuální připojení nelze zadat volání MQDISC.
4. Pokud funkce exit zadá volání MQCONN nebo volání MQCONNX s volbou MQCNO_HANDLE_SHARE_NONE, volání se dokončí s kódem příčiny MQRC_ALREADY_CONNECTED a vrácený manipulátor je stejný jako ten, který byl předán uživatelské proceduře jako parametr.
5. Obecně platí, že pokud funkce uživatelské procedury rozhraní API vydá volání MQI, uživatelské procedury rozhraní API nebudou volány rekurzivně. Pokud však funkce exit zadá volání MQCONNX s volbami MQCNO_HANDLE_SHARE_BLOCK nebo MQCNO_HANDLE_SHARE_NO_BLOCK, volání vrátí nový sdílený manipulátor. To poskytuje výstupní sadu s vlastní manipulací připojení, a tedy s pracovní jednotkou, která je nezávislá na pracovní jednotce aplikace. Sada ukončení může tento manipulátor použít k vložení a získání zpráv v rámci své vlastní jednotky práce a k potvrzení nebo vrácení této jednotky práce; to vše lze provést bez jakéhokoli ovlivnění jednotky práce aplikace.

   Vzhledem k tomu, že funkce uživatelské procedury používá manipulátor připojení, který se liší od manipulátoru používaného aplikací, volání MQ vydaná funkcí uživatelské procedury vedou k vyvolání příslušných funkcí uživatelské procedury rozhraní API. Funkce ukončení lze proto vyvolat rekurzivně. Všimněte si, že pole *ExitUserArea* v MQAXP i oblast řetězce ukončení mají rozsah manipulátoru připojení. Funkce ukončení proto nemůže tyto oblasti použít k tomu, aby signalizovala jiné instanci sama o sobě vyvolané rekurzivně, že je již aktivní.
6. Funkce ukončení mohou také vkládat a získávat zprávy v rámci jednotky práce aplikace. Když aplikace potvrdí nebo odvolá jednotku práce, všechny zprávy v rámci jednotky práce jsou potvrzeny nebo vráceny dohromady, bez ohledu na to, kdo je umístil do jednotky práce (funkce aplikace nebo ukončení). Ukončení však může způsobit, že aplikace překročí limity systému dříve, než by tomu bylo jinak (například překročením maximálního počtu nepotvrzených zpráv v transakci).

   Pokud funkce ukončení používá pracovní jednotku aplikace tímto způsobem, funkce ukončení by se měla obvykle vyhnout volání MQCMIT, protože tím se potvrdí pracovní jednotka aplikace a může dojít k narušení správné funkce aplikace. Funkce exit však může někdy vyžadovat zadání volání MQBACK, pokud funkce exit narazí na závažnou chybu, která brání potvrzení transakce (například chyba při vkládání zprávy jako součásti jednotky práce aplikace). Při volání MQBACK dbejte na to, aby se nezměnily hranice pracovní jednotky aplikace. V této situaci musí funkce ukončení nastavit odpovídající hodnoty, aby se zajistilo, že se aplikaci vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_BACKED_OUT, aby mohla aplikace zjistit, že jednotka práce byla vrácena zpět.

   Pokud funkce uživatelské procedury používá manipulátor připojení aplikace k zadání volání produktu MQ , tato volání sama o sobě nevedou k dalším vyvoláním funkcí uživatelské procedury rozhraní API.
7. Pokud dojde k nestandardnímu ukončení funkce MQXR_BEFORE, může být správce front schopen provést zotavení po selhání. Pokud je to možné, pokračuje správce front ve zpracování, jako by funkce uživatelské procedury vrátila hodnotu MQXCC_FAILED. Pokud se správce front nemůže zotavit, je aplikace ukončena.
8. Pokud dojde k nestandardnímu ukončení funkce uživatelské procedury MQXR_AFTER, může být správce front schopen provést zotavení po selhání. Pokud je to možné, pokračuje správce front ve zpracování, jako by funkce uživatelské procedury vrátila hodnotu MQXCC_FAILED. Pokud se správce front nemůže zotavit, je aplikace ukončena. Uvědomte si, že ve druhém případě jsou zprávy načtené mimo jednotku práce ztraceny (jedná se o stejnou situaci jako aplikace, která selhala bezprostředně po odebrání zprávy z fronty).
9. Proces MCA provádí dvoufázové potvrzování.

   Pokud uživatelská procedura rozhraní API zachytí MQCMIT z připraveného procesu MCA a pokusí se provést akci v rámci pracovní jednotky, akce se nezdaří s kódem příčiny MQRC_UOW_NOT_AVAILABLE.

10. Je-li k dispozici více instalací produktu IBM MQ , použijte uživatelské procedury napsané pro starší verzi produktu IBM MQ, protože nové funkce přidané v novější verzi nemusí pracovat se staršími verzemi. Další informace o změnách mezi verzemi viz Co se změnilo v produktu IBM MQ 8.0.

Struktura parametrů uživatelské procedury rozhraní API IBM MQ (MQAXP)

Struktura MQAXP, externí řídicí blok, se používá jako vstupní nebo výstupní parametr uživatelské procedury rozhraní API. V tomto tématu jsou také uvedeny informace o tom, jak správci front zpracovávají funkce uživatelské procedury.

MQAXP má následující deklaraci C:

```
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 */
};
```

Při vyvolání funkcí v uživatelské proceduře rozhraní API je předán následující seznam parametrů:

StrucId (MQCHAR4)-vstup
Identifikátor struktury parametru ukončení s hodnotou:

```
MQAXP_STRUC_ID.
```

Obslužná rutina ukončení nastaví toto pole při vstupu do každé funkce ukončení.

Verze (MQLONG)-vstup
Číslo verze struktury s hodnotou:

MQAXP_VERSION_1
Struktura parametrů uživatelské procedury rozhraní API verze 1.

MQAXP_VERSION_2
Struktura parametrů uživatelské procedury rozhraní API verze 2.

MQAXP_CURRENT_VERSION
Aktuální číslo verze pro strukturu parametrů uživatelské procedury rozhraní API.

Obslužná rutina ukončení nastaví toto pole při vstupu do každé funkce ukončení.

ExitId (MQLONG)-vstup
Identifikátor uživatelské procedury nastavený při vstupu do uživatelské procedury s uvedením typu uživatelské procedury:

MQXT_API_EXIT
Uživatelská procedura rozhraní API.

ExitReason (MQLONG)-vstup.
Příčina vyvolání uživatelské procedury, nastavit při vstupu na každou funkci uživatelské procedury:

MQXR_CONNECTION
Probíhá vyvolání uživatelské procedury, aby se inicializovala před voláním MQCONN nebo MQCONNX nebo aby se ukončila po volání MQDISC.

MQXR_BEFORE (předchozí)
Uživatelská procedura je vyvolávána před provedením volání rozhraní API nebo před převodem dat na operaci MQGET.

MQXR_AFTER
Uživatelská procedura je vyvolávána po provedení volání rozhraní API.

ExitResponse (MQLONG)-výstup
Odezva z uživatelské procedury, inicializovaná při vstupu do každé funkce uživatelské procedury na:

MQXCC_OK
Pokračujte normálně.

Toto pole musí být nastaveno funkcí exit, aby bylo možné správci front sdělit výsledek provedení funkce exit. Hodnota musí být jedna z následujících:

MQXCC_OK
Funkce ukončení byla úspěšně dokončena. Pokračujte normálně.

Tuto hodnotu lze nastavit všemi funkcemi uživatelské procedury MQXR_ *. Volba ExitResponse2 se používá k rozhodnutí, zda vyvolat funkce ukončení později v řetězci.

MQXCC_FAILED
Funkce ukončení selhala kvůli chybě.

Tuto hodnotu lze nastavit všemi funkcemi uživatelské procedury MQXR_ *. Správce front nastaví parametr CompCode na hodnotu MQCC_FAILED a důvod na hodnotu:

- MQRC_API_EXIT_INIT_ERROR, pokud je funkce MQ_INIT_EXIT
- MQRC_API_EXIT_TERM_ERROR, pokud je funkce MQ_TERM_EXIT
- MQRC_API_EXIT_ERROR pro všechny ostatní funkce ukončení

Nastavené hodnoty mohou být později v řetězci změněny funkcí exit.

Volba ExitResponse2 je ignorována; správce front pokračuje ve zpracování, jako by byla vrácena hodnota MQXR2_SUPPRESS_CHAIN .

MQXCC_SUPPRESS_FUNCTION
Potlačit funkci rozhraní API IBM MQ .

Tuto hodnotu může nastavit pouze funkce MQXR_BEFORE exit. Vynechá volání rozhraní API. Je-li vrácena příkazem MQ_DATA_CONV_ON_GET_EXIT, převod dat se vynechá. Správce front nastaví parametr CompCode na hodnotu MQCC_FAILED a parametr Reason to MQRC_SUPPRESSED_BY_EXIT, nastavené hodnoty však mohou být později v řetězci změněny funkcí uživatelské procedury. Ostatní parametry pro volání zůstávají tak, jak je ukončil. Volba ExitResponse2 se používá k rozhodnutí, zda vyvolat funkce ukončení později v řetězci.

Je-li tato hodnota nastavena funkcí uživatelské procedury MQXR_AFTER nebo MQXR_CONNECTION, pokračuje správce front ve zpracování, jako by bylo vráceno MQXCC_FAILED.

Funkce MQXCC_SKIP_FUNCTION
Přeskočte funkci rozhraní API IBM MQ .

Tuto hodnotu může nastavit pouze funkce MQXR_BEFORE exit. Vynechá volání rozhraní API. Je-li vrácena příkazem MQ_DATA_CONV_ON_GET_EXIT, převod dat se vynechá. Funkce exit musí nastavit CompCode a Reason na hodnoty, které mají být vráceny do aplikace, ale nastavené hodnoty mohou být změněny funkcí exit později v řetězci. Ostatní parametry pro volání zůstávají tak, jak je ukončil. Volba ExitResponse2 se používá k rozhodnutí, zda vyvolat funkce ukončení později v řetězci.

Je-li tato hodnota nastavena funkcí uživatelské procedury MQXR_AFTER nebo MQXR_CONNECTION, pokračuje správce front ve zpracování, jako by bylo vráceno MQXCC_FAILED.

MQXCC_SUPPRESS_EXIT
Potlačit všechny funkce ukončení patřící do sady uživatelských procedur.

Tuto hodnotu lze nastavit pouze pomocí funkcí ukončení MQXR_BEFORE a MQXR_AFTER. Vynechá *všechna* následná vyvolání funkcí ukončení náležících k této sadě uživatelských procedur pro toto logické připojení. Toto vynechání pokračuje, dokud se nevyskytne požadavek na logické odpojení, když je funkce MQ_TERM_EXIT vyvolána s hodnotou ExitReason MQXR_CONNECTION.

Funkce exit musí nastavit CompCode a Reason na hodnoty, které mají být vráceny do aplikace, ale nastavené hodnoty mohou být změněny funkcí exit později v řetězci. Ostatní parametry pro volání zůstávají tak, jak je ukončil. Volba ExitResponse2 je ignorována.

Je-li tato hodnota nastavena funkcí uživatelské procedury MQXR_CONNECTION, pokračuje správce front ve zpracování, jako by bylo vráceno MQXCC_FAILED.

Informace o interakci mezi ExitResponse a ExitResponse2a jejím vlivu na zpracování ukončení viz "Jak správci front zpracovávají funkce uživatelské procedury" na stránce 1559.

ExitResponse2 (MQLONG)-výstup
Jedná se o sekundární kód odezvy uživatelské procedury, který kvalifikuje primární kód odezvy uživatelské procedury pro funkce uživatelské procedury MQXR_BEFORE. Je inicializován na:

```
MQXR2_DEFAULT_CONTINUATION
```

při vstupu do funkce uživatelské procedury volání rozhraní API systému IBM MQ . Pak může být nastavena na jednu z hodnot:

MQXR2_DEFAULT_CONTINUATION
Zda pokračovat s další uživatelskou procedurou v řetězci, v závislosti na hodnotě ExitResponse.

Má-li parametr ExitResponse hodnotu MQXCC_SUPPRESS_FUNCTION nebo MQXCC_SKIP_FUNCTION, vynechte funkce ukončení později v řetězci MQXR_BEFORE a odpovídající funkce ukončení v řetězci MQXR_AFTER. Vyvolejte funkce uživatelské procedury v řetězci MQXR_AFTER, které odpovídají funkcím uživatelské procedury dříve v řetězci MQXR_BEFORE.

Jinak vyvolejte další uživatelskou proceduru v řetězci.

MQXR2_SUPPRESS_CHAIN
Potlačte řetěz.

Vynechte funkce ukončení později v řetězci MQXR_BEFORE a odpovídající funkce ukončení v řetězci MQXR_AFTER pro toto vyvolání volání rozhraní API. Vyvolejte funkce uživatelské procedury v řetězci MQXR_AFTER, které odpovídají funkcím uživatelské procedury dříve v řetězci MQXR_BEFORE.

MQXR2_CONTINUE_CHAIN
Pokračujte dalším výjezdem v řetězci.

Informace o interakci mezi ExitResponse a ExitResponse2a jejím vlivu na zpracování ukončení viz "Jak správci front zpracovávají funkce uživatelské procedury" na stránce 1559.

Zpětná vazba (MQLONG)-vstup/výstup
Sdělujte kódy zpětné vazby mezi vyvoláními funkce ukončení. Toto je inicializováno na:

```
MQFB_NONE (0)
```

před vyvoláním první funkce prvního ukončení v řetězci.

Uživatelské procedury mohou toto pole nastavit na libovolnou hodnotu, včetně platné hodnoty MQFB_ * nebo MQRC_ *. Uživatelské procedury mohou také nastavit toto pole na uživatelem definovanou hodnotu zpětné vazby v rozsahu MQFB_APPL_FIRST až MQFB_APPL_LAST.

APICallerType (MQLONG)-vstup
Typ volajícího rozhraní API označující, zda je volající rozhraní API produktu IBM MQ externí nebo interní pro správce front: MQXACT_EXTERNAL nebo MQXACT_INTERNAL.

ExitUserArea (MQBYTE16)-vstupní/výstupní
Uživatelská oblast, která je k dispozici pro všechny uživatelské procedury přidružené ke konkrétnímu objektu ExitInfo. Je inicializován na MQXUA_NONE (binární nuly pro délku oblasti ExitUser) před vyvoláním první funkce uživatelské procedury (MQ_INIT_EXIT) pro hconn. Od té doby jsou veškeré změny provedené v tomto poli funkcí exit zachovány ve všech vyvoláních funkcí stejné uživatelské procedury.

Toto pole je zarovnáno s násobkem 4 MQLONGs.

Uživatelské procedury mohou také ukotvit libovolné úložiště, které alokují z této oblasti.

Pro každé připojení hconn má každá uživatelská procedura v řetězci uživatelských procedur jinou oblast ExitUser. Oblast ExitUsernelze sdílet s uživatelskými procedurami v řetězu a obsah oblasti ExitUserpro jednu uživatelskou proceduru není k dispozici pro jinou uživatelskou proceduru v řetězu.

Pro programy v jazyce C je konstanta MQXUA_NONE_ARRAY také definována se stejnou hodnotou jako MQXUA_NONE, ale jako pole znaků namísto řetězce.

Délka tohoto pole je dána hodnotou MQ_EXIT_USER_AREA_LENGTH.

ExitData (MQCHAR32)-vstup
Ukončete data, nastavte na vstupu pro každou funkci ukončení na 32 znaků dat specifických pro ukončení, která jsou poskytnuta v uživatelské proceduře. Pokud v uživatelské proceduře nedefinujete žádnou hodnotu, bude toto pole prázdné.

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

ExitInfoNázev (MQCHAR48)-vstup
Název informací o ukončení, nastavený na vstupu pro každou funkci ukončení na ApiExit_name uvedenou v definicích ukončení v sekcích.

ExitPDArea (MQBYTE48)-vstupní/výstupní
Oblast pro určování problémů, inicializovaná na MQXPDA_NONE (binární nuly pro délku pole) pro každé vyvolání funkce ukončení.

Pro programy v jazyce C je konstanta MQXPDA_NONE_ARRAY také definována se stejnou hodnotou jako MQXPDA_NONE, ale jako pole znaků místo řetězce.

Obslužná rutina ukončení vždy zapisuje tuto oblast do trasování IBM MQ na konci uživatelské procedury, i když je funkce úspěšná.

Délka tohoto pole je dána hodnotou MQ_EXIT_PD_AREA_LENGTH.

QMgrName (MQCHAR48)-vstup
Název správce front, ke kterému je aplikace připojena, který vyvolal uživatelskou proceduru v důsledku zpracování volání rozhraní API IBM MQ .

Pokud je název správce front zadaný ve voláních MQCONN nebo MQCONNX prázdný, je toto pole stále nastaveno na název správce front, ke kterému je aplikace připojena, bez ohledu na to, zda se jedná o server nebo klienta.

Obslužná rutina ukončení nastaví toto pole při vstupu do každé funkce ukončení.

Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

ExitChainAreaPtr (PMQACH)-vstupní/výstupní
Používá se ke komunikaci dat mezi vyvoláními různých uživatelských procedur v řetězci. Je nastaven na ukazatel NULL před vyvoláním první funkce (MQ_INIT_EXIT s ExitReason MQXR_CONNECTION) první uživatelské procedury v řetězci uživatelských procedur. Hodnota vrácená uživatelskou procedurou při jednom vyvolání je předána dalšímu vyvolání.

Další podrobnosti o použití oblasti řetězce ukončení viz “Záhlaví oblasti řetězce ukončení a oblasti řetězce ukončení (MQACH)” na stránce 1563 .

Hconfig (MQHCONFIG)-vstup
Popisovač konfigurace představující sadu inicializovaných funkcí. Tato hodnota je generována správcem front ve funkci MQ_INIT_EXIT a později předána funkci uživatelské procedury rozhraní API. Je nastaven při vstupu do každé funkce ukončení.

Příkaz Hconfig můžete použít jako ukazatel na strukturu MQIEP k provedení volání MQI a DCI. Před použitím parametru HConfig jako ukazatele na strukturu MQIEP musíte zkontrolovat, zda první 4 bajty rozhraní HConfig odpovídají identifikátoru StrucId struktury MQIEP.

Funkce (MQLONG)-vstup
Identifikátor funkce, platné hodnoty, pro které jsou konstanty MQXF_ * popsané v části "Externí konstanty" na stránce 1564.

Obslužná rutina uživatelské procedury nastaví toto pole na správnou hodnotu při vstupu do každé funkce uživatelské procedury v závislosti na volání rozhraní API IBM MQ , které vedlo k vyvolání uživatelské procedury.

ExitMsgManipulátor (MQHMSG)-vstup/výstup
Je-li funkce MQXF_GET a ExitReason má hodnotu MQXR_AFTER, vrátí se v tomto poli platný popisovač zprávy, který umožňuje uživatelské proceduře rozhraní API přístup k polím deskriptoru zpráv a k dalším vlastnostem, které odpovídají řetězci ExitProperties zadanému ve struktuře MQXEPO při registraci uživatelské procedury rozhraní API.

Všechny vlastnosti deskriptoru bez zpráv, které jsou vráceny v popisovači ExitMsg, nebudou k dispozici v popisovači MsgHandle ve struktuře MQGMO, pokud byl zadán, nebo v datech zprávy.

Má-li funkce MQXF_GET a ExitReason hodnotu MQXR_BEFORE, pak pokud uživatelský program nastaví toto pole na hodnotu MQHM_NONE, potlačí naplnění vlastností popisovače ExitMsg.

Toto pole není nastaveno, pokud je verze menší než MQAXP_VERSION_2.

Jak správci front zpracovávají funkce uživatelské procedury

Zpracování provedené správcem front při návratu z funkce ukončení závisí na ExitResponse i ExitResponse2.

Tabulka 835 na stránce 1560 shrnuje možné kombinace a jejich účinky pro funkci ukončení MQXR_BEFORE a zobrazuje:

- Kdo nastaví parametry CompCode a Důvod volání rozhraní API
- Zda jsou vyvolány zbývající funkce uživatelské procedury v řetězci MQXR_BEFORE a odpovídající funkce uživatelské procedury v řetězci MQXR_AFTER.
- Zda je vyvoláno volání rozhraní API

Pro funkci ukončení MQXR_AFTER:

- CompCode a příčina jsou nastaveny stejným způsobem jako MQXR_BEFORE.
- Parametr ExitResponse2 je ignorován (zbývající funkce ukončení v řetězci MQXR_AFTER jsou vždy vyvolány).
- MQXCC_SUPPRESS_FUNCTION a MQXCC_SKIP_FUNCTION nejsou platné

Pro funkci ukončení MQXR_CONNECTION:

- CompCode a příčina jsou nastaveny stejným způsobem jako MQXR_BEFORE.
- Volba ExitResponse2 je ignorována.
- MQXCC_SUPPRESS_FUNCTION, MQXCC_SKIP_FUNCTION, MQXCC_SUPPRESS_EXIT nejsou platné

Ve všech případech, kdy uživatelská procedura nebo správce front nastaví volbu CompCode a příčina, lze nastavení hodnot změnit pomocí uživatelské procedury vyvolané později nebo voláním rozhraní API (pokud je volání rozhraní API vyvoláno později).

Tabulka 835. MQXR_BEFORE zpracování ukončení			
Hodnota ExitResponse	**CompCode a příčina nastavena pomocí**	**Hodnota řetězce ExitResponse2 (výchozí pokračování)**	**Hodnota rozhraní API ExitResponse2 (výchozí pokračování)**
MQXCC_OK	exit	Y	Y
MQXCC_SUPPRESS_EXIT	exit	Y	Y
MQXCC_SUPPRESS_FUNCTION	správce front	N	N
FUNKCE MQXCC_SKIP	exit	N	N
MQXCC_FAILED	správce front	N	N

Jak klienti zpracovávají funkce ukončení

Obecně platí, že klienti zpracovávají funkce ukončení stejným způsobem jako serverové aplikace a atribut *QMgrName* v této struktuře platí bez ohledu na to, zda je funkce na serveru nebo klientovi.

Klient však nemá žádný koncept souboru *mqs.ini* , takže sekce *ApiExitCommon* a *APIExitTemplate* neplatí. Použije se pouze sekce *ApiExitLokální* a tato sekce je nakonfigurována v souboru *mqclient.ini* .

Struktura kontextu uživatelské procedury rozhraní API IBM MQ (MQAXC)

Struktura MQAXC, externí řídicí blok, se používá jako vstupní parametr pro uživatelskou proceduru rozhraní API.

MQAXC má následující deklaraci C:

```
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 */
 };
```

Parametry pro MQAXC jsou:

StrucId (MQCHAR4)-vstup
Identifikátor struktury kontextu ukončení s hodnotou `MQAXC_STRUC_ID`. Pro programy v jazyce C je definována také konstanta MQAXC_STRUC_ID_ARRAY se stejnou hodnotou jako MQAXC_STRUC_ID, ale jako pole znaků namísto řetězce.

Obslužná rutina ukončení nastaví toto pole při vstupu do každé funkce ukončení.

Verze (MQLONG)-vstup
Číslo verze struktury s hodnotou:

MQAXC_VERSION_2
Číslo verze pro strukturu kontextu ukončení.

MQAXC_CURRENT_VERSION
Aktuální číslo verze pro strukturu kontextu ukončení.

Obslužná rutina ukončení nastaví toto pole při vstupu do každé funkce ukončení.

Prostředí (MQLONG)-vstup
Prostředí, ze kterého bylo vydáno volání rozhraní API IBM MQ , které mělo za následek řízenou funkci ukončení. Platné hodnoty pro toto pole jsou:

MQXE_OTHER
Tato hodnota je konzistentní s vyvoláními, která uživatelská procedura rozhraní API vidí, pokud je uživatelská procedura volána ze serverové aplikace. To znamená, že uživatelská procedura rozhraní API je na klientovi spuštěna beze změny a nevidí nic jiného.

Pokud uživatelská procedura skutečně potřebuje určit, zda je spuštěna na klientovi, může tak učinit pomocí polí *ChannelName* a *ChannelDefinition* .

MQXE_MCA
Agent oznamovacího kanálu

MQXE_MCA_SVRCONN
Agent kanálu zpráv jednající jménem klienta.

MQXE_COMMAND_SERVER
Příkazový server

MQXE_MQSC
Interpret příkazů runmqsc

Obslužná rutina ukončení nastaví toto pole při vstupu do každé funkce ukončení.

UserId (MQCHAR12)-vstup
ID uživatele přidružené k aplikaci. Zejména v případě připojení klienta toto pole obsahuje ID uživatele adoptovaného uživatele na rozdíl od ID uživatele, pod kterým je spuštěn kód kanálu. Pokud prázdné ID uživatele proudí z klienta, neprovede se žádná změna ID uživatele, které se již používá. To znamená, že není přijato žádné nové ID uživatele.

Obslužná rutina ukončení nastaví toto pole při vstupu do každé funkce ukončení. Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH.

V případě klienta se jedná o ID uživatele odeslané z klienta na server. Všimněte si, že se nemusí jednat o efektivní ID uživatele, pro které je klient spuštěn ve správci front, protože může existovat konfigurace MCAUser nebo CHLAUTH, která změní ID uživatele.

SecurityId (MQBYTE40)-vstup
Rozšíření ID uživatele, který spouští aplikaci. Jeho délka je dána hodnotou MQ_SECURITY_ID_LENGTH.

V případě klienta se jedná o ID uživatele odeslané z klienta na server. Všimněte si, že se nemusí jednat o efektivní ID uživatele, pro které je klient spuštěn ve správci front, protože může existovat konfigurace MCAUser nebo CHLAUTH, která změní ID uživatele.

ConnectionName (MQCHAR264)-vstup
Pole s názvem připojení je nastaveno na adresu klienta. Například pro TCP/IP by to byla adresa IP klienta.

Délka tohoto pole je dána hodnotou MQ_CONN_NAME_LENGTH.

V případě klienta se jedná o adresu partnera správce front.

LongMCAUserIdLength (MQLONG)-vstup
Délka dlouhého identifikátoru uživatele MCA.

Při připojení MCA ke správci front je toto pole nastaveno na délku dlouhého identifikátoru uživatele MCA (nebo na nulu, pokud takový identifikátor neexistuje).

V případě klienta se jedná o dlouhý identifikátor uživatele klienta.

LongRemoteUserIdDélka (MQLONG)-vstup
Délka dlouhého identifikátoru vzdáleného uživatele.

Při připojení MCA ke správci front je toto pole nastaveno na délku dlouhého identifikátoru vzdáleného uživatele. Jinak bude toto pole nastaveno na nulu

V případě klienta nastavte toto pole na nulu.

LongMCAUserIdPtr (MQPTR)-vstup
Adresa dlouhého identifikátoru uživatele MCA.

Při připojení MCA ke správci front je toto pole nastaveno na adresu dlouhého identifikátoru uživatele MCA (nebo na nulový ukazatel, pokud takový identifikátor neexistuje).

V případě klienta se jedná o dlouhý identifikátor uživatele klienta.

LongRemoteUserIdPtr (MQPTR)-vstup
Adresa dlouhého identifikátoru vzdáleného uživatele.

Při připojení MCA ke správci front je toto pole nastaveno na adresu dlouhého identifikátoru vzdáleného uživatele (nebo na nulový ukazatel, pokud takový identifikátor neexistuje).

V případě klienta nastavte toto pole na nulu.

ApplName (MQCHAR28)-vstup
Název aplikace nebo komponenty, která vydala volání rozhraní API IBM MQ .

Pravidla pro generování ApplName jsou stejná jako pro generování výchozího názvu pro MQPUT.

Hodnota tohoto pole je nalezena dotazováním operačního systému na název programu. Jeho délka je dána hodnotou MQ_APPL_NAME_LENGTH.

ApplType (MQLONG)-vstup
Typ aplikace nebo komponenty, která vydala volání rozhraní API IBM MQ .

Hodnota je MQAT_DEFAULT pro platformu, na které je aplikace kompilována, nebo se rovná jedné z definovaných hodnot MQAT_ *.

Obslužná rutina ukončení nastaví toto pole při vstupu do každé funkce ukončení.

ProcessId (MQPID)-vstup
Identifikátor procesu operačního systému.

Je-li to vhodné, nastaví obslužná rutina ukončení toto pole při vstupu do každé funkce ukončení.

ThreadId (MQTID)-vstup
Identifikátor podprocesu MQ . Jedná se o stejný identifikátor, který se používá v trasování produktu MQ a ve výpisech paměti produktu FFST , ale může se lišit od identifikátoru podprocesu operačního systému.

Je-li to vhodné, nastaví obslužná rutina ukončení toto pole při vstupu do každé funkce ukončení.

ChannelName (MQCHAR)-vstup
Název kanálu, doplněný mezerami, je-li použitelný a známý.

Není-li použitelné, je toto pole nastaveno na znaky NULL.

Reserved1 (MQBYTE4)-vstup
Toto pole je vyhrazeno.

ChanneDefinition (PMQCD)-vstup
Ukazatel na používanou definici kanálu, je-li použitelná a známá.

Není-li použitelné, je toto pole nastaveno na znaky NULL.

Mějte na paměti, že ukazatel je dokončen pouze v případě, že je připojení zpracováno jménem kanálu IBM MQ a že byla přečtena definice kanálu.

Při prvním volání MQCONN pro daný kanál není na serveru určena definice kanálu. Kromě toho, pokud je ukazatel vyplněn, musí být struktura (a jakékoli dílčí struktury), na kterou ukazatel odkazuje,

považována za jen pro čtení; jakákoli aktualizace struktury by vedla k nepředvídatelným výsledkům a není podporována.

V případě klienta obsahují pole jiná než pole s hodnotou určenou pro klienta hodnoty, které jsou vhodné pro klientskou aplikaci.

Záhlaví oblasti řetězce ukončení a oblasti řetězce ukončení (MQACH)

V případě potřeby může funkce exit získat úložiště pro oblast řetězce exit a nastavit ExitChainAreaPtr v MQAXP tak, aby ukazovalo na toto úložiště.

Uživatelské procedury (buď stejné, nebo různé uživatelské funkce) mohou získat více oblastí řetězce uživatelských procedur a vzájemně je propojit. Oblasti řetězce ukončení musí být přidány nebo odebrány z tohoto seznamu pouze při volání z obslužné rutiny ukončení. To zajišťuje, že neexistují žádné problémy serializace způsobené tím, že různé podprocesy přidávají nebo odebírají oblasti ze seznamu současně.

Oblast řetězce ukončení musí začínat strukturou záhlaví MQACH, jejíž deklarace C je:

```
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 */
 };
```

Pole v záhlaví oblasti řetězce ukončení jsou:

StrucId (MQCHAR4)-vstup
Identifikátor struktury oblasti řetězce uživatelských procedur s počáteční hodnotou, definovanou hodnotou MQACH_DEFAULT, MQACH_STRUC_ID.

Pro programy v jazyce C je definována také konstanta MQACH_STRUC_ID_ARRAY, která má stejnou hodnotu jako MQACH_STRUC_ID, ale jako pole znaků namísto řetězce.

Verze (MQLONG)-vstup
Číslo verze struktury:

MQACH_VERSION_1
Číslo verze pro strukturu parametru ukončení.

MQACH_CURRENT_VERSION
Aktuální číslo verze pro strukturu kontextu ukončení.

Počáteční hodnota tohoto pole definovaná parametrem MQACH_DEFAULT je MQACH_CURRENT_VERSION.

Poznámka: Pokud zavedete novou verzi této struktury, rozvržení existující části se nezmění. Funkce ukončení musí zkontrolovat, zda je číslo verze stejné nebo větší než nejnižší verze obsahující pole, která funkce ukončení potřebuje použít.

StrucLength (MQLONG)-vstup
Délka struktury MQACH. Uživatelské procedury mohou pomocí tohoto pole určit začátek dat uživatelské procedury a nastavit jej na délku struktury vytvořené uživatelskou procedurou.

Počáteční hodnota tohoto pole definovaná parametrem MQACH_DEFAULT je MQACH_CURRENT_LENGTH.

ChainAreaDélka (MQLONG)-vstup
Délka oblasti řetězce ukončení, nastavená na celkovou délku aktuální oblasti řetězce ukončení, včetně záhlaví MQACH.

Počáteční hodnota tohoto pole definovaná parametrem MQACH_DEFAULT je nula.

ExitInfoNázev (MQCHAR48)-vstup
Název informací o ukončení.

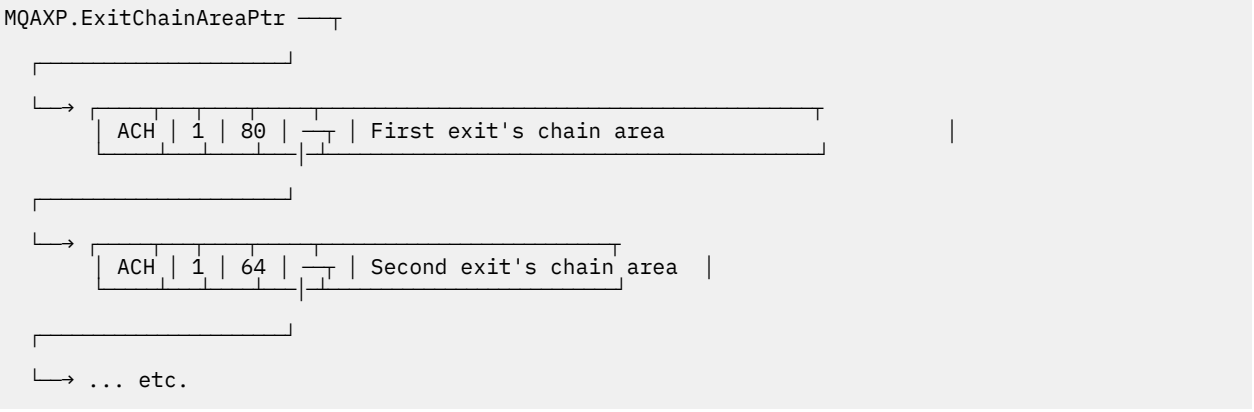
Když uživatelská procedura vytvoří strukturu MQACH, musí inicializovat toto pole s vlastním názvem ExitInfo, aby později mohla být tato struktura MQACH nalezena buď jinou instancí této uživatelské procedury, nebo spolupracující uživatelskou procedurou.

Počáteční hodnota tohoto pole definovaná parametrem MQACH_DEFAULT je řetězec s nulovou délkou ({""}).

NextChainAreaPtr (PMQACH)-vstup
Ukazatel na další oblast řetězce uživatelské procedury s počáteční hodnotou, definovanou hodnotou MQACH_DEFAULT, ukazatele Null (NULL).

Funkce ukončení musí uvolnit úložiště pro všechny oblasti řetězce ukončení, které získají, a manipulovat s ukazateli řetězce, aby odebraly své oblasti řetězce ukončení ze seznamu.

Oblast výstupního řetězce může být konstruována takto:



Externí konstanty

Toto téma slouží jako referenční informace pro externí konstanty, které jsou k dispozici pro rozhraní API.

Pro uživatelské procedury rozhraní API jsou k dispozici následující externí konstanty:

MQXF_ * (identifikátory funkce ukončení)

```
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_ * (příčiny ukončení)

```
MQXR_BEFORE                     1       X'00000001'
MQXR_AFTER                      2       X'00000002'
MQXR_CONNECTION                 3       X'00000003'
```

MQXE_ * (prostředí)

```
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* _ * (další konstanty)

```
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* _ * (konstanty null)

```
MQXPDA_NONE                     X'00...00' (48 nulls)
MQXPDA_NONE_ARRAY               '\0','\0',...,'\0','\0'
```

MQXCC_ * (kódy dokončení)

```
MQXCC_FAILED                   -8
```

MQRC_ * (kódy příčiny)

MQRC_API_EXIT_ERROR 2374 X'00000946'
Vyvolání funkce uživatelské procedury vrátilo neplatný kód odezvy nebo se nějakým způsobem nezdařilo a správce front nemůže určit další akci, která má být provedena.

Zkontrolujte obě pole ExitResponse a ExitResponse2 v systému MQAXP, abyste určili chybný kód odezvy, a změňte uživatelskou proceduru tak, aby vracela platný kód odezvy.

MQRC_API_EXIT_INIT_ERROR 2375 X'00000947'
Správce front zjistil chybu při inicializaci prováděcího prostředí pro funkci uživatelské procedury rozhraní API.

MQRC_API_EXIT_TERM_ERROR 2376 X'00000948'
Správce front zjistil chybu při zavírání prováděcího prostředí pro funkci uživatelské procedury rozhraní API.

MQRC_EXIT_REASON_ERROR 2377 X'00000949'
Hodnota pole ExitReason poskytnutá ve volání MQXEP (exit entry point registration call) je chybná.

Zkontrolujte hodnotu pole ExitReason , abyste určili a opravili chybnou hodnotu příčiny ukončení.

MQRC_RESERVED_VALUE_ERROR 2378 X'0000094A'
Hodnota vyhrazeného pole je chybná.

Zkontrolujte hodnotu pole Vyhrazeno, abyste určili a opravili hodnotu Vyhrazeno.

Typedefs pro jazyk C

Toto téma poskytuje informace o typedefs přidružených k uživatelské procedury rozhraní API, které jsou k dispozici v jazyce C.

Zde jsou typy jazyka C přidružené k uživatelské procedury rozhraní 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;
```

Volání registrace bodu předání řízení uživatelskému programu (MQXEP)

Pomocí těchto informací získáte informace o vyvolání jazyka MQXEP, MQXEP C a prototypu funkce MQXEP C.

Použijte volání MQXEP pro:

1. Registrujte body vyvolání uživatelské procedury rozhraní API IBM MQ před a po, ve kterých se mají vyvolat funkce uživatelské procedury.
2. Určit vstupní body funkce ukončení
3. Zrušit registraci vstupních bodů funkce ukončení

Obvykle byste měli kódovat volání MQXEP ve funkci uživatelské procedury MQ_INIT_EXIT, ale můžete je zadat v libovolné následné funkci uživatelské procedury.

Pokud použijete volání MQXEP k registraci již registrované funkce uživatelské procedury, druhé volání MQXEP se úspěšně dokončí a nahradí registrovanou funkci uživatelské procedury.

Pokud použijete volání MQXEP k registraci funkce uživatelské procedury s hodnotou NULL, volání MQXEP bude úspěšně dokončeno a funkce uživatelské procedury bude zrušena.

Pokud se volání MQXEP používají k registraci, deregistraci a opětovné registraci určité funkce uživatelské procedury během doby trvání požadavku na připojení, funkce dříve registrované uživatelské procedury se znovu aktivuje. Jakékoli úložiště, které je stále přiděleno a přidruženo k této instanci funkce ukončení, je k dispozici pro použití funkcemi ukončení. (Toto úložiště je obvykle uvolněno během vyvolání funkce ukončení ukončení.)

Rozhraní pro MQXEP je:

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason)
```

kde:

Hconfig (MQHCONFIG)-vstup
Popisovač konfigurace představující uživatelskou proceduru rozhraní API, která zahrnuje sadu inicializovaných funkcí. Tato hodnota je generována správcem front bezprostředně před vyvoláním funkce MQ_INIT_EXIT a je předána v MQAXP každé funkci uživatelské procedury rozhraní API.

ExitReason (MQLONG)-vstup.
Důvod, proč je vstupní bod registrován, z následujících důvodů:

- Inicializace nebo ukončení úrovně připojení (MQXR_CONNECTION)
- Před voláním rozhraní API IBM MQ (MQXR_BEFORE)
- Po volání rozhraní API IBM MQ (MQXR_AFTER)

Funkce (MQLONG)-vstup
Identifikátor funkce, platné hodnoty, pro které jsou konstanty MQXF_ * (viz “Externí konstanty” na stránce 1564).

EntryPoint (PMQFUNC)-vstup
Adresa vstupního bodu pro funkci ukončení, která má být registrována. Hodnota NULL označuje buď, že funkce ukončení nebyla poskytnuta, nebo že předchozí registrace funkce ukončení byla zrušena.

ExitOpts(MQXEPO)
Uživatelské procedury rozhraní API mohou určovat volby, které řídí způsob registrace uživatelských procedur rozhraní API. Je-li pro toto pole zadán ukazatel Null, předpokládají se výchozí hodnoty struktury MQXEPO.

CompCode (MQLONG)-výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny, který kvalifikuje kód dokončení.

Je-li kód dokončení MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED:

MQRC_HCONFIG_ERROR
(2280, X'8E8') Zadaný popisovač konfigurace je neplatný. Použijte popisovač konfigurace z MQAXP.

MQRC_EXIT_REASON_ERROR
(2377, X' 949 ') Dodaný důvod vyvolání funkce ukončení je buď neplatný, nebo není platný pro dodaný identifikátor funkce ukončení.

Buď použijte jednu z platných příčin vyvolání funkce ukončení (hodnota MQXR_ *), nebo použijte platnou kombinaci identifikátoru funkce a příčiny ukončení. (Viz Tabulka 836 na stránce 1567.)

MQRC_FUNCTION_ERROR
(2281, X'8E9') Zadaný identifikátor funkce není platný pro příčinu ukončení rozhraní API. V následující tabulce jsou uvedeny platné kombinace identifikátorů funkcí a ExitReasons.

Tabulka 836. Platné kombinace identifikátorů funkcí a ExitReasons

Funkce	**ExitReason**
MQXF_INIT MQXF_TERM	MQXR_CONNECTION

Tabulka 836. Platné kombinace identifikátorů funkcí a ExitReasons (pokračování)	
Funkce	**ExitReason**
MQXF_CONN MQXF_CONNX MQXF_DISC MQXF_OPEN MQXF_CLOSE MQXF_PUT1 MQXF_PUT MQXF_GET MQXF_INQ MQXF_SET MQXF_BEGIN MQXF_CMIT MQXF_BACK MQXF_STAT MQXF_CB MQXF_CTL Zpětné volání MQXF_CALLBACK MQXF_SUB MQXF_SUBRQ	MQXR_BEFORE (předchozí) MQXR_AFTER
MQXF_DATA_CONV_ON_GET	MQXR_BEFORE (předchozí)

MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Pokus o registraci nebo zrušení registrace funkce ukončení se nezdařil kvůli problému prostředku.

Chyba MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Pokus o registraci nebo zrušení registrace funkce ukončení neočekávaně selhal.

MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Neplatný název ExitProperties .

MQRC_XEPO_CHYBA
(2507, X'09CB') Struktura voleb ukončení není platná.

Vyvolání jazyka MQXEP C

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason);
```

Deklarace pro seznam parametrů:

```
    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 */
```

Prototyp funkce MQXEP C

```
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 */
```

Funkce ukončení

Tento oddíl poskytuje některé obecné informace, které vám pomohou při používání volání funkce, a popisuje, jak vyvolat jednotlivé funkce ukončení.

Tyto informace slouží k pochopení obecných pravidel pro uživatelské procedury rozhraní API a k nastavení a vyčištění prováděcího prostředí uživatelské procedury.

Obecná pravidla pro uživatelské procedury rozhraní API

Při vyvolání uživatelských procedur rozhraní API platí následující obecná pravidla:

- Ve všech případech jsou funkce uživatelské procedury rozhraní API řízeny před ověřením parametrů volání rozhraní API a před jakýmikoli kontrolami zabezpečení (v případě MQCONN, MQCONNX nebo MQOPEN).
- Hodnoty polí zadaných do uživatelské procedury a výstupu z ní jsou:
  - Na vstupu do funkce uživatelské procedury rozhraní API *před* IBM MQ může být hodnota pole nastavena aplikačním programem nebo předchozím vyvoláním funkce uživatelské procedury.
  - Na výstupu funkce uživatelské procedury rozhraní API *před* IBM MQ může být hodnota pole ponechána beze změny nebo nastavena na jinou hodnotu funkcí uživatelské procedury.
  - Na vstupu do funkce uživatelské procedury rozhraní API *po* IBM MQ může být hodnotou pole hodnota nastavená správcem front po zpracování volání rozhraní API IBM MQ nebo může být nastavena na hodnotu vyvoláním funkce předchozí uživatelské procedury v řetězci funkcí uživatelské procedury.
  - Na výstupu funkce uživatelské procedury volání rozhraní API *po* IBM MQ může být hodnota pole ponechána beze změny nebo nastavena na jinou hodnotu funkcí uživatelské procedury.
- Funkce ukončení musí komunikovat se správcem front pomocí polí ExitResponse a ExitResponse2 .
- Pole CompCode a Kód příčiny komunikují zpět s aplikací. Funkce správce front a uživatelské procedury mohou nastavit pole CompCode a Kód příčiny.
- Volání MQXEP vrací funkce ukončení, které volají MQXEP, nové kódy příčiny. Funkce ukončení však mohou tyto nové kódy příčiny překládat do všech existujících kódů příčin, kterým mohou stávající a nové aplikace porozumět.
- Každý prototyp funkce ukončení má podobné parametry jako funkce rozhraní API s další úrovní nesměru s výjimkou parametru CompCode a Reason.
- Uživatelské procedury rozhraní API mohou vyvolávat volání MQI (kromě MQDISC), ale tato volání MQI sama nevyvolávají uživatelské procedury rozhraní API.

Všimněte si, že bez ohledu na to, zda je aplikace na serveru nebo klientovi, nemůžete předpovědět řazení volání uživatelské procedury rozhraní API. Volání BEFORE uživatelské procedury rozhraní API nemusí být okamžitě následováno voláním funkce AFTER .

Za voláním funkce BEFORE může následovat jiné volání BEFORE . Příklad:

PŘED MQCTL
Zpětné volání BEFORE
PŘED PŘÍKAZEM MQPUT
PO PROVEDENÍ PŘÍKAZU MQPUT
Zpětné volání AFTER
Po provedení operace MQCTL

, nebo

PŘED XAOPEN

PŘED MQCONNX
AFTER MQCONNX
PO OTEVŘENÍ XAOPEN

V klientu existuje uživatelská procedura, která může upravit chování volání MQCONN nebo MQCONNX s názvem uživatelská procedura `PreConnect` . Uživatelská procedura `PreConnect` může upravit libovolný z parametrů volání MQCONN nebo MQCONNX včetně názvu správce front. Klient nejprve volá tuto uživatelskou proceduru a poté vyvolá volání MQCONN nebo MQCONNX. Všimněte si, že pouze počáteční volání MQCONN nebo MQCONNX vyvolá uživatelskou proceduru rozhraní API; jakákoli následná volání opětovného připojení nemají žádný vliv.

Prováděcí prostředí

Obecně jsou všechny chyby z funkcí ukončení sdělovány zpět obslužné rutině ukončení pomocí polí ExitResponse a ExitResponse2 v MQAXP.

Tyto chyby jsou následně převedeny na hodnoty MQCC_ * a MQRC_ * a předány zpět aplikaci v polích CompCode a Reason. Avšak všechny chyby zjištěné v logice obslužné rutiny uživatelské procedury jsou komunikovány zpět do aplikace jako hodnoty MQCC_ * a MQRC_ * v polích CompCode a Kód příčiny.

Pokud funkce MQ_TERM_EXIT vrátí chybu:

- Volání MQDISC již bylo provedeno.
- Není žádná jiná příležitost řídit *po funkci uživatelské procedury* MQ_TERM_EXIT (a tudíž provést vyčištění prováděcího prostředí uživatelské procedury)
- Vyčištění prováděcího prostředí ukončení se neprovádí

Uživatelskou proceduru nelze uvolnit, protože se stále používá. Také další registrované uživatelské procedury dále v řetězci ukončení, pro které byla uživatelská procedura *před* úspěšná, budou řízeny v opačném pořadí.

Nastavení prováděcího prostředí ukončení

Při zpracování explicitního volání MQCONN nebo MQCONNX nastavuje logika zpracování ukončení prostředí pro provedení uživatelské procedury před vyvoláním funkce inicializace uživatelské procedury (MQ_INIT_EXIT). Nastavení prováděcího prostředí ukončení zahrnuje načtení uživatelské procedury, získání úložiště pro a inicializaci struktur parametrů uživatelské procedury. Manipulátor konfigurace ukončení je také přidělen.

Dojde-li během této fáze k chybám, volání MQCONN nebo MQCONNX se nezdaří s kódem CompCode MQCC_FAILED a jedním z následujících kódů příčiny:

MQRC_API_EXIT_LOAD_ERROR
Pokus o načtení modulu uživatelské procedury rozhraní API se nezdařil.

MQRC_API_EXIT_NOT_FOUND
Funkce uživatelské procedury rozhraní API nebyla v modulu uživatelské procedury rozhraní API nalezena.

MQRC_STORAGE_NOT_AVAILABLE
Pokus o inicializaci prováděcího prostředí pro funkci uživatelské procedury rozhraní API se nezdařil, protože byla k dispozici nedostatečná paměť.

MQRC_API_EXIT_INIT_ERROR
Při inicializaci prováděcího prostředí pro funkci uživatelské procedury rozhraní API došlo k chybě.

Vyčištění prováděcího prostředí ukončení

Při zpracování explicitního volání MQDISC nebo implicitního požadavku na odpojení v důsledku ukončení aplikace může být po vyvolání funkce ukončení uživatelské procedury (MQ_TERM_EXIT), pokud je registrována, nutné vyčistit prostředí pro provádění uživatelské procedury.

Vyčištění prováděcího prostředí ukončení zahrnuje uvolnění úložiště pro struktury parametrů ukončení, případně odstranění všech modulů dříve načtených do paměti.

Pokud během této fáze dojde k chybám, explicitní volání MQDISC selže s kódem CompCode MQCC_FAILED a následujícím kódem příčiny (chyby nejsou zvýrazněny v implicitních požadavcích na odpojení):

MQRC_API_EXIT_TERM_ERROR
Při zavírání prováděcího prostředí pro funkci uživatelské procedury rozhraní API došlo k chybě. Uživatelská procedura by neměla vracet žádné selhání z MQDISC před nebo po volání funkce uživatelské procedury rozhraní API MQ_TERM*.

Uživatelské procedury rozhraní API na klientech

Klient používá uživatelskou proceduru PreConnect k úpravě chování volání MQCONN a MQCONNX a nepodporuje vlastnosti uživatelské procedury rozhraní API.

Ukončení PreConnect

V klientu lze pomocí uživatelské procedury PreConnect vyhledat definici kanálu z centrálního úložiště, například ze serveru LDAP.

Uživatelská procedura PreConnect může také upravit libovolný parametr nebo všechny parametry samotného volání MQCONN nebo MQCONNX, například název správce front.

V případě klientských aplikací musí být před uživatelskou procedurou rozhraní API volána uživatelská procedura PreConnect , protože uživatelská procedura rozhraní API MQCONN nebo MQCONNX je volána pouze v případě, že je znám název správce front a tento název lze změnit pomocí uživatelské procedury PreConnect .

Všimněte si, že pouze počáteční volání MQCONN nebo MQCONNX vyvolá uživatelskou proceduru.

Vlastnosti uživatelské procedury rozhraní API

Na serveru mohou uživatelské procedury rozhraní API registrovat strukturu MQXEPO při inicializaci. Struktura MQXEPO obsahuje pole ExitProperties , které podrobně popisuje skupinu vlastností, o které má uživatelská procedura zájem. To má za následek generování samostatného popisovače vlastnosti zprávy, se kterým může uživatelská procedura manipulovat odděleně od libovolného popisovače vlastnosti zprávy aplikace.

Na klientovi nejsou podporovány vlastnosti uživatelské procedury rozhraní API. Dojde-li k pokusu o registraci názvu skupiny vlastností na klientovi, funkce selže s kódem příčiny MQRC_EXIT_PROPS_NOT_SUPPORTED.

Vrácení-MQ_BACK_EXIT

MQ_BACK_EXIT poskytuje funkci ukončení vrácení, která má provést *před* a *po* odvolání. Použijte identifikátor funkce MQXF_BACK s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* návratových funkcích volání.

Rozhraní k této funkci je:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Začátek-MQ_BEGIN_EXIT

MQ_BEGIN_EXIT poskytuje funkci uživatelské procedury pro zahájení, která má být provedena *před* a *po zpracování volání* MQBEGIN. Použijte identifikátor funkce MQXF_BEGIN s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích uživatelské procedury volání MQBEGIN.

Rozhraní k této funkci je:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pBeginVolby (PMQBO)-vstup/výstup
Ukazatel na počáteční volby.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
        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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Zpětné volání-MQ_CALLBACK_EXIT

MQ_CALLBACK_EXIT poskytuje funkci ukončení pro provedení *před* a *po* zpracování zpětného volání. Použijte identifikátor funkce MQXF_CALLBACK s příčinami uživatelské procedury MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích uživatelské procedury volání zpětného volání.

Rozhraní k této funkci je:

```
MQ_CALLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts,
                  &pBuffer, &pMQCBContext)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení

Hconn (MQHCONN)-vstupní/výstupní
Manipulátor připojení

pMsgPopis
deskriptor zprávy

pGetMsgOpts
Volby, které řídí akci MQGET

pBuffer
Oblast, která má obsahovat data zprávy

pMQCBContext
Kontextová data pro zpětné volání

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
        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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts, &pBuffer,
               &pContext);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 contain the message data */
PPMQCBC   ppContext;)    /* Context data for the callback */
```

Poznámky k použití

1. Uživatelská procedura zpětného volání je vyvolána před vyvoláním spotřebitele a po dokončení funkce spotřebitele. Ačkoli jsou struktury MQMD a MQGMO upravitelné, změna hodnot v uživatelské proceduře před opětovným načtením zprávy z fronty neřídí, protože zpráva již byla odebrána z fronty, která má být doručena do funkce spotřebitele.

Správa funkcí zpětného volání-MQ_CB_EXIT

MQ_CB_EXIT poskytuje funkci ukončení pro provedení *před* a *po* volání MQCB. Použijte identifikátor funkce MQXF_CB s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích ukončení volání MQCB.

Rozhraní k této funkci je:

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &pCallbackDesc,
            &Hobj, &pMsgDesc, &pGetMsgOpts, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení

Hconn (MQHCONN)-vstupní/výstupní
Manipulátor připojení

Operace (MQLONG)-vstup/výstup
Hodnota operace

pCallbackPopis (PMQCBD)-vstup/výstup
Deskriptor zpětného volání

Hobj (MQHOBJ)-vstup/výstup
Popisovač objektu

pMsgDesc (PMQMD)-vstup/výstup
deskriptor zprávy

pGetMsgOpts (PMQGMO)-vstup/výstup
Volby, které řídí akci MQCB

CompCode (MQLONG)-vstup/výstup
Kód dokončení

Příčina (MQLONG)-vstup/výstup
Kvalifikující kód příčiny CompCode

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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. */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &Hobj, &pMsgDesc,
            &pGetMsgOpts, &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Zavřít-MQ_CLOSE_EXIT

MQ_CLOSE_EXIT poskytuje funkci ukončení zavření, která má provést *před* a *po* zpracování volání MQCLOSE. Použijte identifikátor funkce MQXF_CLOSE s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích uživatelské procedury volání MQCLOSE.

Rozhraní k této funkci je:

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHobj,
               &Options, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pHobj (PMQHOBJ)-vstup
Ukazatel na popisovač objektu.

Volby (MQLONG)-vstup/výstup
Volby zavření.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext,&Hconn, &pHobj, &Options,
               &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Potvrzení-MQ_CMIT_EXIT

MQ_CMIT_EXIT poskytuje funkci uživatelské procedury potvrzení pro provedení *před* a *po* zpracování potvrzení. Použijte identifikátor funkce MQXF_CMIT s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* potvrzení funkcí uživatelské procedury volání.

Pokud operace potvrzení selže a transakce je vrácena zpět, volání MQCMIT selže s MQCC_WARNING a MQRC_BACKED_OUT. Tyto návratové kódy a kódy příčiny jsou předány do libovolného *po* funkcích uživatelské procedury MQCMIT, aby funkce uživatelské procedury daly indikaci, že jednotka práce byla vrácena zpět.

Rozhraní k této funkci je:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
        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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext,&Hconn, &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Poznámky k použití

1. Zde popsané rozhraní funkce MQ_GET_EXIT se používá jak pro funkci uživatelské procedury MQXF_GET, tak pro funkci uživatelské procedury “MQXF_DATA_CONV_ON_GET” na stránce 1584 .

   Pro tyto dvě funkce ukončení jsou definovány samostatné vstupní body, takže pro zachycení *obou* volání MQXEP musí být použito dvakrát; pro toto volání použijte identifikátor funkce MQXF_GET.

   Vzhledem k tomu, že rozhraní MQ_GET_EXIT je stejné pro MQXF_GET i MQXF_DATA_CONV_ON_GET, lze použít funkci jedné uživatelské procedury pro obojí; pole *Function* ve struktuře MQAXP označuje, která funkce uživatelské procedury byla vyvolána. Alternativně lze volání MQXEP použít k registraci různých funkcí ukončení pro tyto dva případy.

Připojit a připojit rozšíření-MQ_CONNX_EXIT

MQ_CONNX_EXIT poskytuje funkci uživatelské procedury připojení k provedení *před* a *po zpracování* MQCONN a funkci uživatelské procedury rozšíření připojení k provedení *před* a *po zpracování* MQCONNX.

Stejné rozhraní, jak je zde popsáno, je vyvoláno pro funkce uživatelské procedury volání MQCONN i MQCONNX.

Když agent kanálu zpráv (MCA) odpovídá na příchozí připojení klienta, může se agent MCA připojit a provést určitý počet volání rozhraní API systému IBM MQ dříve, než bude stav klienta plně znám. Tato rozhraní API volají funkce uživatelské procedury rozhraní API s MQAXC na základě samotného programu MCA (například v polích UserId a ConnectionName MQAXC).

Když agent MCA odpovídá na následná příchozí volání rozhraní API klienta, je struktura MQAXC založena na příchozím klientovi a odpovídajícím způsobem nastavuje pole UserId a ConnectionName .

Název správce front nastavený aplikací ve volání MQCONN nebo MQCONNX je předán základnímu volání připojení. Jakýkoli pokus *před* MQ_CONNX_EXIT o změnu názvu správce front nemá žádný vliv.

Použijte identifikátory funkcí MQXF_CONN a MQXF_CONNX s důvody ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích ukončení volání MQCONN a MQCONNX.

Uživatelská procedura MQ_CONNX_EXIT volaná z důvodu MQXR_BEFORE *nesmí* vydávat žádná volání rozhraní API IBM MQ , protože v tuto chvíli nebylo nastaveno správné prostředí.

MQ_CONNX_EXIT nemůže volat MQDISC z volání uživatelské procedury rozhraní API pro připojení, pro které se volá. Toto omezení platí pro uživatelské procedury rozhraní API klienta i serveru.

Rozhraní MQCONN a MQCONNX je identické:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOpts,
&pHconn, &CompCode, &Reason);
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

pQMgrNázev (PMQCHAR)-vstup
Ukazatel na název správce front zadaný ve volání MQCONNX. Uživatelská procedura nesmí změnit tento název ve volání MQCONN nebo MQCONNX.

pConnectVolby (PMQCNO)-vstupní/výstupní
Ukazatel na volby, které řídí akci volání MQCONNX.

Podrobnosti viz "MQCNO-Volby připojení" na stránce 318.

Pro funkci uživatelské procedury MQXF_CONN ukazuje volba pConnectna výchozí strukturu voleb připojení (MQCNO_DEFAULT).

pHconn (PMQHCONN)-vstup
Ukazatel na manipulátor připojení.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení)

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOps,
               &pHconn, &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Poznámky k použití

1. Zde popsané rozhraní funkce MQ_CONNX_EXIT se používá jak pro volání MQCONN, tak pro volání MQCONNX. Pro tato dvě volání jsou však definovány samostatné vstupní body. Chcete-li zachytit *obě* volání, musí být volání MQXEP použito alespoň dvakrát-jednou s identifikátorem funkce MQXF_CONN a znovu s MQXF_CONNX.

   Vzhledem k tomu, že rozhraní MQ_CONNX_EXIT je pro MQCONN a MQCONNX stejné, lze pro obě volání použít funkci jedné uživatelské procedury; pole *Function* ve struktuře MQAXP označuje probíhající volání. Alternativně lze volání MQXEP použít k registraci různých funkcí ukončení pro obě volání.
2. Když agent kanálu zpráv (MCA) odpovídá na příchozí připojení klienta, může agent MCA zadat určitý počet volání produktu MQ dříve, než bude stav klienta plně znám. Tato volání produktu MQ vedou k vyvolání funkcí uživatelské procedury rozhraní API se strukturou MQAXC obsahující data související s agentem MCA, nikoli s klientem (například identifikátor uživatele a název připojení). Jakmile je však stav klienta plně znám, následná volání produktu MQ vedou k vyvolání funkcí uživatelské procedury rozhraní API s příslušnými daty klienta ve struktuře MQAXC.
3. Všechny funkce MQXR_BEFORE jsou vyvolány před provedením ověřování parametrů správcem front. Parametry mohou být proto neplatné (včetně neplatných ukazatelů pro adresy parametrů).

   Funkce MQ_CONNX_EXIT je vyvolána před provedením všech kontrol autorizace správcem front.
4. Funkce uživatelské procedury nesmí měnit název správce front určeného ve volání MQCONN nebo MQCONNX. Pokud je název změněn funkcí ukončení, výsledky nejsou definovány.
5. Funkce ukončení MQXR_BEFORE pro MQ_CONNX_EXIT nemůže vydávat jiná volání MQ než MQXEP.

Zpětné volání řízení-MQ_CTL_EXIT

MQ_CTL_EXIT poskytuje funkci uživatelské procedury požadavku na odběr, kterou lze provést *před* a *po* zpracování řídicího zpětného volání. Použijte identifikátor funkce MQXF_CTL s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* řídicích funkcích procedury volání zpětného volání.

Rozhraní k této funkci je:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason)
```

kde parametry jsou:

Hconn (MQHCONN)-vstupní/výstupní
Manipulátor připojení.

Vstup/výstup operace (MQLONG)
Operace, která se zpracovává na zpětném volání definovaném pro uvedený popisovač objektu

ControlOpts (MQCTLO) vstupní/výstupní
Volby, které řídí akci MQCTL

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Odpojit-MQ_DISC_EXIT

MQ_DISC_EXIT poskytuje funkci uživatelské procedury odpojení, která má provést *před* a *po* zpracování uživatelské procedury MQDISC. Použijte identifikátor funkce MQXF_DISC s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích ukončení volání MQDISC.

Rozhraní k této funkci je

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
&CompCode, &Reason);
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

pHconn (PMQHCONN)-vstup
Ukazatel na manipulátor připojení.

*Pro volání MQDISC před*je hodnota tohoto pole jedna z následujících:

- Manipulátor připojení vrácený při volání MQCONN nebo MQCONNX
- Nula, pro prostředí, kde se adaptér specifický pro prostředí připojil ke správci front
- Hodnota nastavená předchozím vyvoláním funkce ukončení

*Pro volání MQDISC po*je hodnota tohoto pole nula nebo hodnota nastavená předchozím vyvoláním funkce ukončení.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
        MQAXP           ExitParms;      /* Exit parameter structure */
        MQAXC           ExitContext;    /* Exit context structure */
        PMQHCONN        pHconn;         /* Ptr to Connection handle */
        MQLONG          CompCode;       /* Completion code */
        MQLONG          Reason;         /* Reason code */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
              &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Získat-MQ_GET_EXIT

Modul MQ_GET_EXIT poskytuje funkci uživatelské procedury get, kterou lze provést *před* a *po zpracování volání* MQGET.

Existují dva identifikátory funkce:

1. Použijte příkaz MQXF_GET s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích uživatelské procedury volání MQGET.
2. Informace o použití identifikátoru funkce MQXF_DATA_CONV_ON_GET viz “MQXF_DATA_CONV_ON_GET” na stránce 1584 .

Rozhraní k této funkci je:

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Hobj (MQHOBJ)-vstup/výstup
Popisovač objektu.

pMsgDesc (PMQMD)-vstup/výstup
Ukazatel na deskriptor zprávy.

pGetMsgOpts (PMQGMO)-vstup/výstup
Ukazatel pro získání voleb zprávy.

BufferLength (MQLONG)-vstup/výstup
Délka vyrovnávací paměti zpráv.

pBuffer (PMQBYTE)-vstupní/výstupní
Ukazatel na vyrovnávací paměť zpráv.

pDataDélka (PMQLONG)-vstup/výstup
Ukazatel na pole délky dat.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
```

```
              &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
              &CompCode, &Reason)
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Poznámky k použití

1. Zde popsané rozhraní funkce MQ_GET_EXIT se používá jak pro funkci uživatelské procedury MQXF_GET, tak pro funkci uživatelské procedury “MQXF_DATA_CONV_ON_GET” na stránce 1584 .

   Pro tyto dvě funkce ukončení jsou definovány samostatné vstupní body, takže pro zachycení *obou* volání MQXEP musí být použito dvakrát; pro toto volání použijte identifikátor funkce MQXF_GET.

   Vzhledem k tomu, že rozhraní MQ_GET_EXIT je stejné pro MQXF_GET i MQXF_DATA_CONV_ON_GET, lze použít funkci jedné uživatelské procedury pro obojí; pole *Function* ve struktuře MQAXP označuje, která funkce uživatelské procedury byla vyvolána. Alternativně lze volání MQXEP použít k registraci různých funkcí ukončení pro tyto dva případy.

MQXF_DATA_CONV_ON_GET

Identifikátor funkce MQXF_DATA_CONV_ON_GET se používá s MQ_GET_EXIT.

Informace o rozhraní pro toto volání a ukázkovou deklaraci jazyka C viz MQ_GET_EXIT .

Poznámky k použití

Je-li registrován, je tento vstupní bod volán, když zprávy dorazí do aplikace, ale dříve, než dojde k převodu dat. To může být užitečné, pokud uživatelská procedura rozhraní API potřebuje provést zpracování, jako např. dešifrování nebo dekompresi, než je zpráva předána do konverze dat. Uživatelská procedura může v případě potřeby způsobit vynechání převodu dat vrácením příkazu MQXCC_SUPPRESS_FUNCTION; další informace naleznete ve struktuře MQAXP .

Registrace pro tento vstupní bod na klientovi má za následek, že konverze dat bude provedena lokálně na klientském počítači. Pro správnou operaci proto může být nezbytné nainstalovat uživatelské procedury pro převod aplikací na klienta. Všimněte si, že MQXF_DATA_CONV_ON_GET se také používá pro asynchronní spotřebu.

Při použití volání MQ_GET_EXITpoužijte příkaz MQXF_DATA_CONV_ON_GET s příčinou ukončení MQXR_BEFORE k registraci *před* funkcí uživatelské procedury převodu dat MQGET.

Neexistuje žádná funkce uživatelské procedury MQXR_AFTER pro MQXF_DATA_CONV_ON_GET; funkce uživatelské procedury MQXR_AFTER pro MQXF_GET poskytuje požadovanou schopnost zpracování uživatelské procedury po převodu dat.

Pro volání MQ_GET_EXITjsou definovány samostatné vstupní body, takže pro zachycení *obou* funkcí ukončení musí být volání MQXEP použito dvakrát; pro toto volání použijte identifikátor funkce MQXF_DATA_CONV_ON_GET.

Vzhledem k tomu, že rozhraní MQ_GET_EXIT je stejné pro MQXF_GET i MQXF_DATA_CONV_ON_GET, lze použít funkci jedné uživatelské procedury pro obojí; pole *Function* ve struktuře MQAXP označuje, která funkce uživatelské procedury byla vyvolána. Alternativně lze volání MQXEP použít k registraci různých funkcí ukončení pro tyto dva případy.

Inicializace-MQ_INIT_EXIT

MQ_INIT_EXIT poskytuje inicializaci na úrovni připojení, což je indikováno nastavením parametru ExitReason v MQAXP na MQXR_CONNECTION.

Během inicializace si všimněte následujícího:

- Funkce MQ_INIT_EXIT volá MQXEP k registraci příkazových slov rozhraní API IBM MQ a bodů ENTRY a EXIT, o které má zájem.
- Uživatelské procedury nemusí zachycovat všechna příkazová slova rozhraní IBM MQ API. Funkce ukončení jsou vyvolány pouze v případě, že byl zaregistrován zájem.
- Paměť, která má být použita uživatelskou procedurou, lze získat při inicializaci.
- Pokud volání této funkce selže, volání MQCONN nebo MQCONNX, které jej vyvolalo, se také nezdaří s kódem CompCode a důvodem, který závisí na hodnotě pole ExitResponse v MQAXP.
- Uživatelská procedura MQ_INIT_EXIT nesmí vydávat volání rozhraní API IBM MQ , protože v tuto chvíli nebylo nastaveno správné prostředí.
- Pokud dojde k selhání MQ_INIT_EXIT s MQXCC_FAILED, vrátí se správce front z volání MQCONN nebo MQCONNX, které jej volalo s MQCC_FAILED a MQRC_API_EXIT_ERROR.
- Pokud správce front zjistí chybu při inicializaci prováděcího prostředí funkce uživatelské procedury rozhraní API před vyvoláním prvního MQ_INIT_EXIT, vrátí se správce front z volání MQCONN nebo MQCONNX, které vyvolalo MQ_INIT_EXIT s MQCC_FAILED a MQRC_API_EXIT_INIT_ERROR.

Rozhraní pro MQ_INIT_EXIT je:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

CompCode (MQLONG)-vstup/výstup
Ukazatel na kód dokončení, platné hodnoty jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Ukazatel na kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Kód dokončení CompCode a příčina vrácená aplikaci závisí na hodnotě pole ExitResponse v prostředí MQAXP.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
        MQAXP           ExitParms;      /* Exit parameter structure */
        MQAXC           ExitContext;    /* Exit context structure */
        MQLONG          CompCode;       /* Completion code */
        MQLONG          Reason;         /* Reason code */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Poznámky k použití

1. Funkce MQ_INIT_EXIT může vyvolat volání MQXEP za účelem registrace adres funkcí uživatelské procedury pro konkrétní volání produktu MQ , která mají být zachycena. Není nutné zachytávat všechna volání produktu MQ ani volání MQXR_BEFORE a MQXR_AFTER. Například sada ukončení se může rozhodnout zachytit pouze volání MQXR_BEFORE příkazu MQPUT.
2. Úložiště, které mají být použity funkcemi ukončení v sadě uživatelských procedur, lze získat pomocí funkce MQ_INIT_EXIT. Funkce ukončení mohou také získat paměť, když jsou vyvolány, podle potřeby. Před ukončením sady ukončení by však měla být uvolněna všechna úložiště; funkce MQ_TERM_EXIT může uvolnit úložiště nebo funkci ukončení vyvolanou dříve.
3. Pokud MQ_INIT_EXIT vrátí MQXCC_FAILED v poli `ExitResponse` MQAXP nebo dojde k selhání jiným způsobem, volání MQCONN nebo MQCONNX, které způsobilo vyvolání MQ_INIT_EXIT, se rovněž nezdaří s parametry **CompCode** a **Reason** nastavenými na příslušné hodnoty.
4. Funkce MQ_INIT_EXIT nemůže vydat jiná volání produktu MQ než MQXEP.

Dotaz-MQ_INQ_EXIT

MQ_INQ_EXIT poskytuje funkci uživatelské procedury dotazu, která má provést *před* a *po* zpracování volání MQINQ. Použijte identifikátor funkce MQXF_INQ s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích ukončení volání MQINQ.

Rozhraní k této funkci je:

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Hobj (MQHOBJ)-vstup
Popisovač objektu.

SelectorCount (MQLONG)-vstup
Počet selektorů

pSelectors (PMQLONG)-vstupní/výstupní
Ukazatel na pole hodnot selektoru.

IntAttrPočet (MQLONG)-vstup
Počet celočíselných atributů.

pIntAttrs (PMQLONG)-vstupní/výstupní
Ukazatel na pole celočíselných hodnot atributu.

CharAttrDélka (MQLONG)-vstup/výstup
Délka pole znakových atributů.

pCharAttrs (PMQCHAR)-vstupní/výstupní
Ukazatel na pole znakových atributů.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Otevřít-MQ_OPEN_EXIT

MQ_OPEN_EXIT poskytuje funkci otevřené uživatelské procedury pro provedení *před* a *po* zpracování volání MQOPEN. Použijte identifikátor funkce MQXF_OPEN s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích uživatelské procedury volání MQOPEN.

Rozhraní k této funkci je

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
&pHobj, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pObjPopis (PMQOD)-vstup/výstup
Ukazatel na deskriptor objektu.

Volby (MQLONG)-vstup/výstup
Otevřít volby.

pHobj (PMQHOBJ)-vstup
Ukazatel na popisovač objektu.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Vložení-MQ_PUT_EXIT

MQ_PUT_EXIT poskytuje funkci uživatelské procedury vložení k provedení *před* a *po* zpracování volání MQPUT. Použijte identifikátor funkce MQXF_PUT s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích uživatelské procedury volání MQPUT.

Rozhraní k této funkci je:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Hobj (MQHOBJ)-vstup/výstup
Popisovač objektu.

pMsgDesc (PMQMD)-vstup/výstup
Ukazatel na deskriptor zprávy.

pPutMsgOpts (PMQPMO)-vstup/výstup
Ukazatel pro vložení voleb zprávy.

BufferLength (MQLONG)-vstup/výstup
Délka vyrovnávací paměti zpráv.

pBuffer (PMQBYTE)-vstupní/výstupní
Ukazatel na vyrovnávací paměť zpráv.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Poznámky k použití

- Zprávy sestavy generované správcem front přeskočí normální zpracování volání. V důsledku toho nemohou být takové zprávy zachyceny funkcí MQ_PUT_EXIT nebo funkcí MQPUT1 . Zprávy sestav generované agentem kanálu zpráv jsou však zpracovány normálně, a proto je lze zachytit pomocí funkce MQ_PUT_EXIT nebo funkce MQ_PUT1_EXIT . Chcete-li zajistit zachycení všech zpráv sestavy generovaných agentem MCA, měli byste použít volby MQ_PUT_EXIT a MQ_PUT1_EXIT .

Put1 - MQ_PUT1_EXIT

MQ_PUT1_EXIT poskytuje funkci uživatelské procedury *vložit pouze jednu zprávu* k provedení *před* a *po zpracování volání* MQPUT1 . Použijte identifikátor funkce MQXF_PUT1 s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* MQPUT1 .

Rozhraní k této funkci je:

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
&pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pObjPopis (PMQOD)-vstup/výstup
Ukazatel na deskriptor objektu.

pMsgDesc (PMQMD)-vstup/výstup
Ukazatel na deskriptor zprávy.

pPutMsgOpts (PMQPMO)-vstup/výstup
Ukazatel pro vložení voleb zprávy.

BufferLength (MQLONG)-vstup/výstup
Délka vyrovnávací paměti zpráv.

pBuffer (PMQBYTE)-vstupní/výstupní
Ukazatel na vyrovnávací paměť zpráv.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
              &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Nastavit-MQ_SET_EXIT

MQ_SET_EXIT poskytuje funkci uživatelské procedury nastavení, která má být provedena *před* a *po* zpracování volání MQSET. Použijte identifikátor funkce MQXF_SET s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích ukončení volání MQSET.

Rozhraní k této funkci je:

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttr, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Hobj (MQHOBJ)-vstup
Popisovač objektu.

SelectorCount (MQLONG)-vstup
Počet selektorů

pSelectors (PMQLONG)-vstupní/výstupní
Ukazatel na pole hodnot selektoru.

IntAttrPočet (MQLONG)-vstup
Počet celočíselných atributů.

pIntAttrs (PMQLONG)-vstupní/výstupní
Ukazatel na pole celočíselných hodnot atributu.

CharAttrDélka (MQLONG)-vstup/výstup
Délka pole znakových atributů.

pCharAttrs (PMQCHAR)-vstupní/výstupní
Ukazatel na hodnoty znakových atributů.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Stav-MQ_STAT_EXIT

MQ_STAT_EXIT poskytuje funkci ukončení stavu pro provedení *před* a *po* zpracování volání MQSTAT. Použijte identifikátor funkce MQXF_STAT s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích uživatelské procedury volání MQSTAT.

Rozhraní k této funkci je:

```
MQ_STAT_EXIT (&ExitParms, &ExitContext, &Hconn, &Type, &pStatus
              &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Typ (MQLONG)-vstup
Typ informací o stavu, které se mají načíst.

pStatus (PMQSTS)-výstup
Ukazatel na vyrovnávací paměť stavu.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Ukončení-MQ_TERM_EXIT

MQ_TERM_EXIT poskytuje ukončení úrovně připojení, registrované s identifikátorem funkce MQXF_TERM a ExitReason MQXR_CONNECTION. Je-li registrován, je MQ_TERM_EXIT volán jednou pro každý požadavek na odpojení.

V rámci ukončení může být uvolněno úložiště, které již není vyžadováno ukončením, a může být provedeno jakékoli požadované vyčištění.

Dojde-li k selhání MQ_TERM_EXIT s MQXCC_FAILED, vrátí se správce front z MQDISC, který jej volal, s MQCC_FAILED a MQRC_API_EXIT_ERROR.

Pokud správce front zjistí chybu při ukončování prováděcího prostředí funkce uživatelské procedury rozhraní API po vyvolání poslední MQ_TERM_EXIT, vrátí se správce front z volání MQDISC, které vyvolalo MQ_TERM_EXIT s MQCC_FAILED a MQRC_API_EXIT_TERM_ERROR

Rozhraní k této funkci je:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Kód dokončení CompCode a příčina vrácená aplikaci závisí na hodnotě pole ExitResponse v prostředí MQAXP.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
        MQAXP           ExitParms;      /* Exit parameter structure */
        MQAXC           ExitContext;    /* Exit context structure */
        MQLONG          CompCode;       /* Completion code */
        MQLONG          Reason;         /* Reason code */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Poznámky k použití

1. Funkce MQ_TERM_EXIT je volitelná. Není nutné, aby ukončovací sada zaregistrovala ukončovací proceduru, pokud neexistuje ukončovací zpracování, které by se mělo provést.

   Pokud funkce patřící do sady uživatelských procedur získají prostředky během připojení, funkce MQ_TERM_EXIT je vhodným bodem pro uvolnění těchto prostředků, například pro dynamické uvolnění úložiště.

2. Pokud je při zadání volání MQDISC registrována funkce MQ_TERM_EXIT, je funkce exit vyvolána po vyvolání všech funkcí uživatelské procedury MQDISC.
3. Pokud MQ_TERM_EXIT vrátí hodnotu MQXCC_FAILED v poli `ExitResponse` prostředí MQAXP nebo jiným způsobem selže, volání MQDISC, které způsobilo vyvolání MQ_TERM_EXIT, se také nezdaří s parametry **CompCode** a **Reason** nastavenými na příslušné hodnoty.

Registrovat odběr-MQ_SUB_EXIT

MQ_SUB_EXIT poskytuje funkci ukončení pro provedení *před* a *po* zpracování opětovné registrace odběru. Použijte identifikátor funkce MQXF_SUB s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích ukončení volání registrace odběru.

Rozhraní k této funkci je:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstupní/výstupní
Manipulátor připojení.

pSubPopis-vstup/výstup
Pole selektorů atributů.

pHobj -vstup/výstup
Popisovač objektu

pHsub (MQHOBJ) vstupní/výstupní
Popisovač odběru

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub,
             &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
        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 */
```

Požadavek na odběr-MQ_SUBRQ_EXIT

MQ_SUBRQ_EXIT poskytuje funkci uživatelské procedury požadavku na odběr k provedení *před* a *po* zpracování požadavku na odběr. Použijte identifikátor funkce MQXF_SUBRQ s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci *před* a *po* funkcích ukončení volání požadavku na odběr.

Rozhraní k této funkci je:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
               &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstupní/výstupní
Manipulátor připojení.

pHsub (MQHOBJ) vstupní/výstupní
Popisovač odběru

Vstup/výstup akce (MQLONG)
Akce

pSubRqOpts (MQSRO)

CompCode (MQLONG)-vstup/výstup
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo

Příčina (MQLONG)-vstup/výstup
Kód příčiny, který kvalifikuje kód dokončení.

Pokud je kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Není důvod k hlášení.

Pokud je kód dokončení MQCC_FAILED nebo MQCC_WARNING, funkce ukončení může nastavit pole kódu příčiny na libovolnou platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
               &CompCode, &Reason);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje funkci ukončení xa_close, která má být provedena před a po zpracování xa_close. Použijte identifikátor funkce MQXF_XACLOSE s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání před a po volání xa_close.

Rozhraní k této funkci je:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXa_info (PMQCHAR)-vstupní/výstupní
Informace o správci prostředků specifickém pro instanci.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje uživatelskou proceduru xa_commit, která má být provedena před a po zpracování xa_commit. Použijte identifikátor funkce MQXF_XACOMMIT s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí uživatelské procedury volání před a po xa_commit.

Rozhraní k této funkci je:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje uživatelskou proceduru xa_complete, která má být provedena před a po zpracování xa_complete. Použijte identifikátor funkce MQXF_XACOMPLETE s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání před a po xa_complete.

Rozhraní k této funkci je:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pHandle (PMQLONG)-vstup/výstup
Ukazatel na asynchronní operaci.

pRetVal (PMQLONG)-vstupní/výstupní
Návratová hodnota asynchronní operace.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje funkci ukončení xa_end, která se má provést před a po zpracování xa_end. Použijte identifikátor funkce MQXF_XAEND s důvody ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání před a po volání xa_end.

Rozhraní k této funkci je:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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_ZAPOMNĚT_EXIT

XA_ZAPOMNĚT_EXIT poskytuje funkci ukončení xa_forget, která má být provedena před a po zpracování xa_forget. Použijte identifikátor funkce MQXF_XAFORGET s důvody ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání před voláním a po volání xa_forget.

Rozhraní k této funkci je:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje funkci uživatelské procedury xa_open, která má být provedena před a po zpracování xa_open. Použijte identifikátor funkce MQXF_XAOPEN s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí uživatelské procedury volání před a po xa_open.

Rozhraní k této funkci je:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXa_info (PMQCHAR)-vstupní/výstupní
Informace o správci prostředků specifickém pro instanci.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje funkci ukončení xa_prepare, která má být provedena před a po zpracování xa_prepare. Použijte identifikátor funkce MQXF_XAPREPARE s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí uživatelské procedury před a po volání xa_prepare.

Rozhraní k této funkci je:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje funkci ukončení xa_recover, která má být provedena před a po zpracování xa_recover. Použijte identifikátor funkce MQXF_XARECOVER s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání před a po volání xa_recover.

Rozhraní k této funkci je:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Počet (MQLONG)-vstup/výstup
Maximální počet identifikátorů XID v poli XID

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje funkci uživatelské procedury xa_rollback, která se má provést před a po zpracování xa_rollback. Použijte identifikátor funkce MQXF_XAROLLBACK s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání před voláním a po volání xa_rollback.

Rozhraní k této funkci je:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje uživatelskou proceduru xa_start, která má být provedena před a po zpracování xa_start. Použijte identifikátor funkce MQXF_XASTART s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání před a po volání xa_start.

Rozhraní k této funkci je:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje funkci ukončení ax_reg pro provedení před a po zpracování ax_reg. Použijte identifikátor funkce MQXF_AXREG s příčinami ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání před a po ax_reg.

Rozhraní k této funkci je:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 poskytuje funkci ukončení ax_unreg pro provedení před a po zpracování ax_unreg. Použijte identifikátor funkce MQXF_AXUNREG s důvody ukončení MQXR_BEFORE a MQXR_AFTER k registraci funkcí ukončení volání ax_unreg před a po.

Rozhraní k této funkci je:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura parametrů ukončení.

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu ukončení.

Rmid (MQLONG)-vstup/výstup
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstup/výstup
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odpověď z volání XA.

Vyvolání jazyka C

Správce front logicky definuje následující proměnné:

```
     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 */
```

Správce front poté logicky volá uživatelskou proceduru následujícím způsobem:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

Váš výstup musí odpovídat následujícímu prototypu funkce C:

```
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 */
```

Obecné informace o vyvolání funkcí ukončení

Toto téma obsahuje některé obecné pokyny, které vám pomohou naplánovat uživatelské procedury, zejména v souvislosti se zpracováním chyb a neočekávaných událostí.

Selhání ukončení

Pokud se funkce ukončení nestandardně ukončí po destruktivním volání MQGET mimo synchronizační bod, ale před předáním zprávy aplikaci, může se obslužná rutina ukončení zotavit ze selhání a předat řízení aplikaci.

V tomto případě může být zpráva ztracena. Toto je podobné tomu, co se stane, když aplikace selže okamžitě po přijetí zprávy z fronty.

Volání MQGET může být dokončeno s MQCC_FAILED a MQRC_API_EXIT_ERROR.

Pokud dojde k nestandardnímu ukončení funkce uživatelské procedury volání rozhraní API *před* , obslužná rutina uživatelské procedury se může zotavit ze selhání a předat řízení aplikaci bez zpracování volání rozhraní API. V této události musí funkce ukončení obnovit všechny prostředky, které vlastní.

Pokud se používají zřetězené uživatelské procedury, lze řídit uživatelské procedury *po* volání rozhraní API pro libovolné *dříve* volání rozhraní API, které bylo úspěšně řízeno, samy o sobě. Volání rozhraní API může selhat s MQCC_FAILED a MQRC_API_EXIT_ERROR.

Příklad ošetření chyb pro funkce ukončení

Následující diagram zobrazuje body (e *N*). při kterém se mohou vyskytnout chyby. Je to pouze příklad, který ukazuje, jak se uživatelské procedury chovají a měly by být čteny společně s následující tabulkou. V tomto příkladu jsou před každým voláním rozhraní API a po něm vyvolány dvě funkce ukončení, které zobrazují chování se zřetězenými uživatelskými procedurami.

```
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
```

V následující tabulce jsou uvedeny akce, které mají být provedeny v jednotlivých chybových bodech. Byla pokryta pouze podmnožina chybových bodů, protože zde uvedená pravidla se mohou vztahovat na všechny ostatní. Jedná se o akce, které určují zamýšlené chování v jednotlivých případech.

Tabulka 837. Chyby uživatelské procedury rozhraní API a příslušné akce, které se mají provést

Err Pt	Popis	Akce
e1	Při nastavování prostředí došlo k chybě.	1. Vrátit nastavení prostředí zpět podle potřeby 2. Nepoužívat funkce ukončení 3. Selhání MQCONN s MQCC_FAILED, MQRC_API_EXIT_LOAD_ERROR

Tabulka 837. Chyby uživatelské procedury rozhraní API a příslušné akce, které se mají provést (pokračování)

Err Pt	Popis	Akce
e2	Funkce MQ_INIT_EXIT byla dokončena s: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED: 1. Vyčistit prostředí 2. Selhání MQCONN s MQCC_FAILED, MQRC_API_EXIT_INIT_ERROR • Pro MQXCC_ * 1. Vystupovat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Vyčistit prostředí
e3	*Před* dokončením funkce MQ_CONNX_EXIT 1: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED: 1. Funkce jednotky MQ_TERM_EXIT 2. Vyčistit prostředí 3. Nezdařilo se volání MQCONN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Vystupovat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. V případě potřeby řídit funkci MQ_TERM_EXIT 3. V případě potřeby vyčistit prostředí
e4	*Před* dokončením funkce MQ_CONNX_EXIT 2: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_CONNX_EXIT 1 2. Funkce jednotky MQ_TERM_EXIT 3. Vyčistit prostředí 4. Nezdařilo se volání MQCONN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Vystupovat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Jednotka *po* funkci MQ_CONNX_EXIT 1, není-li uživatelská procedura potlačena 3. V případě potřeby řídit funkci MQ_TERM_EXIT 4. V případě potřeby vyčistit prostředí
e5	Volání MQCONN se nezdařilo.	1. Předejte MQCONN CompCode a důvod 2. Jednotka *po funkci* MQ_CONNX_EXIT 2, pokud *před* MQ_CONNX_EXIT 2 uspěla a uživatelská procedura není potlačena 3. Jednotka *za funkcí* MQ_CONNX_EXIT 1, pokud *před* MQ_CONNX_EXIT 1 uspěla a uživatelská procedura není potlačena 4. Funkce jednotky MQ_TERM_EXIT 5. Vyčistit prostředí

Tabulka 837. Chyby uživatelské procedury rozhraní API a příslušné akce, které se mají provést (pokračování)

Err Pt	Popis	Akce
e6	*Po* dokončení funkce MQ_CONNX_EXIT 2: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_CONNX_EXIT 1 2. Dokončit volání MQCONN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Vystupovat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. V případě potřeby použijte jednotku *po* funkci MQ_CONNX_EXIT 1.
e7	*Po* dokončení funkce MQ_CONNX_EXIT 1: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED dokončete volání MQCONN s MQCC_FAILED, MQRC_API_EXIT_ERROR • V případě MQXCC_ * se jedná o hodnoty MQXCC_ * a MQXR2_ * [1]
e8	*Před* dokončením funkce MQ_OPEN_EXIT 1: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED dokončete volání MQOPEN s MQCC_FAILED, MQRC_API_EXIT_ERROR • V případě MQXCC_ * se jedná o hodnoty MQXCC_ * a MQXR2_ * [1]
e9	*Před* dokončením funkce MQ_OPEN_EXIT 2: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_OPEN_EXIT 1 2. Dokončit volání MQOPEN s MQCC_FAILED, MQRC_API_EXIT_ERROR • V případě MQXCC_ * se jedná o hodnoty MQXCC_ * a MQXR2_ * [1]
e10	Volání MQOPEN se nezdařilo	1. Předejte příkaz MQOPEN CompCode a důvod 2. Jednotka *po funkci* MQ_OPEN_EXIT 2, není-li uživatelská procedura potlačena 3. Jednotka *po funkci* MQ_OPEN_EXIT 1, není-li uživatelská procedura potlačena a není-li potlačena zřetězená uživatelská procedura
e11	*Po* dokončení funkce MQ_OPEN_EXIT 2: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_OPEN_EXIT 1 2. Dokončit volání MQOPEN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Vystupovat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Jednotka *po* funkci MQ_OPEN_EXIT 1, není-li uživatelská procedura potlačena

Tabulka 837. Chyby uživatelské procedury rozhraní API a příslušné akce, které se mají provést (pokračování)		
Err Pt	**Popis**	**Akce**
e25	*Po* dokončení funkce MQ_DISC_EXIT 2: • MQXCC_FAILED • MQXCC_ *	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_DISC_EXIT 1 2. Funkce jednotky MQ_TERM_EXIT 3. Vyčistit prováděcí prostředí uživatelské procedury 4. Dokončit volání MQDISC s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Vystupovat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Funkce jednotky MQ_TERM_EXIT 3. Vyčistit prováděcí prostředí uživatelské procedury

Poznámka:

1. Hodnoty MQXCC_ * a MQXR2_* a jejich odpovídající akce jsou definovány v části Způsob, jakým správci front zpracovávají funkce ukončení.

Pole ExitResponse jsou nastavena nesprávně

Toto téma poskytuje informace o tom, co se stane, když je pole ExitResponse nastaveno na cokoli kromě podporovaných hodnot.

Je-li pole ExitResponse nastaveno na jinou hodnotu, než je jedna z podporovaných hodnot, platí následující akce:

- Pro funkci uživatelské procedury rozhraní API *před* MQCONN nebo MQDISC:
  - Hodnota ExitResponse2 je ignorována.
  - Nejsou vyvolány žádné další funkce *před* ukončením v řetězci ukončení (pokud existují); samotné volání rozhraní API není vydáno.
  - Pro všechny uživatelské procedury *před* , které byly úspěšně volány, jsou uživatelské procedury *po* volány v opačném pořadí.
  - Je-li registrována, funkce ukončení uživatelské procedury pro ty *před* funkcemi uživatelské procedury MQCONN nebo MQDISC v řetězci, které byly úspěšně vyvolány, jsou řízeny k vyčištění po těchto funkcích uživatelské procedury.
  - Volání MQCONN nebo MQDISC se nezdařilo s chybou MQRC_API_EXIT_ERROR.
- Pro funkci uživatelské procedury rozhraní API *před* IBM MQ jinou než MQCONN nebo MQDISC:
  - Hodnota ExitResponse2 je ignorována.
  - Nejsou vyvolány žádné další funkce *před* nebo *po* převodu dat v řetězci ukončení (pokud existují).
  - Pro všechny uživatelské procedury *před* , které byly úspěšně volány, jsou uživatelské procedury *po* volány v opačném pořadí.
  - Samotné volání rozhraní API IBM MQ není vydáno.
  - Volání rozhraní API IBM MQ se nezdařilo s chybou MQRC_API_EXIT_ERROR.
- Pro *po funkci uživatelské procedury rozhraní API produktu* MQCONN nebo MQDISC:
  - Hodnota ExitResponse2 je ignorována.
  - Zbývající funkce ukončení, které byly úspěšně volány před voláním rozhraní API, jsou volány v opačném pořadí.

  - Je-li registrována, funkce ukončení uživatelské procedury pro tyto funkce *před* nebo *po* funkcích uživatelské procedury MQCONN nebo MQDISC v řetězu, které byly úspěšně vyvolány, jsou řízeny k vyčištění po uživatelské proceduře.
  - Hodnota CompCode závažnější hodnoty MQCC_WARNING a hodnota CompCode vrácená uživatelskou procedurou je vrácena aplikaci.
  - Aplikaci je vrácena příčina MQRC_API_EXIT_ERROR.
  - Volání rozhraní API IBM MQ bylo úspěšně vydáno.
- Pro funkci uživatelské procedury *po volání rozhraní API produktu* IBM MQ jinou než MQCONN nebo MQDISC:
  - Hodnota ExitResponse2 je ignorována.
  - Zbývající funkce ukončení, které byly úspěšně volány před voláním rozhraní API, jsou volány v opačném pořadí.
  - Hodnota CompCode závažnější hodnoty MQCC_WARNING a hodnota CompCode vrácená uživatelskou procedurou je vrácena aplikaci.
  - Aplikaci je vrácena příčina MQRC_API_EXIT_ERROR.
  - Volání rozhraní API IBM MQ bylo úspěšně vydáno.
- Pro převod dat *před* ve funkci get exit:
  - Hodnota ExitResponse2 je ignorována.
  - Zbývající funkce ukončení, které byly úspěšně volány před voláním rozhraní API, jsou volány v opačném pořadí.
  - Zpráva není převedena a nepřevedená zpráva je vrácena aplikaci.
  - Hodnota CompCode závažnější hodnoty MQCC_WARNING a hodnota CompCode vrácená uživatelskou procedurou je vrácena aplikaci.
  - Aplikaci je vrácena příčina MQRC_API_EXIT_ERROR.
  - Volání rozhraní API IBM MQ bylo úspěšně vydáno.

  Poznámka: Protože chyba je s uživatelskou procedurou, je lepší vrátit MQRC_API_EXIT_ERROR než vrátit MQRC_NOT_CONVERTED.

Pokud funkce exit nastaví pole ExitResponse2 na jinou hodnotu, než je jedna z podporovaných hodnot, místo toho se předpokládá hodnota MQXR2_DEFAULT_CONTINUATION .

Referenční informace o rozhraní instalovatelných služeb

Tato kolekce témat poskytuje referenční informace pro instalovatelné služby.

Funkce a datové typy jsou uvedeny v abecedním pořadí v rámci skupiny pro každý typ služby.

Související pojmy

ALW Instalovatelné služby a komponenty pro systémy UNIX, Linux a Windows

IBM i Instalovatelné služby a komponenty pro systém IBM i

IBM i Referenční informace o rozhraní instalovatelných služeb pro systém IBMi

Související úlohy

Rozšíření prostředků správce front

ALW Konfigurace instalovatelných služeb

Jak jsou funkce zobrazeny

Jak jsou dokumentovány funkce instalovatelných služeb.

Pro každou funkci je uveden popis včetně identifikátoru funkce (pro MQZEP).

Parametry jsou uvedeny v pořadí, v jakém se musí vyskytnout. Všichni musí být přítomni.

Každý název parametru je následován svým datovým typem. Jedná se o základní datové typy popsané v souboru “Základní datové typy” na stránce 235.

Vyvolání jazyka C je také poskytnuto po popisu parametrů.

MQZ_AUTHENTICATE_USER-Ověření uživatele

Tuto funkci poskytuje komponenta autorizační služby MQZAS_VERSION_5 a je vyvolána správcem front za účelem ověření uživatele nebo nastavení polí kontextu identity. Vyvolá se při zřízení kontextu uživatelské aplikace IBM MQ .

Kontext aplikace je vytvořen během volání připojení v místě, kde je inicializován uživatelský kontext aplikace, a v každém bodě, kde je změněn uživatelský kontext aplikace. Při každém volání připojení jsou informace o uživatelském kontextu aplikace znovu získány v poli *IdentityContext* .

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_AUTHENTICATE_USER.

Syntaxe

MQZ_AUTHENTICATE_USER (*QMgrName* , *SecurityParms* , *ApplicationContext* , *IdentityContext* , *CorrelationPtr* , *ComponentData* , *Continuation* , *CompCode* , *Příčina*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

SecurityParms
Typ: MQCSP-vstup

Parametry zabezpečení. Data týkající se ID uživatele, hesla a typu ověření. Pokud je atribut AuthenticationType struktury MQCSP uveden jako MQCSP_AUTH_USER_ID_AND_PWD, ID uživatele i heslo jsou porovnány s ekvivalentními poli v parametru IdentityContext (MQZIC), aby se určilo, zda se shodují. Další informace viz téma “MQCSP-parametry zabezpečení” na stránce 337.

Během volání MQCONN MQI tento parametr obsahuje hodnotu null nebo výchozí hodnoty.

ApplicationContext
Typ: MQZAC-vstup

Kontext aplikace. Údaje týkající se volající aplikace. Podrobnosti viz MQZAC-Kontext aplikace .

Během každého volání MQCONN nebo MQCONNX MQI jsou znovu získány informace o kontextu uživatele ve struktuře MQZAC.

IdentityContext
Typ: MQZIC-vstup/výstup

Kontext identity. Na vstupu do funkce ověření uživatele identifikuje aktuální kontext identity. Funkce ověření uživatele to může změnit, což znamená, že správce front převezme nový kontext identity. Další podrobnosti o struktuře MQZIC viz MQZIC-kontext identity .

CorrelationPtr
Typ: MQPTR-výstup

Korelační ukazatel. Určuje adresu všech korelačních dat. Tento ukazatel je následně předán dalším voláním OAM.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru délky ComponentDatavolání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Příznak pokračování. Můžete zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na jiných komponentách.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                   IdentityContext, &CorrelationPtr, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Deklarujte parametry předané službě následujícím způsobem:

```
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-Zkontrolujte oprávnění

Tuto funkci poskytuje komponenta autorizační služby MQZAS_VERSION_1 a spouští ji správce front, aby zkontroloval, zda má entita oprávnění k provedení konkrétní akce nebo akcí na určeném objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_CHECK_AUTHORITY.

Syntaxe

MQZ_CHECK_AUTHORITY(*QMgrName* , *EntityName* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityName
Typ: MQCHAR12 -vstup

Název entity. Název entity, jejíž autorizace k objektu má být zkontrolována. Maximální délka řetězce je 12 znaků; je-li kratší, je zprava vyplněn mezerami. Název není ukončen znakem null.

Není nezbytné, aby byla tato entita známa základní bezpečnostní službě. Není-li známo, použijí se pro kontrolu oprávnění speciální skupiny **nikdo** (do které se předpokládá, že patří všechny entity). Prázdný název je platný a lze jej použít tímto způsobem.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem EntityName. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, ke kterému je požadován přístup. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

Oprávnění
Typ: MQLONG-vstup

Oprávnění ke kontrole. Pokud je kontrolována jedna autorizace, je toto pole shodné s příslušnou operací autorizace (konstanta MQZAO_ *). Pokud se kontroluje více než jedna autorizace, je to bitová OR odpovídajících konstant MQZAO_ *.

Pro použití volání MQI platí následující oprávnění:

MQZAO_CONNECT
Schopnost používat volání MQCONN.

MQZAO_BROWSE
Schopnost použít volání MQGET s volbou procházení.

To umožňuje zadat volbu MQGMO_BROWSE_FIRST, MQGMO_BROWSE_MSG_UNDER_CURSOR nebo MQGMO_BROWSE_NEXT ve volání MQGET.

MQZAO_INPUT
Hlavní. Schopnost použít volání MQGET s volbou vstupu.

To umožňuje zadat volbu MQOOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE nebo MQOO_INPUT_AS_Q_DEF pro volání MQOPEN.

MQZAO_OUTPUT
Schopnost používat volání MQPUT.

To umožňuje zadat volbu MQOO_OUTPUT ve volání MQOPEN.

MQZAO_INQUIRE
Schopnost používat volání MQINQ.

To umožňuje zadat volbu MQO_INQUIRE ve volání MQOPEN.

MQZAO_SET
Schopnost použít volání MQSET.

To umožňuje zadání volby MQOO_SET ve volání MQOPEN.

MQZAO_PASS_IDENTITY_CONTEXT
Schopnost předat kontext identity.

To umožňuje zadat volbu MQOO_PASS_IDENTITY_CONTEXT pro volání MQOPEN a volbu MQPMO_PASS_IDENTITY_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_PASS_ALL_CONTEXT
Schopnost předat celý kontext.

To umožňuje zadat volbu MQOO_PASS_ALL_CONTEXT pro volání MQOPEN a volbu MQPMO_PASS_ALL_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_SET_IDENTITY_CONTEXT
Schopnost nastavit kontext identity.

To umožňuje, aby volba MQOOO_SET_IDENTITY_CONTEXT byla určena pro volání MQOPEN a volba MQPMO_SET_IDENTITY_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_SET_ALL_CONTEXT
Schopnost nastavit celý kontext.

To umožňuje zadat volbu MQOO_SET_ALL_CONTEXT pro volání MQOPEN a volbu MQPMO_SET_ALL_CONTEXT pro volání MQPUT a MQPUT1 .

OPRÁVNĚNÍ uživatele MQZAO_ALTERNATE_USER_AUTHORITY
Schopnost používat alternativní oprávnění uživatele.

To umožňuje zadat volbu MQOOO_ALTERNATE_USER_AUTHORITY pro volání MQOPEN a volbu MQPMO_ALTERNATE_USER_AUTHORITY pro volání MQPUT1 .

MQZAO_ALL_MQI
Všechny autorizace MQI.

To povolí všechny autorizace.

Pro administraci správce front platí následující autorizace:

MQZAO_CREATE
Schopnost vytvářet objekty určeného typu.

MQZAO_DELETE
Schopnost odstranit určený objekt.

MQZAO_DISPLAY
Schopnost zobrazit atributy uvedeného objektu.

MQZAO_CHANGE
Schopnost změnit atributy uvedeného objektu.

MQZAO_CLEAR
Schopnost odstranit všechny zprávy z uvedené fronty.

MQZAO_AUTHORIZE
Schopnost autorizovat ostatní uživatele pro uvedený objekt.

MQZAO_CONTROL
Schopnost spustit nebo zastavit modul listener, službu nebo objekt kanálu, který není klientem, a schopnost provést příkaz ping na objekt kanálu, který není klientem.

MQZAO_CONTROL_EXTENDED
Schopnost resetovat pořadové číslo nebo vyřešit neověřenou zprávu pro objekt kanálu, který není klientem.

MQZAO_ALL_ADMIN
Schopnost nastavit kontext identity.

Všechna administrativní oprávnění jiná než MQZAO_CREATE.

Následující autorizace platí jak pro použití MQI, tak pro administraci správce front:

MQZAO_ALL
Všechna oprávnění, jiná než MQZAO_CREATE.

MQZAO_NONE
Žádná oprávnění.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

Pokud volání komponenty selže (tzn. *CompCode* vrátí MQCC_FAILED) a parametr *Continuation* je MQZCI_DEFAULT nebo MQZCI_CONTINUE, správce front bude i nadále volat další komponenty, pokud existují.

Pokud je volání úspěšné (tj. *CompCode* vrací MQCC_OK), nejsou volány žádné další komponenty bez ohledu na nastavení volby *Continuation* .

Pokud volání selže a parametr *Continuation* je MQZCI_STOP, nejsou volány žádné další komponenty a chyba je vrácena správci front. Komponenty neznají předchozí volání, takže parametr *Continuation* je před voláním vždy nastaven na hodnotu MQZCI_DEFAULT.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                     ObjectType, Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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 -Zkontrolujte oprávnění (rozšířené)

Tuto funkci poskytuje komponenta autorizační služby MQZAS_VERSION_2 a spouští ji správce front, aby zkontroloval, zda má entita oprávnění k provedení konkrétní akce nebo akcí na určeném objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_CHECK_AUTHORITY.

MQZ_CHECK_AUTHORITY_2 je jako MQZ_CHECK_AUTHORITY, ale s parametrem **EntityName** nahrazeným parametrem **EntityData** .

Syntaxe

MQZ_CHECK_AUTHORITY_2(*QMgrName* , *EntityData* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity s oprávněním k objektu, který má být zkontrolován. Podrobnosti viz “MQZED-Deskriptor entit” na stránce 1672.

Není nezbytné, aby byla tato entita známa základní bezpečnostní službě. Není-li známo, použijí se pro kontrolu oprávnění speciální skupiny **nikdo** (do které se předpokládá, že patří všechny entity). Prázdný název je platný a lze jej použít tímto způsobem.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityData*. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, ke kterému je požadován přístup. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Oprávnění ke kontrole. Pokud je kontrolována jedna autorizace, je toto pole shodné s příslušnou operací autorizace (konstanta MQZAO_ *). Pokud se kontroluje více než jedna autorizace, je to bitová OR odpovídajících konstant MQZAO_ *.

Pro použití volání MQI platí následující oprávnění:

MQZAO_CONNECT
Schopnost používat volání MQCONN.

MQZAO_BROWSE
Schopnost použít volání MQGET s volbou procházení.

To umožňuje zadat volbu MQGMO_BROWSE_FIRST, MQGMO_BROWSE_MSG_UNDER_CURSOR nebo MQGMO_BROWSE_NEXT ve volání MQGET.

MQZAO_INPUT
Hlavní. Schopnost použít volání MQGET s volbou vstupu.

To umožňuje zadat volbu MQOOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE nebo MQOO_INPUT_AS_Q_DEF pro volání MQOPEN.

MQZAO_OUTPUT
Schopnost používat volání MQPUT.

To umožňuje zadat volbu MQOO_OUTPUT ve volání MQOPEN.

MQZAO_INQUIRE
Schopnost používat volání MQINQ.

To umožňuje zadat volbu MQO_INQUIRE ve volání MQOPEN.

MQZAO_SET
Schopnost použít volání MQSET.

To umožňuje zadání volby MQOO_SET ve volání MQOPEN.

MQZAO_PASS_IDENTITY_CONTEXT
Schopnost předat kontext identity.

To umožňuje zadat volbu MQOO_PASS_IDENTITY_CONTEXT pro volání MQOPEN a volbu MQPMO_PASS_IDENTITY_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_PASS_ALL_CONTEXT
Schopnost předat celý kontext.

To umožňuje zadat volbu MQOO_PASS_ALL_CONTEXT pro volání MQOPEN a volbu MQPMO_PASS_ALL_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_SET_IDENTITY_CONTEXT
Schopnost nastavit kontext identity.

To umožňuje, aby volba MQOOO_SET_IDENTITY_CONTEXT byla určena pro volání MQOPEN a volba MQPMO_SET_IDENTITY_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_SET_ALL_CONTEXT
Schopnost nastavit celý kontext.

To umožňuje zadat volbu MQOO_SET_ALL_CONTEXT pro volání MQOPEN a volbu MQPMO_SET_ALL_CONTEXT pro volání MQPUT a MQPUT1 .

OPRÁVNĚNÍ uživatele MQZAO_ALTERNATE_USER_AUTHORITY
Schopnost používat alternativní oprávnění uživatele.

To umožňuje zadat volbu MQOOO_ALTERNATE_USER_AUTHORITY pro volání MQOPEN a volbu MQPMO_ALTERNATE_USER_AUTHORITY pro volání MQPUT1 .

MQZAO_ALL_MQI
Všechny autorizace MQI.

To povolí všechny autorizace.

Pro administraci správce front platí následující autorizace:

MQZAO_CREATE
Schopnost vytvářet objekty určeného typu.

MQZAO_DELETE
Schopnost odstranit určený objekt.

MQZAO_DISPLAY
Schopnost zobrazit atributy uvedeného objektu.

MQZAO_CHANGE
Schopnost změnit atributy uvedeného objektu.

MQZAO_CLEAR
Schopnost odstranit všechny zprávy z uvedené fronty.

MQZAO_AUTHORIZE
Schopnost autorizovat ostatní uživatele pro uvedený objekt.

MQZAO_CONTROL
Schopnost spustit nebo zastavit modul listener, službu nebo objekt kanálu, který není klientem, a schopnost provést příkaz ping na objekt kanálu, který není klientem.

MQZAO_CONTROL_EXTENDED
Schopnost resetovat pořadové číslo nebo vyřešit neověřenou zprávu pro objekt kanálu, který není klientem.

MQZAO_ALL_ADMIN
Schopnost nastavit kontext identity.

Všechna administrativní oprávnění jiná než MQZAO_CREATE.

Následující autorizace platí jak pro použití MQI, tak pro administraci správce front:

MQZAO_ALL
Všechna oprávnění, jiná než MQZAO_CREATE.

MQZAO_NONE
Žádná oprávnění.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_CHECK_AUTHORITY_2 (QMgrName, &EntityData, EntityType,
```

```
                        ObjectName, ObjectType, Authority, ComponentData,
                        &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Zkontrolujte, zda je uživatel privilegovaný

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_6 a je vyvolána správcem front s cílem určit, zda je určený uživatel privilegovaným uživatelem.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_CHECK_PRIVILEGED.

Syntaxe

MQZ_CHECK_PRIVILEGED(*QMgrName* , *EntityData* , *EntityType* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Údaje týkající se entity, která má být zkontrolována. Další informace viz “MQZED-Deskriptor entit” na stránce 1672.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený volbou EntityData. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ComponentData
Typ: MQBYTExComponentDataLength -vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

Pokud volání komponenty selže (tzn. *CompCode* vrátí MQCC_FAILED) a parametr *Continuation* je MQZCI_DEFAULT nebo MQZCI_CONTINUE, správce front bude i nadále volat další komponenty, pokud existují.

Pokud je volání úspěšné (tj. *CompCode* vrací MQCC_OK), nejsou volány žádné další komponenty bez ohledu na nastavení volby *Continuation* .

Pokud volání selže a parametr *Continuation* je MQZCI_STOP, nejsou volány žádné další komponenty a chyba je vrácena správci front. Komponenty neznají předchozí volání, takže parametr *Continuation* je před voláním vždy nastaven na hodnotu MQZCI_DEFAULT.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_PRIVILEGED
(2584, X'A18') Tento uživatel není ID oprávněného uživatele.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_CHECK_PRIVILEGED (QMgrName, &EntityData, EntityType,
                      ComponentData, &Continuation,
                      &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Zkopírovat všechna oprávnění

Tuto funkci poskytuje komponenta služby autorizace. Správce front spustí kopírování všech autorizací, které jsou aktuálně platné pro objekt odkazu, do jiného objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_COPY_ALL_AUTHORITY.

Syntaxe

MQZ_COPY_ALL_AUTHORITY(*QMgrName* , *RefObjectName* , *ObjectName* , *ObjectType* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

Název RefObject
Typ: MQCHAR48 -vstup

Název referenčního objektu. Název referenčního objektu, oprávnění, pro která se mají kopírovat. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, pro který mají být nastaveny přístupy. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený pomocí *RefObjectName* a *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

ComponentData
Typ: MQBYTExComponentDataLength -vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru délky ComponentDatavolání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_REF_OBJECT
(2294, X'8F6') Objekt odkazu je neznámý.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName, ObjectType,
                        ComponentData, &Continuation, &CompCode,
                        &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Oprávnění k odstranění

Tato funkce je poskytována komponentou služby autorizace a je spuštěna správcem front za účelem odstranění všech autorizací přidružených k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_DELETE_AUTHORITY.

Syntaxe

MQZ_DELETE_AUTHORITY(*QMgrName* , *ObjectName* , *ObjectType* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, pro který se mají odstranit přístupy. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru délky ComponentDatavolání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_DELETE_AUTHORITY (QMgrName, ObjectName, ObjectType, ComponentData,
                      &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Výčet dat oprávnění

Tuto funkci poskytuje komponenta autorizační služby MQZAS_VERSION_4 a správce front ji opakovaně spouští, aby načetl všechna data oprávnění, která odpovídají kritériím výběru zadaným při prvním vyvolání.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_ENUMERATE_AUTHORITY_DATA.

Syntaxe

MQZ_ENUMERATE_AUTHORITY_DATA(*QMgrName* , *StartEnumeration* , *Filter* , *AuthorityBufferLength* , *AuthorityBuffer* , *AuthorityDataLength* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

StartEnumeration
Typ: MQLONG-vstup

Příznak označující, zda může volání spustit výčet. Označuje, zda může volání spustit výčet dat oprávnění, nebo pokračovat ve výčtu dat oprávnění spuštěného předchozím voláním MQZ_ENUMERATE_AUTHORITY_DATA. Hodnota je jedna z následujících hodnot:

MQZSE_START
Počáteční výčet. Volání se spustí s touto hodnotou, aby se spustil výčet dat oprávnění. Parametr **Filter** uvádí kritéria výběru, která se mají použít k výběru dat oprávnění vrácených tímto a následnými voláními.

MQZSE_CONTINUE
Pokračovat ve výčtu. Volání se spustí s touto hodnotou, aby pokračovalo ve výčtu dat oprávnění. Parametr **Filter** je v tomto případě ignorován a lze jej zadat jako ukazatel Null (kritéria výběru jsou určena parametrem **Filter** určeným voláním, které mělo hodnotu *StartEnumeration* nastavenou na MQZSE_START).

Filtr
Typ: MQZAD-vstup

Filtr. Je-li *StartEnumeration* MQZSE_START, *Filter* uvádí kritéria výběru, která se mají použít k výběru dat oprávnění, která se mají vrátit. Pokud je *Filter* ukazatel null, nejsou použita žádná kritéria výběru, to znamená, že jsou vrácena všechna data oprávnění. Podrobnosti o kritériích výběru, která lze použít, naleznete v části "MQZAD-Data oprávnění" na stránce 1669 .

Má-li parametr *StartEnumeration* hodnotu MQZSE_CONTINUE, je parametr *Filter* ignorován a lze jej zadat jako ukazatel Null.

AuthorityBufferDélka
Typ: MQLONG-vstup

Délka řetězce *AuthorityBuffer*. Jedná se o délku parametru **AuthorityBuffer** v bajtech. Vyrovnávací paměť oprávnění musí být dostatečně velká, aby pojmula data, která mají být vrácena.

AuthorityBuffer
Typ: MQZAD-výstup

Data oprávnění. Jedná se o vyrovnávací paměť, ve které jsou vrácena data oprávnění. Vyrovnávací paměť musí být dostatečně velká, aby pojmula strukturu MQZAD, strukturu MQZED, plus nejdelší definovaný název entity a nejdelší název domény.

Poznámka: Poznámka: Tento parametr je definován jako MQZAD, protože MQZAD se vždy vyskytuje na začátku vyrovnávací paměti. Pokud je však vyrovnávací paměť deklarována jako MQZAD, bude vyrovnávací paměť příliš malá-musí být větší než MQZAD, aby mohla pojmout MQZAD, MQZED a názvy entit a domén.

AuthorityDataDélka
Typ: MQLONG-výstup

Délka dat vrácených v souboru *AuthorityBuffer*. Pokud je vyrovnávací paměť oprávnění příliš malá, je parametr *AuthorityDataLength* nastaven na požadovanou délku vyrovnávací paměti a volání vrátí kód dokončení MQCC_FAILED a kód příčiny MQRC_BUFFER_LENGTH_ERROR.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru délky ComponentDatavolání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_ENUMERATE_AUTHORITY_DATA to má stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_NO_DATA_AVAILABLE
(2379, X'94B') Data nejsou k dispozici.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration, &Filter,
                              AuthorityBufferLength,
                              &AuthorityBuffer,
                              &AuthorityDataLength, ComponentData,
                              &Continuation, &CompCode,
                              &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Volný uživatel

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_5 a je spuštěna správcem front za účelem uvolnění přidruženého přiděleného prostředku.

Spustí se po dokončení spuštění aplikace ve všech uživatelských kontextech, například během volání MQDISC MQI.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_FREE_USER.

Syntaxe

MQZ_FREE_USER(*QMgrName* , *FreeParms* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

FreeParms
Typ: MQZFP-vstup

Volné parametry. Struktura obsahující data týkající se prostředku, který má být uvolněn. Podrobnosti viz “MQZFP-Volné parametry” na stránce 1674.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru délky ComponentDatavolání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Příznak pokračování. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na jiných komponentách.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
```

```
                    IdentityContext, CorrelationPtr, ComponentData,
                    &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Získat oprávnění

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_1 a je spuštěna správcem front za účelem načtení oprávnění, které má entita pro přístup k určenému objektu, včetně oprávnění (pokud je entitou činitel), která jsou vlastněna skupinami, jejichž je činitel členem. Oprávnění z generických profilů jsou zahrnuta do vrácené sady oprávnění.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_GET_AUTHORITY.

Syntaxe

MQZ_GET_AUTHORITY(*QMgrName* , *EntityName* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityName
Typ: MQCHAR12 -vstup

Název entity. Název entity, jejíž přístup k objektu má být načten. Maximální délka řetězce je 12 znaků; je-li kratší, je zprava vyplněn mezerami. Název není ukončen znakem null.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityName*. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, ke kterému se má získat přístup. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Oprávnění subjektu. Pokud má entita jedno oprávnění, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, toto pole je bitové OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_GET_AUTHORITY to má stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_GET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, &Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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 -Získat oprávnění (rozšířené)

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_2 a je spuštěna správcem front za účelem načtení oprávnění, které má entita pro přístup k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_GET_AUTHORITY.

MQZ_GET_AUTHORITY_2 je jako MQZ_GET_AUTHORITY, ale s parametrem **EntityName** nahrazeným parametrem **EntityData** .

Syntaxe

MQZ_GET_AUTHORITY_2(*QMgrName* , *EntityData* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity, pro kterou má být načtena autorizace k objektu. Podrobnosti viz “MQZED-Deskriptor entit” na stránce 1672.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityData*. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, pro který se má načíst oprávnění entity. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Oprávnění subjektu. Pokud má entita jedno oprávnění, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, toto pole je bitové OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_GET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                     ObjectType, &Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-získat explicitní oprávnění

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_1 a je spuštěna správcem front za účelem načtení oprávnění, které má entita pro přístup k určenému objektu, včetně oprávnění (pokud je entitou činitel), která jsou vlastněna skupinami, jejichž je činitel členem. Oprávnění z generických profilů jsou zahrnuta do vrácené sady oprávnění.

V systému AIX and Linuxje pro vestavěného správce OAM (object authority manager) systému IBM MQ vrácené oprávnění, které je vlastněné pouze primární skupinou činitele.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_GET_EXPLICIT_AUTHORITY.

Syntaxe

MQZ_GET_EXPLICIT_AUTHORITY(*QMgrName* , *EntityName* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityName
Typ: MQCHAR12 -vstup

Název entity. Název entity, pro kterou se má načíst přístup k objektu. Maximální délka řetězce je 12 znaků; je-li kratší, je zprava vyplněn mezerami. Název není ukončen znakem null.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityName*. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, pro který se má načíst oprávnění entity. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Oprávnění subjektu. Pokud má entita jedno oprávnění, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, toto pole je bitové OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_GET_AUTHORITY to má stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,
                            ObjectName, ObjectType, &Authority,
                            ComponentData, &Continuation,
                            &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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 -Získat explicitní oprávnění (rozšířené)

Tuto funkci poskytuje komponenta služby autorizace MQZAS_VERSION_2 a spouští ji správce front za účelem načtení oprávnění, které má pojmenovaná skupina pro přístup k určenému objektu (ale bez

dalšího oprávnění skupiny **nikdo**), nebo oprávnění, které má primární skupina pojmenovaného činitele pro přístup k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_GET_EXPLICIT_AUTHORITY.

MQZ_GET_EXPLICIT_AUTHORITY_2 je jako MQZ_GET_EXPLICIT_AUTHORITY, ale s parametrem **EntityName** nahrazeným parametrem **EntityData** .

Syntaxe

MQZ_GET_EXPLICIT_AUTHORITY_2(*QMgrName* , *EntityData* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity, jejíž oprávnění k objektu má být načteno. Podrobnosti viz “MQZED-Deskriptor entit” na stránce 1672.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityData*. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, pro který se má načíst oprávnění entity. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Oprávnění subjektu. Pokud má entita jedno oprávnění, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, toto pole je bitové OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_GET_EXPLICIT_AUTHORITY_2 (QMgrName, &EntityData, EntityType,
                              ObjectName, ObjectType, &Authority,
                              ComponentData, &Continuation,
                              &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Inicializace služby autorizace

Tuto funkci poskytuje komponenta služby autorizace a spouští ji správce front během konfigurace komponenty. Očekává se volání MQZEP za účelem poskytnutí informací správci front.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_INIT_AUTHORITY.

Syntaxe

MQZ_INIT_AUTHORITY(*Hconfig* , *Options* , *QMgrName* , *ComponentDataLength* , *ComponentData* , *Version* , *CompCode* , *Reason*)

Parametry

Hconfig
Typ: MQHCONFIG-vstup

Popisovač konfigurace. Tento popisovač představuje konkrétní inicializovanou komponentu. Komponenta ji použije při volání správce front pomocí funkce MQZEP.

Volby
Typ: MQLONG-vstup

Volby inicializace. Musí to být jedna z následujících hodnot:

MQZIO_PRIMARY
Primární inicializace.

MQZIO_SECONDARY
Sekundární inicializace.

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

ComponentDataDélka
Typ: MQLONG-vstup

Délka dat komponenty. Délka oblasti *ComponentData* v bajtech. Tato délka je definována v konfiguračních datech komponenty.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Toto je inicializováno na všechny nuly před voláním primární inicializační funkce komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí (včetně inicializační funkce) poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Verze
Typ: MQLONG-vstup/výstup

Číslo verze. Na vstupu do inicializační funkce identifikuje nejvyšší číslo verze, kterou správce front podporuje. Funkce inicializace musí v případě potřeby tuto hodnotu změnit na verzi rozhraní, které podporuje. Pokud správce front při návratu nepodporuje verzi vrácenou komponentou, volá funkci MQZ_TERM_AUTHORITY komponenty a již tuto komponentu nepoužívá.

Podporovány jsou následující hodnoty:

MQZAS_VERSION_1
Verze 1.

MQZAS_VERSION_2
Verze 2.

MQZAS_VERSION_3
Verze 3.

MQZAS_VERSION_4
Verze 4.

MQZAS_VERSION_5
Verze 5.

MQZAS_VERSION_6
IBM WebSphere MQ 6.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_INITIALIZATION_FAILED
(2286, X'8EE') Inicializace se nezdařila z nedefinované příčiny.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,
                    ComponentData, &Version, &CompCode,
                    &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Služba autorizace dotazování

Tuto funkci poskytuje komponenta autorizační služby MQZAS_VERSION_5 a je spuštěna správcem front za účelem dotazování na podporované funkce.

Pokud se používá více komponent služby, volají se komponenty služby v opačném pořadí, než v jakém byly nainstalovány.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_INQUIRE.

Syntaxe

MQZ_INQUIRE(*QMgrName* , *SelectorCount* , *Selectors* , *IntAttrCount* , *IntAttrs* , *CharAttrLength* , *CharAttrs* , *SelectorReturned* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

SelectorCount
Typ: MQLONG-vstup

Počet selektorů. Počet selektorů dodaných v parametru **Selectors** .

Hodnota musí být v rozsahu 0 až 256.

Selektory.
Typ: MQLONGxSelectorPočet-vstup

Pole selektorů. Každý selektor identifikuje požadovaný atribut a musí být jedním z následujících:

- MQIACF_INTERFACE_VERSION (celé číslo)
- MQIACF_USER_ID_SUPPORT (celé číslo)
- MQCACF_SERVICE_COMPONENT (znak)

Selektory lze zadat v libovolném pořadí. Počet selektorů v poli je označen parametrem **SelectorCount** .

Celočíselné atributy identifikované selektory jsou vráceny v parametru **IntAttrs** ve stejném pořadí, v jakém jsou uvedeny v souboru *Selectors*.

Znakové atributy identifikované selektory jsou vráceny v parametru **CharAttrs** ve stejném pořadí, v jakém se objevují *Selectors*.

IntAttrCount
Typ: MQLONG-vstup

Počet celočíselných atributů zadaných v parametru IntAttrs .

Hodnota musí být v rozsahu 0 až 256.

IntAttrs
Typ: MQLONG x IntAttrPočet-výstup

Celočíselné atributy. Pole celočíselných atributů. Celočíselné atributy jsou vráceny ve stejném pořadí jako odpovídající celočíselné selektory v poli *Selectors* .

Počet CharAttr
Typ: MQLONG-vstup

Délka vyrovnávací paměti atributů znaků. Délka parametru **CharAttrs** v bajtech.

Hodnota musí být alespoň součtem délek požadovaných znakových atributů. Pokud nejsou požadovány žádné znakové atributy, nula je platná hodnota.

CharAttrs
Typ: MQLONG x CharAttrPočet-výstup

Vyrovnávací paměť atributů znaků. Vyrovnávací paměť obsahující atributy znaků, zřetězená dohromady. Atributy znaků jsou vráceny ve stejném pořadí jako odpovídající selektory znaků v poli *Selectors* .

Délka vyrovnávací paměti je dána parametrem počtu CharAttr.

SelectorReturned
Typ: MQLONG x SelectorCount -vstup

Byl vrácen selektor. Pole hodnot identifikujících, které atributy byly vráceny ze sady požadované selektory v parametru Selektory. Počet hodnot v tomto poli je označen parametrem **SelectorCount** . Každá hodnota v poli se vztahuje k selektoru z odpovídající pozice v poli Selektory. Každá hodnota je jedna z následujících:

MQZSL_VRÁCENO
Byl vrácen atribut požadovaný odpovídajícím selektorem v parametru **Selectors** .

MQZSL_NOT_VRÁCENO
Atribut požadovaný odpovídajícím selektorem v parametru **Selectors** nebyl vrácen.

Pole je inicializováno se všemi hodnotami jako *MQZSL_NOT_RETURNED*. Když komponenta služby autorizace vrátí atribut, nastaví odpovídající hodnotu v poli na *MQZSL_NOT_RETURNED*. To umožňuje ostatním komponentám služby autorizace, ke kterým je provedeno volání inquire, identifikovat, které atributy již byly vráceny.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
Nedostatek prostoru pro znakové atributy.

MQRC_INT_COUNT_TOO_SMALL
Nedostatek prostoru pro celočíselné atributy.

Je-li *CompCode* MQCC_FAILED:

MQRC_SELECTOR_COUNT_ERROR
Počet selektorů není platný.

MQRC_SELECTOR_ERROR
Selektor atributů není platný.

MQRC_SELECTOR_LIMIT_PŘEKROČENO
Bylo zadáno příliš mnoho selektorů.

MQRC_INT_ATTR_COUNT_ERROR
Počet celočíselných atributů není platný.

MQRC_INT_ATTRS_ARRAY_ERROR
Pole celočíselných atributů není platné.

MQRC_CHAR_ATTR_LENGTH_ERROR
Počet znakových atributů není platný.

MQRC_CHAR_ATTRS_ERROR
Řetězec znakových atributů není platný.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_INQUIRE (QMgrName, SelectorCount, Selectors, IntAttrCount,
                  &IntAttrs, CharAttrLength, &CharAttrs,
                  SelectorReturned, ComponentData, &Continuation,
                  &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Obnovit všechny autorizace

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_3 a je vyvolána správcem front za účelem aktualizace seznamu autorizací, které jsou interně uchovávány komponentou.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_REFRESH_CACHE (8L).

Syntaxe

MQZ_REFRESH_CACHE(*QMgrName* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY to má stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_WARNING:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Vyvolání jazyka C

```
MQZ_REFRESH_CACHE (QMgrName, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
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-Nastavit oprávnění

Tuto funkci poskytuje komponenta služby autorizace MQZAS_VERSION_1 a spouští ji správce front, aby nastavil oprávnění, které má entita pro přístup k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_SET_AUTHORITY.

Poznámka: Tato funkce přepíše všechna existující oprávnění. Chcete-li zachovat existující oprávnění, musíte je znovu nastavit pomocí této funkce.

Syntaxe

```
MQZ_SET_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName ,
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )
```

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityName
Typ: MQCHAR12 -vstup

Název entity. Název entity, pro kterou se má načíst přístup k objektu. Maximální délka řetězce je 12 znaků; je-li kratší, je zprava vyplněn mezerami. Název není ukončen znakem null.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityName*. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, ke kterému je požadován přístup. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Oprávnění subjektu. Pokud je nastaveno jedno oprávnění, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Pokud je nastaveno více než jedno oprávnění, toto pole je bitové OR odpovídajících konstant MQZAO_ *.

ComponentDatarname>
Typ: MQBYTExComponentDataLength -vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_GET_AUTHORITY to má stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_SET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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 -Nastavit oprávnění (rozšířené)

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_2 a je spuštěna správcem front za účelem nastavení oprávnění, které má entita pro přístup k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_SET_AUTHORITY.

Poznámka: Tato funkce přepíše všechna existující oprávnění. Chcete-li zachovat existující oprávnění, musíte je znovu nastavit pomocí této funkce.

MQZ_SET_AUTHORITY_2 je jako MQZ_SET_AUTHORITY, ale s parametrem **EntityName** nahrazeným parametrem **EntityData** .

Syntaxe

MQZ_SET_AUTHORITY_2(*QMgrName* , *EntityData* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity, jejíž oprávnění k objektu má být nastaveno. Podrobnosti viz “MQZED-Deskriptor entit” na stránce 1672.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityData*. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -vstup

Název objektu. Název objektu, pro který má být nastaveno oprávnění entity. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí to být jedna z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Oprávnění subjektu. Pokud je nastaveno jedno oprávnění, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Pokud je nastaveno více než jedno oprávnění, toto pole je bitové OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_SET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                     ObjectType, Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Ukončete službu autorizace

Tato funkce je poskytována komponentou služby autorizace a je spuštěna správcem front, pokud již nevyžaduje služby této komponenty. Funkce musí provést vyčištění požadované komponentou.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_TERM_AUTHORITY.

Syntaxe

MQZ_TERM_AUTHORITY(*Hconfig* , *Options* , *QMgrName* , *ComponentData* , *CompCode* , *Reason*)

Parametry

Hconfig
Typ: MQHCONFIG-vstup

Popisovač konfigurace. Tento popisovač představuje konkrétní ukončovanou komponentu. Komponenta ji použije při volání správce front pomocí funkce MQZEP.

Volby
Typ: MQLONG-vstup

Volby ukončení. Musí to být jedna z následujících hodnot:

MQZTO_PRIMARY
Primární ukončení.

MQZTO_SECONDARY
Sekundární ukončení.

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru délky ComponentDatave volání MQZ_INIT_AUTHORITY.

Po dokončení volání MQZ_TERM_AUTHORITY správce front tato data zruší.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_TERMINATION_FAILED
(2287, X'8FF') Ukončení selhalo z nedefinované příčiny.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_TERM_AUTHORITY (Hconfig, Options, QMgrName, ComponentData,
                    &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Název odstranění

Tato funkce je poskytována komponentou služby názvů a je spuštěna správcem front za účelem odstranění položky pro určenou frontu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_DELETE_NAME.

Syntaxe

MQZ_DELETE_NAME(*QMgrName* , *QName* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

QName
Typ: MQCHAR48 -vstup

Název fronty. Název fronty, pro kterou má být položka odstraněna. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru délky ComponentDatave volání MQZ_INIT_NAME.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Musí to být jedna z následujících hodnot:

VÝCHOZÍ
Pokračování závislé na správci front.

MQZCI_STOP
Nepokračujte s další komponentou.

V případě příkazu **MQZ_DELETE_NAME** se správce front nepokusí spustit jinou komponentu bez ohledu na to, co je vráceno v parametru **Continuation** .

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Varování (částečné dokončení).

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_WARNING:

MQRC_UNKNOWN_NAME
(2288, X'8F0') Název fronty nebyl nalezen.

Poznámka: Vrácení tohoto kódu nemusí být možné, pokud základní služba v tomto případě úspěšně odpoví.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_DELETE_NAME (QMgrName, QName, ComponentData, &Continuation,
                 &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Inicializace služby názvů

Tato funkce je poskytována komponentou služby názvů a je spuštěna správcem front během konfigurace komponenty. Očekává se volání MQZEP za účelem poskytnutí informací správci front.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_INIT_NAME.

Syntaxe

MQZ_INIT_NAME(*Hconfig* , *Options* , *QMgrName* , *ComponentDataLength* , *ComponentData* , *Version* , *CompCode* , *Reason*)

Parametry

Hconfig
Typ: MQHCONFIG-vstup

Popisovač konfigurace. Tento popisovač představuje konkrétní inicializovanou komponentu. Komponenta ji použije při volání správce front pomocí funkce MQZEP.

Volby
Typ: MQLONG-vstup

Volby inicializace. Musí to být jedna z následujících hodnot:

MQZIO_PRIMARY
Primární inicializace.

MQZIO_SECONDARY
Sekundární inicializace.

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

ComponentDataDélka
Typ: MQLONG-vstup

Délka dat komponenty. Délka oblasti *ComponentData* v bajtech. Tato délka je definována v konfiguračních datech komponenty.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Toto je inicializováno na všechny nuly před voláním primární inicializační funkce komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí (včetně inicializační funkce) poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Verze
Typ: MQLONG-vstup/výstup

Číslo verze. Na vstupu do inicializační funkce identifikuje nejvyšší číslo verze, kterou správce front podporuje. Funkce inicializace musí v případě potřeby tuto hodnotu změnit na verzi rozhraní, které

podporuje. Pokud správce front při vrácení nepodporuje verzi vrácenou komponentou, volá funkci MQZ_TERM_NAME komponenty a tuto komponentu již dále nepoužívá.

Podporovány jsou následující hodnoty:

MQZAS_VERSION_1
Verze 1.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_INITIALIZATION_FAILED
(2286, X'8EE') Inicializace se nezdařila z nedefinované příčiny.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_INIT_NAME (Hconfig, Options, QMgrName, ComponentDataLength,
               ComponentData, &Version, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Vložit název

Tato funkce je poskytována komponentou služby názvů a je spuštěna správcem front za účelem vložení položky pro určenou frontu obsahující název správce front, který tuto frontu vlastní. Pokud je fronta již ve službě definována, volání se nezdaří.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_INSERT_NAME.

Syntaxe

MQZ_INSERT_NAME(*QMgrName* , *QName* , *ResolvedQMgrName* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

QName
Typ: MQCHAR48 -vstup

Název fronty. Název fronty, pro kterou má být vložena položka. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název ResolvedQMgr
Typ: MQCHAR48 -vstup

Vyřešený název správce front. Název správce front, na kterého se fronta interpretuje. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí (včetně inicializační funkce) poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_NAME.

Pokračování
Typ: MQLONG-vstup/výstup

Indikátor pokračování nastavený podle komponenty. V případě MQZ_INSERT_NAME se správce front nepokusí spustit jinou komponentu, což je vráceno v parametru **Continuation** .

Podporovány jsou následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_Q_ALREADY_EXISTS
(2290, X'8F2') Objekt fronty již existuje.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_INSERT_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,
                 &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Název vyhledávání

Tato funkce je poskytována komponentou služby názvů a je spuštěna správcem front za účelem načtení názvu vlastnícího správce front pro určenou frontu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_LOOKUP_NAME.

Syntaxe

MQZ_LOOKUP_NAME(*QMgrName* , *QName* , *ResolvedQMgrName* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

QName
Typ: MQCHAR48 -vstup

Název fronty. Název fronty, pro kterou má být položka rozlišena. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název ResolvedQMgr
Typ: MQCHAR48 -výstup

Vyřešený název správce front. Pokud je funkce úspěšně dokončena, jedná se o název správce front, který je vlastníkem fronty.

Název vrácený komponentou služby musí být vpravo doplněn mezerami na celou délku parametru; název nesmí být ukončen znakem null nebo obsahovat úvodní nebo vložené mezery.

ComponentData
Typ: MQBYTExComponentDataLength -vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí (včetně inicializační funkce) poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_NAME.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Pro parametr MQZ_LOOKUP_NAME určuje správce front, zda má být spuštěna jiná komponenta služby názvů, a to následujícím způsobem:

- Má-li parametr *CompCode* hodnotu MQCC_OK, nejsou spuštěny žádné další komponenty, bez ohledu na hodnotu vrácenou v poli *Pokračování*.
- Pokud *CompCode* není MQCC_OK, spustí se další komponenta, pokud *Continuation* není MQZCI_STOP.

Podporovány jsou následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_Q_NAME
(2288, X'8F0') Název fronty nebyl nalezen.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_LOOKUP_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,
                 &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Ukončit službu názvů

Tato funkce je poskytována komponentou služby názvů a je spuštěna správcem front, pokud již nevyžaduje služby této komponenty. Funkce musí provést vyčištění požadované komponentou.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_TERM_NAME.

Syntaxe

MQZ_TERM_NAME(*Hconfig* , *Options* , *QMgrName* , *ComponentData* , *CompCode* , *Reason*)

Parametry

Hconfig
Typ: MQHCONFIG-vstup

Popisovač konfigurace. Tento popisovač představuje konkrétní ukončovanou komponentu. Používá ji komponenta při volání správce front s funkcí MQZEP.

Volby
Typ: MQLONG-vstup

Volby ukončení. Musí to být jedna z následujících hodnot:

MQZTO_PRIMARY
Primární ukončení.

MQZTO_SECONDARY
Sekundární ukončení.

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí (včetně inicializační funkce) poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Data komponenty jsou ve sdílené paměti přístupné pro všechny procesy.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_NAME.

Po dokončení volání MQZ_TERM_NAME správce front tato data vyřadí.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_TERMINATION_FAILED
(2287, X'8FF') Ukončení selhalo z nedefinované příčiny.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_TERM_NAME (Hconfig, Options, QMgrName, ComponentData, &CompCode,
               &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-kontext aplikace

Struktura MQZAC se používá ve volání MQZ_AUTHENTICATE_USER pro parametr *ApplicationContext* . Tento parametr určuje data související s volající aplikací.

Tabulka 1 shrnuje pole ve struktuře.

Tabulka 838. Pole v MQZAC

Pole	Popis
StrucId	Identifikátor struktury
verze	Číslo verze struktury
ProcessId	Identifikátor procesu
ThreadId	Identifikátor podprocesu

Tabulka 838. Pole v MQZAC (pokračování)	
Pole	**Popis**
ApplName	Název aplikace
UserID	Identifikátor uživatele
EffectiveUser	Efektivní identifikátor uživatele
Prostředí	Prostředí
CallerType	Typ volajícího
AuthenticationType	Typ ověřování
BindType	Typ vazby

Pole

StrucId
Typ: MQCHAR4 -vstup

Identifikátor struktury. Hodnota je následující:

MQZAC_STRUC_ID
Identifikátor pro strukturu kontextu aplikace.

Pro programovací jazyk C je definována také konstanta MQZAC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZAC_STRUC_ID, ale je to pole znaků místo řetězce.

Verze
Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQZAC_VERSION_1
Struktura kontextu aplikace Version-1 . Konstanta MQZAC_CURRENT_VERSION určuje číslo verze aktuální verze.

ProcessId
Typ: MQPID-vstup

Identifikátor procesu aplikace.

ThreadId
Typ: MQTID-vstup

Identifikátor podprocesu aplikace.

ApplName
Typ: MQCHAR28 -vstup

Název aplikace.

UserID
Typ: MQCHAR12 -vstup

Identifikátor uživatele. V systému AIX and Linux toto pole uvádí skutečné ID uživatele aplikace. V systému Windows toto pole určuje ID uživatele aplikace.

ID EffectiveUser
Typ: MQCHAR12 -vstup

Efektivní identifikátor uživatele. V systému AIX and Linux toto pole určuje efektivní ID uživatele aplikace. V systému Windows je toto pole prázdné.

Prostředí
Typ: MQLONG-vstup

Prostředí. Toto pole určuje prostředí, ze kterého bylo volání provedeno. Pole má jednu z následujících hodnot:

MQXE_COMMAND_SERVER
Příkazový server

MQXE_MQSC
Interpret příkazů **runmqsc**

MQXE_MCA
Agent kanálu zpráv MQXE_OTHER

MQXE_OTHER
Nedefinované prostředí

CallerType
Typ: MQLONG-vstup

Typ volajícího. Toto pole uvádí typ programu, který provedl volání. Pole má jednu z následujících hodnot:

MQXACT_EXTERNÍ
Volání je externí vzhledem ke správci front.

MQXACT_INTERNÍ
Volání je interní pro správce front.

AuthenticationType
Typ: MQLONG-vstup

Typ ověřování. Toto pole určuje typ prováděného ověřování. Pole má jednu z následujících hodnot:

MQZAT_INITIAL_CONTEXT
Volání ověření je způsobeno inicializací kontextu uživatele. Tato hodnota se používá během volání MQCONN nebo MQCONNX.

MQZAT_CHANGE_CONTEXT
Volání ověření je způsobeno změnou kontextu uživatele. Tato hodnota se používá, když agent MCA změní kontext uživatele. Nadřazené téma: MQZAC-

BindType
Typ: MQLONG-vstup

Typ vazby. Toto pole určuje typ používané vazby. Pole má jednu z následujících hodnot:

MQCNO_FASTPATH_BINDING
Rychlospojka.

MQCNO_SHARED_BINDING
Sdílená vazba.

MQCNO_ISOLATED_BINDING
Izolovaná vazba.

C prohlášení

Deklarujte pole struktury takto:

```
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-Data oprávnění

Struktura MQZAD se používá ve volání MQZ_ENUMERATE_AUTHORITY_DATA pro dva parametry, jeden vstup a jeden výstup.

Další informace o parametrech **Filter** a **AuthorityBuffer**
viz “MQZ_ENUMERATE_AUTHORITY_DATA-Výčet dat oprávnění” na stránce 1631 :

- MQZAD se používá pro parametr **Filter** , který je vstupem pro volání. Tento parametr uvádí kritéria výběru, která se mají použít pro výběr dat oprávnění vrácených voláním.
- MQZAD se také používá pro parametr **AuthorityBuffer** , který je výstupem volání. Tento parametr určuje oprávnění pro jednu kombinaci názvu profilu, typu objektu a entity.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 839. Pole v MQZAD

Pole	Popis
StrucId	Identifikátor struktury
verze	Číslo verze struktury
ProfileName	Název profilu
ObjectType	Typ objektu
Oprávnění	Oprávnění
EntityDataPtr	Ukazatel na data entity
EntityType	Typ entity
Volby	Volby

Pole

StrucId
Typ: MQCHAR4 -vstup

Identifikátor struktury. Hodnota je následující:

MQZAD_STRUC_ID
Identifikátor pro datovou strukturu oprávnění.

Pro programovací jazyk C je definována také konstanta MQZAD_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZAD_STRUC_ID, ale je to pole znaků místo řetězce.

Verze
Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQZAD_VERSION_1
Struktura kontextu aplikace Version-1 . Konstanta MQZAD_CURRENT_VERSION určuje číslo verze aktuální verze.

Následující konstanta určuje číslo verze aktuální verze:

MQZAD_CURRENT_VERSION
Aktuální verze datové struktury oprávnění.

ProfileName
Typ: MQCHAR48 -vstup

Název profilu.

Pro parametr **Filter** je toto pole název profilu, pro který jsou požadována data oprávnění. Pokud je název zcela prázdný až do konce pole nebo prvního znaku null, vrátí se data oprávnění pro všechny názvy profilů.

Pro parametr **AuthorityBuffer** je toto pole název profilu, který odpovídá zadaným kritériím výběru.

ObjectType
Typ: MQLONG-vstup

Typ objektu.

Pro parametr **Filter** je toto pole typ objektu, pro který jsou požadována data oprávnění. Pokud je hodnota MQOT_ALL, vrátí se data oprávnění pro všechny typy objektů.

Pro parametr **AuthorityBuffer** je toto pole typem objektu, na který se vztahuje profil identifikovaný parametrem **ProfileName** .

Hodnota je jedna z následujících; pro parametr **Filter** je platná také hodnota MQOT_ALL:

MQOT_AUTH_INFO
Ověřovací informace

MQOT_CHANNEL
Kanál

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta

MQOT_LISTENER
Modul listener

MQOT_NAMELIST
Seznam názvů

MQOT_PROCESS
Definice procesu

MQOT_Q
Fronta

MQOT_Q_MGR
Správce front

MQOT_SERVICE
Služba

Oprávnění
Typ: MQLONG-vstup

Oprávnění.

Pro parametr **Filter** je toto pole ignorováno.

Pro parametr **AuthorityBuffer** toto pole představuje autorizace, které má entita k objektům označeným jako **ProfileName** a **ObjectType**. Pokud má entita pouze jedno oprávnění, pole se rovná příslušné hodnotě autorizace (konstanta MQZAO_ *). Pokud má entita více než jedno oprávnění, pole je bitové OR odpovídajících konstant MQZAO_ *.

EntityDataPtr
Typ: PMQZED-vstup

Adresa struktury MQZED identifikující entitu.

Pro parametr **Filter** toto pole ukazuje na strukturu MQZED, která identifikuje entitu, pro kterou jsou vyžadována data oprávnění. Je-li **EntityDataPtr** ukazatel null, vrátí se data oprávnění pro všechny entity.

Pro parametr **AuthorityBuffer** toto pole ukazuje na strukturu MQZED, která identifikuje entitu, pro kterou byla vrácena data oprávnění.

EntityType
Typ: MQLONG-vstup

Typ entity.

Pro parametr **Filter** toto pole uvádí typ entity, pro který jsou požadována data oprávnění. Je-li hodnota MQZAET_NONE, vrátí se data oprávnění pro všechny typy entit.

Pro parametr **AuthorityBuffer** toto pole určuje typ entity identifikované strukturou MQZED, na kterou odkazuje parametr **EntityDataPtr** .

Hodnota je jedna z následujících; pro parametr **Filter** je platná také hodnota MQZAET_NONE:

MQZAET_PRINCIPAL
Hlavní

MQZAET_GROUP
Skupina

Volby
Typ: MQAUTHOPT-vstup

Volby. Toto pole uvádí volby, které dávají kontrolu nad zobrazenými profily. Musí být zadána jedna z následujících hodnot:

MQAUTHOPT_NAME_ALL_MATCHING
Zobrazí všechny profily

MQAUTHOPT_NAME_EXPLICIT
Zobrazí profily, které mají přesně stejný název, jaký je uveden v poli **ProfileName** .

Kromě toho musí být také specifikována jedna z následujících možností:

MQAUTHOPT_ENTITY_SET
Zobrazte všechny profily, které se používají k výpočtu kumulativního oprávnění, které má entita k objektu uvedenému v parametru **ProfileName** . Parametr **ProfileName** nesmí obsahovat žádné zástupné znaky.

- Pokud je určená entita činitelem, zobrazí se pro každého člena sady {entity, groups} nejpoužitelnější profil, který se vztahuje k objektu.
- Pokud je určenou entitou skupina, zobrazí se nejpoužitelnější profil ze skupiny, který se vztahuje k objektu.
- Je-li zadána tato hodnota, musí být všechny hodnoty **ProfileName**, **ObjectType**, **EntityType**a název entity určené ve struktuře **EntityDataPtr** MQZED neprázdné.

Pokud jste zadali MQAUTHOPT_NAME_ALL_MATCHING, můžete také zadat následující hodnotu:

MQAUTHOPT_ENTITY_EXPLICIT
Zobrazí profily, které mají přesně stejný název entity jako název entity určený ve struktuře **EntityDataPtr** MQZED.

C prohlášení

```
typedef struct tagMQZAD MQZAD;
struct tagMQZAD {
  MQCHAR4   StrucId;        /* Structure identifier */
  MQLONG    Version;        /* Structure version number */
  MQCHAR48  ProfileName;    /* Profile name */
  MQLONG    ObjectType;     /* Object type */
  MQLONG    Authority;      /* Authority */
  PMQZED    EntityDataPtr;  /* Address of MQZED structure identifying an
                               entity */
  MQLONG    EntityType;     /* Entity type */
  MQAUTHOPT Options;        /* Options */
};
```

MQZED-Deskriptor entit

Struktura MQZED se používá v řadě volání služby autorizace k určení entity, pro kterou se má autorizace zkontrolovat.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 840. Pole v MQZED	
Pole	**Popis**
StrucId	Identifikátor struktury
verze	Verze
EntityName Ptr	Název entity
EntityDomainPtr	Ukazatel domény entity
SecurityId	Identifikátor zabezpečení
CorrelationPtr	Korelační ukazatel

Pole

StrucId
Typ: MQCHAR4 -vstup

Identifikátor struktury. Hodnota je následující:

MQZED_STRUC_ID
Identifikátor struktury deskriptoru entity.

Pro programovací jazyk C je definována také konstanta MQZED_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZED_STRUC_ID, ale je to pole znaků místo řetězce.

Verze
Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQZED_VERSION_1
Version-1 struktura deskriptoru entity.

Následující konstanta určuje číslo verze aktuální verze:

MQZED_CURRENT_VERSION
Aktuální verze struktury deskriptoru entity.

EntityNamePtr
Typ: PMQCHAR-vstup

Název profilu.

Adresa názvu entity. Jedná se o ukazatel na název entity, jejíž autorizace se má zkontrolovat.

EntityDomainPtr
Typ: PMQCHAR-vstup

Adresa názvu domény entity. Jedná se o ukazatel na název domény obsahující definici entity, jejíž oprávnění se má zkontrolovat.

SecurityId
Typ: MQBYTE40 -vstup

Oprávnění.

Identifikátor zabezpečení. Jedná se o identifikátor zabezpečení, jehož autorizace se má zkontrolovat.

CorrelationPtr
Typ: MQPTR-vstup

Korelační ukazatel. To usnadňuje předávání korelačních dat mezi funkcí ověření uživatele a dalšími odpovídajícími funkcemi OAM.

C prohlášení

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  PMQCHAR   EntityNamePtr;    /* Address of entity name */
  PMQCHAR   EntityDomainPtr;  /* Address of entity domain name */
  MQBYTE40  SecurityId;       /* Security identifier */
  MQPTR     CorrelationPtr;   /* Address of correlation data */
```

MQZEP-Přidání vstupního bodu komponenty

Komponenta služby spustí tuto funkci během inicializace pro přidání vstupního bodu do vektoru vstupního bodu pro tuto komponentu služby.

Syntaxe

MQZEP (*Hconfig* , *Funkce* , *EntryPoint* , *CompCode* , *Příčina*)

Parametry

Hconfig
Typ: MQHCONFIG-vstup

Popisovač konfigurace. Tento popisovač představuje komponentu, která se konfiguruje pro tuto konkrétní instalovatelnou službu. Musí být stejná jako komponenta předaná funkci konfigurace komponenty správcem front při volání inicializace komponenty.

Funkce
Typ: MQLONG-vstup

Identifikátor funkce. Platné hodnoty pro tuto službu jsou definovány pro každou instalovatelnou službu.

Pokud je MQZEP volán pro stejnou funkci více než jednou, poslední provedené volání poskytne použitý vstupní bod.

EntryPoint
Typ: PMQFUNC-vstup

Vstupní bod funkce. Jedná se o adresu vstupního bodu poskytnutého komponentou k provedení funkce.

Hodnota NULL je platná a označuje, že funkce není touto komponentou poskytnuta. Pro vstupní body, které nejsou definovány pomocí MQZEP, se předpokládá hodnota NULL.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_FUNCTION_ERROR
(2281, X'8E9') Identifikátor funkce není platný.

MQRC_HCONFIG_ERROR
(2280, X'8E8') popisovač konfigurace není platný.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZEP (Hconfig, Function, EntryPoint, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONFIG  Hconfig;     /* Configuration handle */
MQLONG     Function;    /* Function identifier */
PMQFUNC    EntryPoint;  /* Function entry point */
MQLONG     CompCode;    /* Completion code */
MQLONG     Reason;      /* Reason code qualifying CompCode */
```

MQZFP-Volné parametry

Struktura MQZFP se používá ve volání MQZ_FREE_USER pro parametr *FreeParms* . Tento parametr uvádí data související s uvolněným prostředkem.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 841. Pole v MQZFP

Pole	Popis
StrucId	Identifikátor struktury
verze	Verze
Vyhrazeno	Vyhrazené pole
CorrelationPtr	Korelační ukazatel

Pole

StrucId
Typ: MQCHAR4 -vstup

Identifikátor struktury. Hodnota je následující:

MQZIC_STRUC_ID
Identifikátor pro strukturu kontextu identity. Pro programovací jazyk C je definována také konstanta MQZIC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZIC_STRUC_ID, ale je to pole znaků místo řetězce.

Verze
Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQZFP_VERSION_1
Version-1 volná struktura parametrů.

Následující konstanta určuje číslo verze aktuální verze:

MQZFP_CURRENT_VERSION
Aktuální verze struktury volných parametrů.

Vyhrazeno
Typ: MQBYTE8 -vstup

Vyhrazené pole. Počáteční hodnota je null.

CorrelationPtr
Typ: MQPTR-vstup

Korelační ukazatel. Adresa korelačních dat souvisejících s prostředkem, který má být uvolněn.

C prohlášení

```
typedef struct tagMQZFP MQZFP;
struct tagMQZFP {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  MQBYTE8   Reserved;         /* Reserved field */
  MQPTR     CorrelationPtr;   /* Address of correlation data */
};
```

MQZIC-Kontext identity

Struktura MQZIC se používá ve volání MQZ_AUTHENTICATE_USER pro parametr *IdentityContext* .

Struktura MQZIC obsahuje informace o kontextu identity, které identifikují uživatele aplikace, který poprvé vložil zprávu do fronty:

- Správce front vyplní pole *UserIdentifier* názvem, který identifikuje uživatele. Způsob, jakým to může správce front provést, závisí na prostředí, ve kterém je aplikace spuštěna.
- Správce front vyplní pole *AccountingToken* tokenem nebo číslem, které určuje aplikace, jež vložila zprávu.
- Aplikace mohou použít pole *ApplIdentityData* pro jakékoli další informace, které chtějí o uživateli zahrnout (například šifrované heslo).

Vhodně autorizované aplikace mohou nastavit kontext identity pomocí funkce MQZ_AUTHENTICATE_USER.

Identifikátor zabezpečení systému Windows (SID) je uložen v poli *AccountingToken* , když je zpráva vytvořena v adresáři IBM MQ for Windows. Identifikátor SID lze použít k doplnění pole *UserIdentifier* a k zavedení pověření uživatele.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 842. Pole v MQZIC

Pole	Popis
StrucId	Identifikátor struktury
verze	Verze
UserIdentifier	Identifikátor uživatele
AccountingToken	Token evidence
ApplIdentityData	Data identity aplikace

Pole

StrucId
Typ: MQCHAR4 -vstup

Identifikátor struktury. Hodnota je následující:

MQZIC_STRUC_ID
Identifikátor pro strukturu kontextu identity. Pro programovací jazyk C je definována také konstanta MQZIC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZIC_STRUC_ID, ale je to pole znaků místo řetězce.

Verze
Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQZIC_VERSION_1
Struktura kontextu identity Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

MQZIC_CURRENT_VERSION
Aktuální verze struktury kontextu identity.

UserIdentifier
Typ: MQCHAR12 -vstup

Identifikátor uživatele. Toto je část kontextu identity zprávy. *UserIdentifier* uvádí identifikátor uživatele aplikace, která je původcem zprávy. Správce front považuje tyto informace za znaková data, ale nedefinuje jejich formát. Další informace o poli *UserIdentifier* naleznete v tématu “UserIdentifier (MQCHAR12) pro MQMD” na stránce 459.

AccountingToken
Typ: MQBYTE32 -vstup

Účtovací token. Toto je část kontextu identity zprávy. *AccountingToken* umožňuje aplikaci způsobit, že práce vykonaná v důsledku zprávy bude řádně účtována. Správce front považuje tyto informace za řetězec bitů a nekontroluje jejich obsah. Další informace o poli *AccountingToken* viz “AccountingToken (MQBYTE32) pro MQMD” na stránce 461.

ApplIdentityData
Typ: MQCHAR32 -vstup

Údaje o aplikaci týkající se identity. Toto je část kontextu identity zprávy. ApplIdentityData jsou informace definované sadou aplikací, které lze použít k poskytnutí dalších informací o původu zprávy. Může být například nastaven aplikacemi, které jsou spuštěny s vhodným oprávněním uživatele, aby označily, zda jsou data identity důvěryhodná. Další informace o datovém poli ApplIdentityviz “ApplIdentity-data (MQCHAR32) pro MQMD” na stránce 462.

C prohlášení

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  MQCHAR12  UserIdentifier;   /* User identifier */
  MQBYTE32  AccountingToken;  /* Accounting token */
  MQCHAR32  ApplIdentityData; /* Application data relating to identity */
};
```

IBM i Referenční informace o rozhraní instalovatelných služeb na webu IBM i

Pomocí těchto informací pochopíte referenční informace pro instalovatelné služby pro produkt IBM i.

Pro každou funkci je uveden popis včetně identifikátoru funkce (pro MQZEP).

Parametry jsou uvedeny v pořadí, v jakém se musí vyskytnout. Všichni musí být přítomni.

Každý název parametru je následován datovým typem v závorkách. Jedná se o základní datové typy popsané v části “Základní datové typy” na stránce 993.

Vyvolání jazyka C je také poskytnuto po popisu parametrů.

Související pojmy

IBM i Instalovatelné služby a komponenty pro systém IBM i

ALW Instalovatelné služby a komponenty pro systémy UNIX, Linux a Windows

Související odkazy

“Referenční informace o rozhraní instalovatelných služeb” na stránce 1614
Tato kolekce témat poskytuje referenční informace pro instalovatelné služby.

IBM i MQZEP (Přidat vstupní bod komponenty) na IBM i

Tato funkce je vyvolána komponentou služby během inicializace pro přidání vstupního bodu do vektoru vstupního bodu pro tuto komponentu služby.

Syntaxe

```
MQZEP (Hconfig, Function, EntryPoint, CompCode, Reason)
```

Parametry

Volání MQZEP má následující parametry.

Hconfig (MQHCONFIG)-vstup
Popisovač konfigurace.

Tento popisovač představuje komponentu, která se konfiguruje pro tuto konkrétní instalovatelnou službu. Musí být stejný jako ten, který byl předán funkci konfigurace komponenty správcem front při volání inicializace komponenty.

Funkce (MQLONG)-vstup
Identifikátor funkce.

Platné hodnoty pro tuto službu jsou definovány pro každou instalovatelnou službu. Pokud je funkce MQZEP volána více než jednou pro stejnou funkci, poslední provedené volání poskytuje vstupní bod, který se používá.

EntryPoint (PMQFUNC)-vstup
Vstupní bod funkce.

Jedná se o adresu vstupního bodu poskytnutého komponentou k provedení funkce. Hodnota NULL je platná a označuje, že funkce není touto komponentou poskytnuta. NULL se předpokládá pro vstupní body, které nejsou definovány pomocí MQZEP.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_FUNCTION_ERROR
(2281, X'8E9') Identifikátor funkce není platný.

MQRC_HCONFIG_ERROR
(2280, X'8E8') popisovač konfigurace není platný.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZEP (Hconfig, Function, EntryPoint, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONFIG  Hconfig;     /* Configuration handle */
MQLONG     Function;    /* Function identifier */
PMQFUNC    EntryPoint;  /* Function entry point */
MQLONG     CompCode;    /* Completion code */
MQLONG     Reason;      /* Reason code qualifying CompCode */
```

IBM i MQHCONFIG (popisovač konfigurace) na systému IBM i

Datový typ MQHCONFIG představuje popisovač konfigurace, tj. komponentu, která se konfiguruje pro konkrétní instalovatelnou službu. Popisovač konfigurace musí být zarovnán na své přirozené hranici.

Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

C prohlášení

```
typedef void MQPOINTER MQHCONFIG;
```

IBM i PMQFUNC (ukazatel na funkci) na systému IBM i

Ukazatel na funkci.

C prohlášení

```
typedef void MQPOINTER PMQFUNC;
```

IBM i MQZ_AUTHENTICATE_USER (Ověřit uživatele) na IBM i

Tuto funkci poskytuje komponenta autorizační služby MQZAS_VERSION_5 . Je vyvolán správcem front k ověření uživatele nebo k nastavení polí kontextu identity.

Vyvolá se při vytvoření kontextu uživatelské aplikace IBM MQ . K tomu dochází během volání připojení v místě, kde je inicializován uživatelský kontext aplikace, a v každém bodě, kde je změněn uživatelský kontext aplikace. Při každém volání připojení jsou informace o uživatelském kontextu aplikace znovu získány v poli *IdentityContext* .

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_AUTHENTICATE_USER.

Syntaxe

MQZ_AUTHENTICATE_USER *(QMgrName, SecurityParms, ApplicationContext, IdentityContext, CorrelationPtr, ComponentData, Continuation, CompCode, Reason)*

Parametry

Volání MQZ_AUTHENTICATE_USER má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null. Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

SecurityParms (MQCSP)-vstup
Parametry zabezpečení.

Data týkající se ID uživatele, hesla a typu ověření.

Během volání MQCONN MQI tento parametr obsahuje hodnotu null nebo výchozí hodnoty.

ApplicationContext (MQZAC)-vstup
Kontext aplikace.

Údaje týkající se volající aplikace. Podrobnosti viz “MQZAC (kontext aplikace) na systému IBM i” na stránce 1708. Během každého volání MQCONN nebo MQCONNX MQI jsou znovu získány informace o kontextu uživatele ve struktuře MQZAC.

IdentityContext (MQZIC)-vstup/výstup
Kontext identity.

Na vstupu do funkce ověření uživatele identifikuje aktuální kontext identity. Funkce ověření uživatele to může změnit, což znamená, že správce front převezme nový kontext identity. Další podrobnosti o struktuře MQZIC viz “MQZIC (kontext identity) na systému IBM i” na stránce 1714 .

CorrelationPtr (MQPTR)-výstup
Korelační ukazatel.

Určuje adresu všech korelačních dat. Tento ukazatel je poté předán dalším voláním OAM.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty. Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Příznak pokračování.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na jiných komponentách.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                   IdentityContext, &CorrelationPtr, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQCSP     SecurityParms;      /* Security parameters */
MQZAC     ApplicationContext; /* Application context */
MQZIC     IdentityContext;    /* Identity context */
MQPTR     CorrelationPtr;     /* Correlation pointer */
MQBYTE    ComponentData[n];   /* Component data */
MQLONG    Continuation;       /* Continuation indicator set by
                                 component */
MQLONG    CompCode;           /* Completion code */
MQLONG    Reason;             /* Reason code qualifying CompCode */
```

IBM i MQZ_CHECK_AUTHORITY (kontrolní oprávnění) na IBM i

Tato funkce je poskytována komponentou autorizační služby MQZAS_VERSION_1 a je vyvolána správcem front za účelem kontroly, zda má entita oprávnění k provedení konkrétní akce nebo akcí na určeném objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_CHECK_AUTHORITY.

Syntaxe

MQZ_CHECK_AUTHORITY ***(QMgrName, EntityName, EntityType,***
ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)

Parametry

Volání MQZ_CHECK_AUTHORITY má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null. Název správce front je komponentě předán pro

informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

EntityName (MQCHAR12)-vstup
Název entity.

Název entity, jejíž autorizace k objektu má být zkontrolována. Maximální délka řetězce je 12 znaků; je-li kratší, je zprava vyplněn mezerami. Název není ukončen znakem null.

Není nezbytné, aby byla tato entita známa základní bezpečnostní službě. Není-li známa, použijí se pro kontrolu autorizace speciální skupiny **nikdo** (do které se předpokládá, že všechny entity patří). Prázdný název je platný a lze jej použít tímto způsobem.

EntityType (MQLONG)-vstup
Typ entity.

Typ entity určený parametrem *EntityName*. Jedná se o jednu z následujících položek:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, ke kterému je požadován přístup. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType (MQLONG)-vstup
Typ objektu.

Typ entity určený parametrem *ObjectName*. Jedná se o jednu z následujících položek:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

Oprávnění (MQLONG)-vstup
Oprávnění ke kontrole.

Pokud je kontrolována jedna autorizace, je toto pole shodné s příslušnou operací autorizace (konstanta MQZAO_ *). Pokud se kontroluje více než jedna autorizace, je to bitová OR odpovídajících konstant MQZAO_ *.

Pro použití volání MQI platí následující oprávnění:

MQZAO_CONNECT
Schopnost používat volání MQCONN.

MQZAO_BROWSE
Schopnost použít volání MQGET s volbou procházení.

To umožňuje zadat volbu MQGMO_BROWSE_FIRST, MQGMO_BROWSE_MSG_UNDER_CURSOR nebo MQGMO_BROWSE_NEXT ve volání MQGET.

MQZAO_INPUT
Schopnost použít volání MQGET s volbou vstupu.

To umožňuje zadat volbu MQOOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE nebo MQOO_INPUT_AS_Q_DEF pro volání MQOPEN.

MQZAO_OUTPUT
Schopnost používat volání MQPUT.

To umožňuje zadat volbu MQOO_OUTPUT ve volání MQOPEN.

MQZAO_INQUIRE
Schopnost používat volání MQINQ.

To umožňuje zadat volbu MQO_INQUIRE ve volání MQOPEN.

MQZAO_SET
Schopnost použít volání MQSET.

To umožňuje zadání volby MQOO_SET ve volání MQOPEN.

MQZAO_PASS_IDENTITY_CONTEXT
Schopnost předat kontext identity.

To umožňuje zadat volbu MQOO_PASS_IDENTITY_CONTEXT pro volání MQOPEN a volbu MQPMO_PASS_IDENTITY_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_PASS_ALL_CONTEXT
Schopnost předat celý kontext.

To umožňuje zadat volbu MQOO_PASS_ALL_CONTEXT pro volání MQOPEN a volbu MQPMO_PASS_ALL_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_SET_IDENTITY_CONTEXT
Schopnost nastavit kontext identity.

To umožňuje, aby volba MQOOO_SET_IDENTITY_CONTEXT byla určena pro volání MQOPEN a volba MQPMO_SET_IDENTITY_CONTEXT pro volání MQPUT a MQPUT1 .

MQZAO_SET_ALL_CONTEXT
Schopnost nastavit celý kontext.

To umožňuje zadat volbu MQOO_SET_ALL_CONTEXT pro volání MQOPEN a volbu MQPMO_SET_ALL_CONTEXT pro volání MQPUT a MQPUT1 .

OPRÁVNĚNÍ uživatele MQZAO_ALTERNATE_USER_AUTHORITY
Schopnost používat alternativní oprávnění uživatele.

To umožňuje zadat volbu MQOOO_ALTERNATE_USER_AUTHORITY pro volání MQOPEN a volbu MQPMO_ALTERNATE_USER_AUTHORITY pro volání MQPUT1 .

MQZAO_ALL_MQI
Všechny autorizace MQI.

To umožňuje všechna dříve popsaná oprávnění.

Pro administraci správce front platí následující autorizace:

MQZAO_CREATE
Schopnost vytvářet objekty určeného typu.

MQZAO_DELETE
Schopnost odstranit určený objekt.

MQZAO_DISPLAY
Schopnost zobrazit atributy uvedeného objektu.

MQZAO_CHANGE
Schopnost změnit atributy uvedeného objektu.

MQZAO_CLEAR
Schopnost odstranit všechny zprávy z uvedené fronty.

MQZAO_AUTHORIZE
Schopnost autorizovat ostatní uživatele pro uvedený objekt.

MQZAO_CONTROL
Schopnost spouštět, zastavovat nebo testovat spojení s objektem kanálu, který není klientem.

MQZAO_CONTROL_EXTENDED
Schopnost resetovat pořadové číslo nebo vyřešit neověřenou zprávu pro objekt kanálu, který není klientem.

MQZAO_ALL_ADMIN
Všechna administrativní oprávnění jiná než MQZAO_CREATE.

Následující autorizace platí jak pro použití MQI, tak pro administraci správce front:

MQZAO_ALL
Všechna oprávnění, jiná než MQZAO_CREATE.

MQZAO_NONE
Žádná oprávnění.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený podle komponenty.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY to má stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                     ObjectType, Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQCHAR12  EntityName;        /* Entity name */
MQLONG    EntityType;        /* Entity type */
MQCHAR48  ObjectName;        /* Object name */
MQLONG    ObjectType;        /* Object type */
MQLONG    Authority;         /* Authority to be checked */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

MQZ_CHECK_PRIVILEGED-Zkontrolujte, zda je uživatel privilegovaný

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_6 a je vyvolána správcem front s cílem určit, zda je určený uživatel privilegovaným uživatelem.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_CHECK_PRIVILEGED.

Syntaxe

MQZ_CHECK_PRIVILEGED(*QMgrName* , *EntityData* , *EntityType* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -vstup

Název správce front. Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Údaje týkající se entity, která má být zkontrolována. Další informace viz “MQZED-Deskriptor entit” na stránce 1672.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený volbou EntityData. Musí to být jedna z následujících hodnot:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ComponentData
Typ: MQBYTExComponentDataLength -vstup/výstup

Data komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z těchto funkcí komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený podle komponenty. Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tato volba stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

Pokud volání komponenty selže (tzn. *CompCode* vrátí MQCC_FAILED) a parametr *Continuation* je MQZCI_DEFAULT nebo MQZCI_CONTINUE, správce front bude i nadále volat další komponenty, pokud existují.

Pokud je volání úspěšné (tj. *CompCode* vrací MQCC_OK), nejsou volány žádné další komponenty bez ohledu na nastavení volby *Continuation* .

Pokud volání selže a parametr *Continuation* je MQZCI_STOP, nejsou volány žádné další komponenty a chyba je vrácena správci front. Komponenty neznají předchozí volání, takže parametr *Continuation* je před voláním vždy nastaven na hodnotu MQZCI_DEFAULT.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí to být jedna z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina
Typ: MQLONG-výstup

Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_NOT_PRIVILEGED
(2584, X'A18') Tento uživatel není ID oprávněného uživatele.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Kódy příčiny a dokončení rozhraní API.

Vyvolání jazyka C

```
MQZ_CHECK_PRIVILEGED (QMgrName, &EntityData, EntityType,
                      ComponentData, &Continuation,
                      &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQZED     EntityData;        /* Entity name */
MQLONG    EntityType;        /* Entity type */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

IBM i MQZ_COPY_ALL_AUTHORITY (Zkopírovat všechna oprávnění) na systému IBM i

Tuto funkci poskytuje komponenta služby autorizace. Je vyvolán správcem front za účelem zkopírování všech autorizací, které jsou aktuálně platné pro objekt odkazu, do jiného objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_COPY_ALL_AUTHORITY.

Syntaxe

MQZ_COPY_ALL_AUTHORITY (***QMgrName, RefObjectName, ObjectName,***
ObjectType, ComponentData, Continuation, CompCode, Reason)

Parametry

Volání MQZ_COPY_ALL_AUTHORITY má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

RefObjectNázev (MQCHAR48)-vstup
Název referenčního objektu.

Název referenčního objektu, oprávnění, pro která se mají kopírovat. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, pro který mají být nastaveny přístupy. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

ObjectType (MQLONG)-vstup
Typ objektu.

Typ objektu určený pomocí *RefObjectName* a *ObjectName*. Jedná se o jednu z následujících položek:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený podle komponenty.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_COPY_ALL_AUTHORITY to má stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_REF_OBJECT
(2294, X'8F6') Objekt odkazu je neznámý.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName, ObjectType,
                        ComponentData, &Continuation, &CompCode,
                        &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQCHAR48  RefObjectName;     /* Reference object name */
MQCHAR48  ObjectName;        /* Object name */
MQLONG    ObjectType;        /* Object type */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

IBM i MQZ_DELETE_AUTHORITY (Oprávnění k odstranění) na IBM i

Tato funkce je poskytována komponentou služby autorizace a je vyvolána správcem front za účelem odstranění všech autorizací přidružených k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_DELETE_AUTHORITY.

Syntaxe

MQZ_DELETE_AUTHORITY ***(QMgrName, ObjectName, ObjectType,***
ComponentData, Continuation, CompCode, Reason)

Parametry

Volání MQZ_DELETE_AUTHORITY má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, pro který se mají odstranit přístupy. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType (MQLONG)-vstup
Typ objektu.

Typ entity určený parametrem *ObjectName*. Jedná se o jednu z následujících položek:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený podle komponenty.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_DELETE_AUTHORITY to má stejný efekt jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_DELETE_AUTHORITY (QMgrName, ObjectName, ObjectType, ComponentData,
                      &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQCHAR48  ObjectName;        /* Object name */
MQLONG    ObjectType;        /* Object type */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

IBM i MQZ_ENUMERATE_AUTHORITY_DATA (Výčet dat oprávnění) na IBM i

Tuto funkci poskytuje komponenta služby autorizace MQZAS_VERSION_4 a správce front ji opakovaně vyvolává za účelem načtení všech dat oprávnění, která odpovídají kritériím výběru zadaným při prvním vyvolání.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_ENUMERATE_AUTHORITY_DATA.

Syntaxe

MQZ_ENUMERATE_AUTHORITY_DATA (*QMgrName, StartEnumeration,*
Filter, AuthorityBufferLength, AuthorityBuffer, AuthorityDataLength,
ComponentData, Continuation, CompCode, Reason)

Parametry

Volání MQZ_ENUMERATE_AUTHORITY_DATA má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

StartEnumeration (MQLONG)-vstup
Příznak označující, zda má volání zahájit výčet.

Označuje, zda má volání spustit výčet dat oprávnění, nebo pokračovat ve výčtu dat oprávnění spuštěném předchozím voláním MQZ_ENUMERATE_AUTHORITY_DATA. Hodnota je jedna z následujících:

MQZSE_START
Počáteční výčet.

Volání je vyvoláno s touto hodnotou pro spuštění výčtu dat oprávnění. Parametr **Filter** uvádí kritéria výběru, která se mají použít k výběru dat oprávnění vrácených tímto a následnými voláními.

MQZSE_CONTINUE
Pokračovat ve výčtu.

Volání je vyvoláno s touto hodnotou, aby pokračovalo ve výčtu dat oprávnění. Parametr **Filter** je v tomto případě ignorován a lze jej zadat jako ukazatel Null (kritéria výběru jsou určena parametrem **Filter** určeným voláním, které mělo hodnotu *StartEnumeration* nastavenou na MQZSE_START).

Filtr (MQZAD)-vstup
Filtr.

Je-li *StartEnumeration* MQZSE_START, *Filter* uvádí kritéria výběru, která se mají použít k výběru dat oprávnění, která se mají vrátit. Pokud je *Filter* ukazatel null, nejsou použita žádná kritéria výběru, to znamená, že jsou vrácena všechna data oprávnění. Podrobnosti o kritériích výběru, která lze použít, naleznete v části “MQZAD (data oprávnění) na IBM i” na stránce 1710 .

Má-li parametr *StartEnumeration* hodnotu MQZSE_CONTINUE, je parametr *Filter* ignorován a lze jej zadat jako ukazatel Null.

AuthorityBufferDélka (MQLONG)-vstup
Délka řetězce *AuthorityBuffer*.

Jedná se o délku parametru **AuthorityBuffer** v bajtech. Vyrovnávací paměť oprávnění musí být dostatečně velká, aby pojmula data, která mají být vrácena.

AuthorityBuffer (MQZAD)-výstup
Data oprávnění.

Jedná se o vyrovnávací paměť, ve které jsou vrácena data oprávnění. Vyrovnávací paměť musí být dostatečně velká, aby pojmula strukturu MQZAD, strukturu MQZED, plus nejdelší definovaný název entity a nejdelší název domény.

Poznámka: Tento parametr je definován jako MQZAD, protože MQZAD se vždy vyskytuje na začátku vyrovnávací paměti. Pokud je však vyrovnávací paměť skutečně deklarována jako MQZAD, bude vyrovnávací paměť příliš malá-musí být větší než MQZAD, aby mohla obsahovat MQZAD, MQZED a názvy entit a domén.

AuthorityDataDélka (MQLONG)-výstup
Délka dat vrácených v souboru *AuthorityBuffer*.

Jedná se o délku dat vrácených v souboru *AuthorityBuffer*. Pokud je vyrovnávací paměť oprávnění příliš malá, je parametr *AuthorityDataLength* nastaven na požadovanou délku vyrovnávací paměti a volání vrátí kód dokončení MQCC_FAILED a kód příčiny MQRC_BUFFER_LENGTH_ERROR.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený podle komponenty.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_ENUMERATE_AUTHORITY_DATA to má stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_NO_DATA_AVAILABLE
(2379, X'94B') Data nejsou k dispozici.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration, &Filter,
                              AuthorityBufferLength,
                              &AuthorityBuffer,
                              &AuthorityDataLength, ComponentData,
                              &Continuation, &CompCode,
                              &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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-Volný uživatel

Tuto funkci poskytuje komponenta autorizační služby MQZAS_VERSION_5 a je vyvolána správcem front za účelem uvolnění přidruženého přiděleného prostředku. Je vyvolána po dokončení spuštění aplikace ve všech uživatelských kontextech, například během volání MQDISC MQI.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_FREE_USER.

IBM i MQZ_GET_AUTHORITY (Získat oprávnění) na systému IBM i

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_1 a je vyvolána správcem front za účelem načtení oprávnění, které má entita pro přístup k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_GET_AUTHORITY.

Syntaxe

MQZ_GET_AUTHORITY ***(QMgrName, EntityName, EntityType, ObjectName,***
ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)

Parametry

Volání MQZ_GET_AUTHORITY má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

EntityName (MQCHAR12)-vstup
Název entity.

Název entity, jejíž přístup k objektu má být načten. Maximální délka řetězce je 12 znaků; je-li kratší, je zprava vyplněn mezerami. Název není ukončen znakem null.

EntityType (MQLONG)-vstup
Typ entity.

Typ entity určený parametrem *EntityName*. Lze zadat následující hodnotu:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, pro který se má načíst oprávnění entity. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType (MQLONG)-vstup
Typ objektu.

Typ entity určený parametrem *ObjectName*. Jedná se o jednu z následujících položek:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

Oprávnění (MQLONG)-výstup
Oprávnění subjektu.

Pokud má entita jedno oprávnění, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, toto pole je bitové OR odpovídajících konstant MQZAO_ *.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený podle komponenty.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_GET_AUTHORITY to má stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_GET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, &Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQCHAR12  EntityName;        /* Entity name */
MQLONG    EntityType;        /* Entity type */
MQCHAR48  ObjectName;        /* Object name */
MQLONG    ObjectType;        /* Object type */
MQLONG    Authority;         /* Authority of entity */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

IBM i MQZ_GET_EXPLICIT_AUTHORITY (získat explicitní oprávnění) na systému IBM i

Tuto funkci poskytuje komponenta služby autorizace MQZAS_VERSION_1 a je vyvolána správcem front za účelem načtení oprávnění, které má pojmenovaná skupina pro přístup k určenému objektu (ale bez dalšího oprávnění skupiny **nikdo**), nebo oprávnění, které má primární skupina pojmenovaného činitele pro přístup k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_GET_EXPLICIT_AUTHORITY.

Syntaxe

MQZ_GET_EXPLICIT_AUTHORITY (*QMgrName, EntityName, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason*)

Parametry

Volání MQZ_GET_EXPLICIT_AUTHORITY má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

EntityName (MQCHAR12)-vstup
Název entity.

Název entity, ze které má být načten přístup k objektu. Maximální délka řetězce je 12 znaků; je-li kratší, je zprava vyplněn mezerami. Název není ukončen znakem null.

EntityType (MQLONG)-vstup
Typ entity.

Typ entity určený parametrem *EntityName*. Lze zadat následující hodnotu:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, pro který se má načíst oprávnění entity. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType (MQLONG)-vstup
Typ objektu.

Typ entity určený parametrem *ObjectName*. Jedná se o jednu z následujících položek:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

Oprávnění (MQLONG)-výstup
Oprávnění subjektu.

Pokud má entita jedno oprávnění, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, toto pole je bitové OR odpovídajících konstant MQZAO_ *.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený podle komponenty.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_GET_EXPLICIT_AUTHORITY má tato volba stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,
                            ObjectName, ObjectType, &Authority,
                            ComponentData, &Continuation,
                            &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQCHAR12  EntityName;        /* Entity name */
MQLONG    EntityType;        /* Entity type */
MQCHAR48  ObjectName;        /* Object name */
MQLONG    ObjectType;        /* Object type */
MQLONG    Authority;         /* Authority of entity */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

IBM i MQZ_INIT_AUTHORITY (Inicializace autorizační služby) na IBM i

Tato funkce je poskytována komponentou služby autorizace a je vyvolána správcem front během konfigurace komponenty. Očekává se volání MQZEP za účelem poskytnutí informací správci front.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_INIT_AUTHORITY.

Syntaxe

MQZ_INIT_AUTHORITY ***(Hconfig, Options, QMgrName, ComponentDataLength,***
ComponentData, Version, CompCode, Reason)

Parametry

Volání MQZ_INIT_AUTHORITY má následující parametry.

Hconfig (MQHCONFIG)-vstup
Popisovač konfigurace.

Tento popisovač představuje konkrétní inicializovanou komponentu. Komponenta ji použije při volání správce front pomocí funkce MQZEP.

Volby (MQLONG)-vstup
Volby inicializace.

Jedná se o jednu z následujících položek:

MQZIO_PRIMARY
Primární inicializace.

MQZIO_SECONDARY
Sekundární inicializace.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

ComponentDataDélka (MQLONG)-vstup
Délka dat komponenty.

Délka oblasti *ComponentData* v bajtech. Tato délka je definována v konfiguračních datech komponenty.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Toto je inicializováno na všechny nuly před voláním primární inicializační funkce komponenty. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; veškeré změny provedené v této komponentě některou z funkcí (včetně inicializační funkce) poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Verze (MQLONG)-vstup/výstup
Číslo verze.

Při vstupu do funkce inicializace je zde uvedeno *nejvyšší* číslo verze, kterou správce front podporuje. Funkce inicializace musí tuto hodnotu v případě potřeby změnit na verzi rozhraní, kterou *podporuje* . Pokud správce front při návratu nepodporuje verzi vrácenou komponentou, volá funkci MQZ_TERM_AUTHORITY komponenty a tuto komponentu již dále nepoužívá.

Podporovány jsou následující hodnoty:

MQZAS_VERSION_1
Verze 1.

MQZAS_VERSION_2
Verze 2.

MQZAS_VERSION_3
Verze 3.

MQZAS_VERSION_4
Verze 4.

MQZAS_VERSION_5
Verze 5.

MQZAS_VERSION_6
IBM WebSphere MQ 6.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_INITIALIZATION_FAILED
(2286, X'8EE') Inicializace se nezdařila z nedefinované příčiny.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,
                    ComponentData, &Version, &CompCode,
                    &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQHCONFIG  Hconfig;              /* Configuration handle */
MQLONG     Options;              /* Initialization options */
MQCHAR48   QMgrName;             /* Queue manager name */
MQLONG     ComponentDataLength;  /* Length of component data */
MQBYTE     ComponentData[n];     /* Component data */
MQLONG     Version;              /* Version number */
MQLONG     CompCode;             /* Completion code */
MQLONG     Reason;               /* Reason code qualifying CompCode */
```

IBM i MQZ_INQUIRE (služba autorizace dotazování) na IBM i

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_5 a je vyvolána správcem front za účelem dotazování na podporované funkce. Pokud se používá více komponent služby, volají se komponenty služby v opačném pořadí, než v jakém byly nainstalovány.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_INQUIRE.

Syntaxe

DOTAZ_MQZ_INQUIRE
(QMgrName, SelectorCount, Selectors, IntAttrCount, IntAttrs, CharAttrLength, CharAttrs, SelectorReturned, ComponentData, Continuation, CompCode, Reason)

Parametry

Volání MQZ_INQUIRE má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

SelectorCount (MQLONG)-vstup
Počet selektorů.

Počet selektorů dodaných v parametru Selektory.

Hodnota musí být mezi nulou a 256.

Selektory (MQLONG x SelectorCount)-vstup
Selektory.

Pole selektorů. Každý selektor identifikuje požadovaný atribut a musí být jednoho z následujících typů:

- MQIACF_ * (celé číslo)
- MQCACF_ * (znak)

Selektory lze zadat v libovolném pořadí. Počet selektorů v poli je označen parametrem SelectorCount .

Celočíselné atributy identifikované selektory jsou vráceny v parametru IntAttrs ve stejném pořadí, v jakém se objevují v selektorech.

Znakové atributy identifikované selektory jsou vráceny v parametru CharAttrs ve stejném pořadí, v jakém se objevují selektory.

IntAttrPočet (MQLONG)-vstup
Počet celočíselných atributů.

Počet celočíselných atributů zadaných v parametru IntAttrs .

Hodnota musí být v rozsahu 0 až 256.

IntAttrs (počet operací MQLONG x IntAttr)-výstup
Celočíselné atributy.

Pole celočíselných atributů. Celočíselné atributy jsou vráceny ve stejném pořadí jako odpovídající celočíselné selektory v poli Selektory.

CharAttrPočet (MQLONG)-vstup
Délka vyrovnávací paměti atributů znaků.

Délka parametru CharAttrs v bajtech.

Hodnota musí být alespoň součtem délek požadovaných znakových atributů. Pokud nejsou požadovány žádné znakové atributy, nula je platná hodnota.

CharAttrs (MQLONG x CharAttrPočet)-výstup
Vyrovnávací paměť atributů znaků.

Vyrovnávací paměť obsahující atributy znaků, zřetězená dohromady. Atributy znaků jsou vráceny ve stejném pořadí jako odpovídající selektory znaků v poli selektorů.

Délka vyrovnávací paměti je dána parametrem počtu CharAttr.

SelectorReturned (početMQLONGxSelector)-vstup
Byl vrácen selektor.

Pole hodnot identifikujících, které atributy byly vráceny ze sady požadované selektory v parametru Selektory. Počet hodnot v tomto poli je označen parametrem SelectorCount . Každá hodnota v poli se vztahuje k selektoru z odpovídající pozice v poli Selektory. Každá hodnota je jedna z následujících:

MQZSL_VRÁCENO
Byl vrácen atribut požadovaný odpovídajícím selektorem v parametru Selektory.

MQZSL_NOT_VRÁCENO
Atribut požadovaný odpovídajícím selektorem v parametru Selektory nebyl vrácen.

Pole je inicializováno se všemi hodnotami jako *MQZSL_NOT_VRÁCENO*. Když komponenta služby autorizace vrátí atribut, nastaví příslušnou hodnotu v poli na *MQZSL_RETURNS*. To umožňuje ostatním komponentám služby autorizace, ke kterým je provedeno volání inquire, identifikovat, které atributy již byly vráceny.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Příznak pokračování.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na jiných komponentách.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_VAROVÁNÍ
Částečné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_VAROVÁNÍ:

MQRC_CHAR_ATTRS_TOO_SHORT
Nedostatek prostoru pro znakové atributy.

MQRC_INT_COUNT_TOO_SMALL
Nedostatek prostoru pro celočíselné atributy.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_SELECTOR_COUNT_ERROR
Počet selektorů není platný.

MQRC_SELECTOR_ERROR
Selektor atributů není platný.

MQRC_SELECTOR_LIMIT_PŘEKROČENO
Bylo zadáno příliš mnoho selektorů.

MQRC_INT_ATTR_COUNT_ERROR
Počet celočíselných atributů není platný.

MQRC_INT_ATTRS_ARRAY_ERROR
Pole celočíselných atributů není platné.

MQRC_CHAR_ATTR_LENGTH_ERROR
Počet znakových atributů není platný.

MQRC_CHAR_ATTRS_ERROR
Řetězec znakových atributů není platný.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Vyvolání jazyka C

```
MQZ_INQUIRE (QMgrName, SelectorCount, Selectors, IntAttrCount,
                  &IntAttrs, CharAttrLength, &CharAttrs,
                  SelectorReturned, ComponentData, &Continuation,
                  &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
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 */
```

IBM i MQZ_REFRESH_CACHE (Aktualizovat všechny autorizace) na IBM i

Tuto funkci poskytuje komponenta služby autorizace MQZAS_VERSION_3 . Je vyvolán správcem front za účelem aktualizace seznamu autorizací interně uchovávaných komponentou.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_REFRESH_CACHE (8L).

Syntaxe

MQZ_REFRESH_CACHE
(QMgrName, ComponentData, pokračování, CompCode, příčina)

Parametry

***QMgrName* (MQCHAR48)-vstup**
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby jej komponenta používala žádným definovaným způsobem.

***ComponentData* (MQBYTE x *ComponentDataDélka*) -vstupní/výstupní**
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Veškeré změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání funkce komponenty.

Délka této datové oblasti je předána správcem front v parametru *ComponentDataLength* volání MQZ_INIT_AUTHORITY.

***Pokračování* (MQLONG)-výstup**
Indikátor pokračování nastavený podle komponenty.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

V případě MQZ_REFRESH_CACHE má tato akce stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

***CompCode* (MQLONG)-výstup**
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

***Příčina* (MQLONG)-výstup**
Kód příčiny, který je kvalifikovaný *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Vyvolání jazyka C

```
MQZ_REFRESH_CACHE (QMgrName, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

IBM i MQZ_SET_AUTHORITY (Nastavit oprávnění) na IBM i

Tato funkce je poskytována komponentou autorizační služby MQZAS_VERSION_1 a je vyvolána správcem front za účelem nastavení oprávnění, které má entita pro přístup k určenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_SET_AUTHORITY.

Poznámka: Tato funkce přepíše všechna existující oprávnění. Chcete-li zachovat existující oprávnění, musíte je znovu nastavit pomocí této funkce.

Syntaxe

MQZ_SET_AUTHORITY ***(QMgrName, EntityName, EntityType, ObjectName,***
ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)

Parametry

Volání MQZ_SET_AUTHORITY má následující parametry.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

EntityName (MQCHAR12)-vstup
Název entity.

Název entity, pro kterou má být nastaven přístup k objektu. Maximální délka řetězce je 12 znaků; je-li kratší, je zprava vyplněn mezerami. Název není ukončen znakem null.

EntityType (MQLONG)-vstup
Typ entity.

Typ entity určený parametrem *EntityName*. Lze zadat následující hodnotu:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, ke kterému je požadován přístup. Maximální délka řetězce je 48 znaků; pokud je kratší, pak je zprava vyplněna mezerami. Název není ukončen znakem null.

Je-li *ObjectType* MQOT_Q_MGR, je tento název stejný jako *QMgrName*.

ObjectType (MQLONG)-vstup
Typ objektu.

Typ entity určený parametrem *ObjectName*. Jedná se o jednu z následujících položek:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

Oprávnění (MQLONG)-vstup
Oprávnění ke kontrole.

Pokud je nastavena jedna autorizace, toto pole se rovná příslušné operaci autorizace (konstanta MQZAO_ *). Je-li nastavena více než jedna autorizace, jedná se o bitovou OR odpovídajících konstant MQZAO_ *.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený podle komponenty.

Lze zadat následující hodnoty:

VÝCHOZÍ
Pokračování závislé na správci front.

Pro MQZ_SET_AUTHORITY to má stejný efekt jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračujte s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_NOT_AUTHORIZED
(2035, X'7F3') Není autorizováno pro přístup.

MQRC_SERVICE_ERROR
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Vyvolání jazyka C

```
MQZ_SET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQCHAR12  EntityName;        /* Entity name */
MQLONG    EntityType;        /* Entity type */
MQCHAR48  ObjectName;        /* Object name */
MQLONG    ObjectType;        /* Object type */
MQLONG    Authority;         /* Authority to be checked */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

MQZ_TERM_AUTHORITY-Ukončete službu autorizace

Tato funkce je poskytována komponentou služby autorizace a je vyvolána správcem front, pokud již nevyžaduje služby této komponenty. Funkce musí provést vyčištění požadované komponentou.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_TERM_AUTHORITY.

Syntaxe

MQZ_TERM_AUTHORITY *(Hconfig, Options, QMgrName, ComponentData, CompCode, Reason)*

Parametry

Volání MQZ_TERM_AUTHORITY má následující parametry.

Hconfig (MQHCONFIG)-vstup
Popisovač konfigurace.

Tento popisovač představuje konkrétní ukončovanou komponentu.

Volby (MQLONG)-vstup
Volby ukončení.

Jedná se o jednu z následujících položek:

MQZTO_PRIMARY
Primární ukončení.

MQZTO_SECONDARY
Sekundární ukončení.

QMgrName (MQCHAR48)-vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je vyplněn mezerami na celou délku parametru; název není ukončen znakem null.

Název správce front je komponentě předán pro informaci; rozhraní služby autorizace nevyžaduje, aby ji komponenta používala jakýmkoli definovaným způsobem.

ComponentData (MQBYTE x ComponentDatadélka)-vstup/výstup
Data komponenty.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Změny provedené v této komponentě některou z funkcí poskytovaných touto komponentou jsou zachovány a prezentovány při příštím volání jedné z funkcí této komponenty.

Délku této datové oblasti předává správce front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Po dokončení volání MQZ_TERM_AUTHORITY správce front tato data zruší.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

MQCC_FAILED
Volání selhalo.

Příčina (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Není důvod k hlášení.

Má-li parametr *CompCode* hodnotu MQCC_FAILED, postupujte takto:

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Základní služba není k dispozici.

MQRC_TERMINATION_FAILED
(2287, X'8FF') Ukončení selhalo z nedefinované příčiny.

Další informace o těchto kódech příčiny naleznete v tématu Zprávy a kódy příčiny.

Vyvolání jazyka C

```
MQZ_TERM_AUTHORITY (Hconfig, Options, QMgrName, ComponentData,
                    &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQHCONFIG  Hconfig;           /* Configuration handle */
MQLONG     Options;           /* Termination options */
MQCHAR48   QMgrName;          /* Queue manager name */
MQBYTE     ComponentData[n];  /* Component data */
MQLONG     CompCode;          /* Completion code */
MQLONG     Reason;            /* Reason code qualifying CompCode */
```

IBM i MQZAC (kontext aplikace) na systému IBM i

Tento parametr určuje data související s volající aplikací.

Struktura MQZAC se používá ve volání MQZ_AUTHENTICATE_USER pro parametr **ApplicationContext** .

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

MQZAC_STRUC_ID
Identifikátor pro strukturu kontextu aplikace.

Pro programovací jazyk C je definována také konstanta MQZAC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZAC_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro službu.

Verze (MQLONG
Číslo verze struktury.

Hodnota je:

MQZAC_VERSION_1
Struktura kontextu aplikace Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

MQZAC_CURRENT_VERSION
Aktuální verze struktury kontextu aplikace.

Toto je vstupní pole pro službu.

ProcessId (MQPID)
Identifikátor procesu.

Identifikátor procesu aplikace.

ThreadId (MQTID)
Identifikátor podprocesu.

Identifikátor podprocesu aplikace.

ApplName (MQCHAR28)
Název aplikace.

Název aplikace.

UserID (MQCHAR12)
Identifikátor uživatele.

Pro systémy IBM i se jedná o profil uživatele, pod kterým byla vytvořena aplikační úloha. (V systému IBM ise vrátí aktuální profil uživatele, když se provede výměna profilu s rozhraním API QWTSETP v aplikační úloze.)

EffectiveUserID (MQCHAR12)
Efektivní identifikátor uživatele.

Pro systémy IBM i aktuální profil uživatele úlohy aplikace.

Prostředí (MQLONG)
Prostředí.

Toto pole určuje prostředí, ze kterého bylo volání provedeno.

Může mít jednu z následujících hodnot:

MQXE_COMMAND_SERVER
Příkazový server.

MQXE_MQSC
Interpret příkazů `runmqsc` .

MQXE_MCA
Agent oznamovacího kanálu

MQXE_OTHER
Nedefinované prostředí

CallerType (MQLONG)
Typ volajícího.

Toto pole uvádí typ programu, který provedl volání.

Může mít jednu z následujících hodnot:

MQXACT_EXTERNÍ
Volání je externí vzhledem ke správci front.

MQXACT_INTERNÍ
Volání je interní pro správce front.

AuthenticationType (MQLONG)
Typ ověřování.

Toto pole určuje typ prováděného ověřování.

Může mít jednu z následujících hodnot:

MQZAT_INITIAL_CONTEXT
Volání ověření je způsobeno inicializací kontextu uživatele. Tato hodnota se používá během volání `MQCONN` nebo `MQCONNX` .

MQZAT_CHANGE_CONTEXT
Volání ověření je způsobeno změnou kontextu uživatele. Tato hodnota se používá, když agent MCA změní kontext uživatele.

v

BindType (MQLONG)
Typ vazby.

Toto pole určuje typ používané vazby.

Může mít jednu z následujících hodnot:

MQCNO_FASTPATH_BINDING
Rychlospojka.

MQCNO_SHARED_BINDING
Sdílená vazba.

MQCNO_ISOLATED_BINDING
Izolovaná vazba.

C prohlášení

```
typedef struct tagMQZAC MQZAC;
struct tagMQZAC {
  MQCHAR4   StrucId;            /* Structure identifier */
  MQLONG    Version;            /* Structure version number */
  MQPID     ProcessId;          /* Process identifier */
  MQTID     ThreadId;           /* Thread identifier */
  MQCHAR28  ApplName;           /* Application name */
  MQCHAR12  UserID;             /* User identifier */
  MQCHAR12  EffectiveUserID;    /* Effective user identifier */
  MQLONG    Environment;        /* Environment */
  MQLONG    CallerType;         /* Caller type */
  MQLONG    AuthenticationType; /* Authentication type */
  MQLONG    BindType;           /* Bind type */
};
```

IBM i MQZAD (data oprávnění) na IBM i

Struktura MQZAD se používá ve volání MQZ_ENUMERATE_AUTHORITY_DATA pro dva parametry.

Další informace o parametrech **Filter** a **AuthorityBuffer** viz “MQZ_ENUMERATE_AUTHORITY_DATA (Výčet dat oprávnění) na IBM i” na stránce 1690 :

- MQZAD se používá pro parametr **Filter** , který je vstupem pro volání. Tento parametr uvádí kritéria výběru, která se mají použít pro výběr dat oprávnění vrácených voláním.
- MQZAD se také používá pro parametr **AuthorityBuffer** , který je výstupem volání. Tento parametr určuje oprávnění pro jednu kombinaci názvu profilu, typu objektu a entity.

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

MQZAD_STRUC_ID
Identifikátor pro datovou strukturu oprávnění.

Pro programovací jazyk C je definována také konstanta MQZAD_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZAD_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro službu.

Verze (MQLONG)
Číslo verze struktury.

Hodnota je:

MQZAD_VERSION_1
Datová struktura oprávnění Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

MQZAD_CURRENT_VERSION
Aktuální verze datové struktury oprávnění.

Toto je vstupní pole pro službu.

ProfileName (MQCHAR48)
Název profilu.

Pro parametr **Filter** je toto pole název profilu, ze kterého se požadují data oprávnění. Pokud je název zcela prázdný až do konce pole nebo prvního znaku null, vrátí se data oprávnění pro všechny názvy profilů.

Pro parametr **AuthorityBuffer** je toto pole název profilu, který odpovídá zadaným kritériím výběru.

ObjectType (MQLONG)
Typ objektu.

Pro parametr **Filter** je toto pole typ objektu, pro který jsou požadována data oprávnění. Pokud je hodnota MQOT_ALL, vrátí se data oprávnění pro všechny typy objektů.

Pro parametr **AuthorityBuffer** je toto pole typem objektu, na který se vztahuje profil označený **ProfileName** .

Hodnota je jedna z následujících; pro parametr **Filter** je platná také hodnota MQOT_ALL:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQOT_NAMELIST
Seznam názvů.

MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

MQOT_SERVICE
Služba.

Oprávnění (MQLONG)
Oprávnění.

Pro parametr **Filter** je toto pole ignorováno.

Pro parametr **AuthorityBuffer** toto pole představuje autorizace, které má entita k objektům označeným jako **ProfileName** a **ObjectType**. Pokud má entita pouze jedno oprávnění, pole se rovná příslušné hodnotě autorizace (konstanta MQZAO_ *). Pokud má entita více než jedno oprávnění, pole je bitové OR odpovídajících konstant MQZAO_ *.

EntityDataPtr (PMQZED)
Adresa struktury MQZED identifikující entitu.

Pro parametr **Filter** toto pole ukazuje na strukturu MQZED, která identifikuje entitu, ze které jsou vyžadována data oprávnění. Je-li **EntityDataPtr** ukazatel null, vrátí se data oprávnění pro všechny entity.

Pro parametr **AuthorityBuffer** toto pole ukazuje na strukturu MQZED, která identifikuje entitu, ze které pocházejí vrácená data oprávnění.

EntityType (MQLONG)
Typ entity.

Pro parametr **Filter** toto pole uvádí typ entity, pro který jsou požadována data oprávnění. Je-li hodnota MQZAET_NONE, vrátí se data oprávnění pro všechny typy entit.

Pro parametr **AuthorityBuffer** toto pole určuje typ entity identifikované strukturou MQZED, na kterou odkazuje **EntityDataPtr**.

Hodnota je jedna z následujících; pro parametr **Filter** je platná také hodnota MQZAET_NONE:

MQZAET_PRINCIPAL
Hlavní.

MQZAET_GROUP
:NONE.

Volby (MQAUTHOPT)
Volby.

Toto pole uvádí volby, které dávají kontrolu nad zobrazenými profily.

Musí být zadána jedna z následujících možností:

MQAUTHOPT_NAME_ALL_MATCHING
Zobrazí všechny profily

MQAUTHOPT_NAME_EXPLICIT
Zobrazí profily, které mají přesně stejný název, jaký je uveden v poli **ProfileName** .

Kromě toho musí být také specifikována jedna z následujících možností:

MQAUTHOPT_ENTITY_SET
Zobrazte všechny profily použité k výpočtu kumulativního oprávnění, které má entita k objektu uvedenému v souboru **ProfileName**. Pole **ProfileName** nesmí obsahovat žádné zástupné znaky.

- Pokud je určená entita činitelem, zobrazí se pro každého člena sady {entity, groups} nejpoužitelnější profil, který se vztahuje k objektu.
- Pokud je určenou entitou skupina, zobrazí se nejpoužitelnější profil ze skupiny, který se vztahuje k objektu.
- Je-li zadána tato hodnota, musí být všechny hodnoty **ProfileName**, **ObjectType**, **EntityType**a název entity určené ve struktuře **EntityDataPtr** MQZED neprázdné.

Pokud jste zadali *MQAUTHOPT_NAME_ALL_MATCHING*, můžete zadat také následující:

MQAUTHOPT_ENTITY_EXPLICIT
Zobrazí profily, které mají přesně stejný název entity jako název entity určený ve struktuře **EntityDataPtr** MQZED.

C prohlášení

```
typedef struct tagMQZAD MQZAD;
struct tagMQZAD {
  MQCHAR4   StrucId;        /* Structure identifier */
  MQLONG    Version;        /* Structure version number */
  MQCHAR48  ProfileName;    /* Profile name */
  MQLONG    ObjectType;     /* Object type */
  MQLONG    Authority;      /* Authority */
  PMQZED    EntityDataPtr;  /* Address of MQZED structure identifying an
                               entity */
  MQLONG    EntityType;     /* Entity type */
  MQAUTHOPT Options;        /* Options */
};
```

IBM i MQZED (deskriptor entity) na systému IBM i

Struktura MQZED se používá v řadě volání služby autorizace k určení entity, pro kterou se má autorizace zkontrolovat.

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

MQZED_STRUC_ID
Identifikátor struktury deskriptoru entity.

Pro programovací jazyk C je definována také konstanta MQZED_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZED_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro službu.

Verze (MQLONG)
Číslo verze struktury.

Hodnota je:

MQZED_VERSION_1
Version-1 struktura deskriptoru entity.

Následující konstanta určuje číslo verze aktuální verze:

MQZED_CURRENT_VERSION
Aktuální verze struktury deskriptoru entity.

Toto je vstupní pole pro službu.

EntityNamePtr (PMQCHAR)
Adresa názvu entity.

Jedná se o ukazatel na název entity, jejíž autorizace se má zkontrolovat.

EntityDomainPtr (PMQCHAR)
Adresa názvu domény entity.

Jedná se o ukazatel na název domény obsahující definici entity, jejíž oprávnění se má zkontrolovat.

SecurityId (MQBYTE40)
Identifikátor zabezpečení.

Jedná se o identifikátor zabezpečení, jehož autorizace se má zkontrolovat.

CorrelationPtr (MQPTR)
Korelační ukazatel.

To usnadňuje předávání korelačních dat mezi funkcí ověření uživatele a dalšími odpovídajícími funkcemi OAM.

C prohlášení

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  PMQCHAR   EntityNamePtr;    /* Address of entity name */
  PMQCHAR   EntityDomainPtr;  /* Address of entity domain name */
  MQBYTE40  SecurityId;       /* Security identifier */
  MQPTR     CorrelationPtr;   /* Address of correlation data */
```

IBM i MQZFP (Volné parametry) na systému IBM i

Tento parametr uvádí data související s uvolněným prostředkem.

Struktura MQZFP se používá ve volání MQZ_FREE_USER pro parametr **FreeParms** .

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

MQZFP_STRUC_ID
Identifikátor struktury volných parametrů.

Pro programovací jazyk C je definována také konstanta MQZFP_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZFP_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro službu.

Verze (MQLONG)
Číslo verze struktury.

Hodnota je:

MQZFP_VERSION_1
Version-1 volná struktura parametrů.

Následující konstanta určuje číslo verze aktuální verze:

MQZFP_CURRENT_VERSION
Aktuální verze struktury volných parametrů.

Toto je vstupní pole pro službu.

Vyhrazeno (MQBYTE8)
Vyhrazené pole.

Počáteční hodnota je null.

CorrelationPtr (MQPTR)
Korelační ukazatel.

Adresa korelačních dat souvisejících s prostředkem, který má být uvolněn.

C prohlášení

```
typedef struct tagMQZFP MQZFP;
struct tagMQZFP {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  MQBYTE8   Reserved;         /* Reserved field */
  MQPTR     CorrelationPtr;   /* Address of correlation data */
};
```

IBM i MQZIC (kontext identity) na systému IBM i

Struktura MQZIC se používá ve volání MQZ_AUTHENTICATE_USER pro parametr **IdentityContext** .

Struktura MQZIC obsahuje informace o kontextu identity, které identifikují uživatele aplikace, která poprvé vložila zprávu do fronty:

- Správce front vyplní pole UserIdentifier názvem, který identifikuje uživatele. Způsob, jakým to může správce front provést, závisí na prostředí, ve kterém je aplikace spuštěna.
- Správce front vyplní pole AccountingToken tokenem nebo číslem, které určuje aplikace, jež vložila zprávu.
- Aplikace mohou použít pole ApplIdentityData pro jakékoli další informace, které chtějí zahrnout o uživateli (například šifrované heslo).

Vhodně autorizované aplikace mohou nastavit kontext identity pomocí funkce MQZ_AUTHENTICATE_USER.

Identifikátor zabezpečení systému Windows (SID) je uložen v poli AccountingToken při vytváření zprávy v adresáři IBM MQ for Windows. Identifikátor SID lze použít k doplnění pole UserIdentifier a k zavedení pověření uživatele.

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

MQZIC_STRUC_ID
Identifikátor pro strukturu kontextu identity.

Pro programovací jazyk C je definována také konstanta MQZIC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZIC_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro službu.

Verze (MQLONG)
Číslo verze struktury.

Hodnota je:

MQZIC_VERSION_1
Struktura kontextu identity Version-1 .

Následující konstanta určuje číslo verze aktuální verze:

MQZIC_CURRENT_VERSION
Aktuální verze struktury kontextu identity.

Toto je vstupní pole pro službu.

UserIdentifier (MQCHAR12)
Identifikátor uživatele.

Toto je část **kontextu identity** zprávy.

UserIdentifier uvádí identifikátor uživatele aplikace, která zprávu vyvolala. Správce front považuje tyto informace za znaková data, ale nedefinuje jejich formát. Další informace o poli *UserIdentifier* viz "UserIdentifier (MQCHAR12) pro MQMD" na stránce 459.

AccountingToken (MQBYTE32)
Účtovací token.

Toto je část **kontextu identity** zprávy.

Produkt *AccountingToken* umožňuje aplikaci, aby způsobila, že práce vykonaná v důsledku zprávy bude řádně zpoplatněna. Správce front považuje tyto informace za řetězec bitů a nekontroluje jejich obsah. Další informace o poli *AccountingToken* viz "AccountingToken (MQBYTE32) pro MQMD" na stránce 461.

Data ApplIdentity(MQCHAR32)
Údaje o aplikaci týkající se identity.

Toto je část **kontextu identity** zprávy.

ApplIdentityData je informace definovaná sadou aplikací, kterou lze použít k poskytnutí dalších informací o původu zprávy. Může být například nastaven aplikacemi, které jsou spuštěny s vhodným oprávněním uživatele, aby označily, zda jsou data identity důvěryhodná. Další informace o poli *ApplIdentityData* viz "ApplIdentity-data (MQCHAR32) pro MQMD" na stránce 462.

C prohlášení

```
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 */
};
```

Třídy a rozhraní produktu IBM MQ .NET

Třídy a rozhraní produktu IBM MQ .NET jsou uvedeny abecedně. Jsou popsány vlastnosti, metody a konstruktory.

Třída MQAsyncStatus.NET

Parametr `MQAsyncStatus` slouží k dotazování na stav předchozí aktivity MQI, například dotazování na úspěch předchozích operací asynchronního vložení. `MQAsyncStatus` zapouzdřuje funkce datové struktury `MQSTS` .

Třída

```
System.Object
    |
    └─ IBM.WMQ.MQBase
        |
        └─ IBM.WMQ.MQBaseObject
            |
            └─ IBM.WMQ.MQAsyncStatus
```

```
public class IBM.WMQ.MQAsyncStatus extends IBM.WMQ.MQBaseObject;
```

- “Vlastnosti” na stránce 1716
- “Konstruktory” na stránce 1717

Vlastnosti

Test pro vyvolání `MQException` při získávání vlastností.

`public static int CompCode {get;}`

Kód dokončení z první chyby nebo varování.

`public static int Reason {get;}`

Kód příčiny z první chyby nebo varování.

`public static int PutSuccessCount {get;}`

Počet úspěšných asynchronních volání vložení MQI.

`public static int PutWarningCount {get;}`

Počet asynchronních volání vložení MQI, která byla úspěšná s varováním.

`public static int PutFailureCount {get;}`

Počet neúspěšných asynchronních volání vložení MQI.

`public static int ObjectType {get;}`

Typ objektu pro první chybu. Možné jsou následující hodnoty:

- `MQC.MQOT_ALIAS_Q`
- `MQC.MQOT_LOCAL_Q`
- `MQC.MQOT_MODEL_Q`
- `MQC.MQOT_Q`

- MQC.MQOT_REMOTE_Q
- MQC.MQOT_TOPIC
- 0, což znamená, že není vrácen žádný objekt.

public static string ObjectName {get;}

Název objektu.

public static string ObjectQMgrName {get;}

Název správce front objektů.

public static string ResolvedObjectName {get;}

Název vyřešeného objektu.

public static string ResolvedObjectQMgrName {get;}

Název vyřešeného správce front objektů.

Konstruktory

public MQAsyncStatus() throws MQException;

Metoda konstruktoru, vytvoří objekt s poli inicializovanými na nulu nebo prázdný podle potřeby.

Třída MQAuthenticationInformationRecord.NET

MQAuthenticationInformationRecord použijte k uvedení informací o ověřovateli, který se má použít v připojení klienta IBM MQ TLS. MQAuthenticationInformationRecord zapouzdřuje záznam ověřovacích informací MQAIR.

Třída

```
System.Object
    |
    └─ IBM.WMQ.MQAuthenticationInformationRecord
```

```
public class IBM.WMQ.MQAuthenticationInformationRecord extends System.Object;
```


- "Vlastnosti" na stránce 1717
- "Konstruktory" na stránce 1718

Vlastnosti

Test pro vyvolání MQException při získávání vlastností.

public long Version {get; set;}

Číslo verze struktury.

public long AuthInfoType {get; set;}

Typ ověřovacích informací. Tento atribut musí být nastaven na jednu z následujících hodnot:

- OCSP -Kontrola stavu odvolání certifikátu se provádí pomocí protokolu OCSP.
- CRLLDAP -Kontrola stavu odvolání certifikátů se provádí pomocí seznamů odvolaných certifikátů na serverech LDAP.

public string AuthInfoConnName {get; set;}

Název DNS nebo adresa IP hostitele, na kterém je spuštěn server LDAP, s volitelným číslem portu. Toto klíčové slovo je požadované.

public string LDAPPassword {get; set;}

Heslo přidružené k rozlišujícímu názvu uživatele, který přistupuje k serveru LDAP. Tato vlastnost platí pouze v případě, že je parametr **AuthInfoType** nastaven na hodnotu CRLLDAP.

public string LDAPUserName {get; set;}

Rozlišující název uživatele, který přistupuje k serveru LDAP. Když nastavíte tuto vlastnost, LDAPUserNameLength a LDAPUserNamePtr se automaticky nastaví správně. Tato vlastnost se používá pouze v případě, že je volba AuthInfoType nastavena na hodnotu CRLLDAP.

public string OCSPResponderURL {get; set;}

Adresa URL, na níž lze kontaktovat odpovídací modul OCSP. Tato vlastnost se používá pouze v případě, že je vlastnost AuthInfoType nastavena na hodnotu OCSP .

V tomto poli se rozlišují malá a velká písmena. Musí začínat řetězcem http:// malými písmeny. Zbytek URL může v závislosti na implementaci serveru OCSP rozlišovat malá a velká písmena.

Konstruktory

MQAuthenticationInformationRecord();

Třída MQDestination.NET

Použijte MQDestination pro přístup k metodám, které jsou společné pro MQQueue a MQTopic. MQDestination je abstraktní základní třída a nelze ji převést na instanci.

Třída

```
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;

- “Vlastnosti” na stránce 1718
- “metody” na stránce 1719
- “Konstruktory” na stránce 1720

Vlastnosti

Test pro vyvolání MQException při získávání vlastností.

public DateTime CreationDateTime {get;}

Datum a čas vytvoření fronty nebo tématu. Tato vlastnost, která byla původně obsažena v souboru MQQueue, byla přesunuta do základní třídy MQDestination .

Není zde žádná výchozí hodnota.

public int DestinationType {get;}

Celočíselná hodnota popisující typ použitého místa určení. Tato hodnota, inicializovaná z konstruktoru podtříd MQQueue nebo MQTopic, může mít jednu z těchto hodnot:

- MQOT_Q
- MQOT_TOPIC

Není zde žádná výchozí hodnota.

metody

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);

Vyvolá výjimku MQException.

Získá zprávu z fronty, pokud je cílem objekt MQQueue , nebo z tématu, pokud je cílem objekt MQTopic , s použitím výchozí instance MQGetMessageOptions pro získání.

Pokud se příkaz get nezdaří, objekt MQMessage se nezmění. V případě úspěchu jsou deskriptor zprávy a datové části zprávy MQMessage nahrazeny deskriptorem zprávy a daty zprávy z příchozí zprávy.

Všechna volání IBM MQ z určitého MQQueueManager jsou synchronní. Proto, pokud provedete operaci get s wait, všechny ostatní podprocesy používající stejný MQQueueManager budou blokovány pro další volání IBM MQ , dokud nebude volání Get dokončeno. Potřebujete-li pro přístup k produktu IBM MQ současně více podprocesů, musí každý podproces vytvořit svůj vlastní objekt MQQueueManager .

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá pole v deskriptoru zprávy jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry MessageId a CorrelationId byly nastaveny podle potřeby.

Klient s možností opětovného připojení vrátí kód příčiny MQRC_BACKED_OUT po úspěšném opětovném připojení pro zprávy přijaté pod MQGM_SYNCPOINT.

Volby getMessage
Volby, které řídí akci získání.

Použití volby MQC.MQGMO_CONVERT může vést k výjimce s kódem příčiny MQC.MQRC_CONVERTED_STRING_TOO_BIG při převodu z jednobajtových znakových kódů na dvoubajtové kódy. V tomto případě se zpráva zkopíruje do vyrovnávací paměti bez převodu.

Není-li parametr *getMessageOptions* uveden, použije se volba zprávy MQGMO_NOWAIT.

Pokud použijete volbu MQGMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Velikost MaxMsg
Největší zpráva, kterou má tento objekt zprávy přijmout. Pokud je zpráva ve frontě větší než tato velikost, dojde k jedné ze dvou věcí:

- Je-li v objektu MQGetMessageOptions nastaven příznak MQGMO_ACCEPT_TRUNCATED_MSG , zpráva se vyplní co nejvíce dat zprávy. Dojde k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_ACCEPTED .
- Není-li příznak MQGMO_ACCEPT_TRUNCATED_MSG nastaven, zpráva zůstane ve frontě. Dojde k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_FAILED .

Není-li zadána hodnota *MaxMsgSize* , bude načtena celá zpráva.

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

Vyvolá výjimku MQException.

Vloží zprávu do fronty, pokud je cílem objekt MQQueue , nebo publikuje zprávu do tématu, pokud je cílem objekt MQTopic .

Úpravy objektu MQMessage po provedení volání Put neovlivní skutečnou zprávu ve frontě IBM MQ nebo tématu publikování.

Produkt Put aktualizuje vlastnosti MessageId a CorrelationId objektu MQMessage a nevymaže data zprávy. Další volání Put nebo Get odkazují na aktualizované informace v objektu MQMessage . Například v následujícím úseku kódu obsahuje první zpráva a a druhou ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

zpráva
Objekt MQMessage obsahující data deskriptoru zprávy a zprávu, která má být odeslána. Deskriptor zprávy může být změněn v důsledku této metody. Hodnoty v deskriptoru zprávy bezprostředně po dokončení této metody jsou hodnoty, které byly vloženy do fronty nebo publikovány do tématu.

Klientovi s možností opětovného připojení jsou vráceny následující kódy příčiny:

- MQRC_CALL_INTERRUPTED , pokud je připojení přerušeno při spuštění volání Put pro trvalou zprávu a opětovné připojení je úspěšné.
- MQRC_NONE je-li připojení úspěšné při spuštění volání Put pro dočasnou zprávu (viz Obnova aplikace).

Volby putMessage
Volby, které řídí akci vložení.

Není-li parametr *putMessageOptions* uveden, použije se výchozí instance MQPutMessageOptons .

Pokud použijete volbu MQPMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Poznámka: Chcete-li pro jednoduchost a výkon vložit jednu zprávu do fronty, použijte objekt MQQueueManager.Put . Pro tento účel byste měli mít objekt MQQueue .

Konstruktory
MQDestination je abstraktní základní třída a nelze ji převést na instanci. Přistupujte k cílům pomocí konstruktorů MQQueue a MQTopic nebo pomocí konstruktorů MQQueueManager.AccessQueue a MQQueueManager.AccessTopic methods.

Třída MQEnvironment.NET

Pomocí volby MQEnvironment můžete řídit, jak je konstruktor MQQueueManager volán, a vybrat připojení IBM MQ MQI client . Třída MQEnvironment obsahuje vlastnosti, které řídí chování IBM MQ.

Třída

```
System.Object
        |
        └─ IBM.WMQ.MQEnvironment
```

```
public class IBM.WMQ.MQEnvironment extends System.Object;
```

- “Vlastnosti-pouze klient” na stránce 1721
- “Vlastnosti” na stránce 1722
- “Konstruktory” na stránce 1723

Vlastnosti-pouze klient

Test pro vyvolání `MQException` při získávání vlastností.

`public static int CertificateValPolicy {get; set;}`

Nastavte, která zásada ověření certifikátu TLS se používá k ověření digitálních certifikátů přijatých ze vzdálených partnerských systémů. Platné jsou tyto hodnoty:

- MQC.CERTIFICATE_VALIDATION_POLICY_ANY
- MQC.CERTIFICATE_VALIDATION_POLICY_RFC5280

`public static ArrayList EncryptionPolicySuiteB {get; set;}`

Nastavte úroveň šifrování vyhovující standardu Suite B. Platné jsou tyto hodnoty:

- MQC.MQ_SUITE_B_NONE -Toto je výchozí hodnota.
- MQC.MQ_SUITE_B_128_BIT
- MQC.MQ_SUITE_B_192_BIT

`public static string Channel {get; set;}`

Název kanálu pro připojení k cílovému správci front. Před vytvořením instance `MQQueueManager` v režimu klienta musíte nastavit vlastnost kanálu.

`public static int FipsRequired {get; set;}`

Uveďte `MQC.MQSSL_FIPS_YES` , chcete-li použít pouze algoritmy certifikované FIPS, pokud se šifrování provádí v produktu IBM MQ. Výchozí hodnota je `MQC.MQSSL_FIPS_NO`.

Je-li konfigurován kryptografický hardware, používají se kryptografické moduly, které poskytuje hardwarový produkt. V závislosti na používaném hardwaru nemusí být tato zařízení certifikována podle standardu FIPS na konkrétní úrovni.

`public static string Hostname {get; set;}`

Název hostitele TCP/IP počítače, na kterém je umístěn server IBM MQ . Není-li název hostitele nastaven a nejsou-li nastaveny žádné vlastnosti potlačení, použije se pro připojení k lokálnímu správci front režim vazeb serveru.

`public static int Port {get; set;}`

Port, ke kterému se má připojit. Jedná se o port, na kterém server IBM MQ naslouchá příchozím požadavkům na připojení. Výchozí hodnota je 1414.

`public static string SSLCipherSpec {get; set;}`

Nastavte volbu `SSLCipherSpec` na hodnotu CipherSpec nastavenou na kanálu SVRCONN, chcete-li povolit zabezpečení TLS pro připojení. Výchozí hodnota je Null a pro připojení není povolen protokol TLS.

`public static string sslPeerName {get; set;}`

Vzor rozlišujícího názvu. Je-li nastavena hodnota `sslCipherSpec` , lze tuto proměnnou použít k zajištění správného použití správce front. Je-li hodnota nastavena na null (výchozí), DN správce front se neprovede. `sslPeerName` se ignoruje, pokud má `sslCipherSpec` hodnotu null.

V 9.3.0 V 9.3.0 **`public static string SSLKeyRepository {get; set;}`**

Prostý text nebo šifrovaná přístupová fráze pro přístup k úložišti klíčů. Přístupové fráze úložiště klíčů jsou šifrovány pro použití klientskými aplikacemi pomocí obslužného programu **runmqicred** .

Je-li parametr `SSLKeyRepositoryPassword` nastaven na hodnotu null (výchozí), použije se hodnota proměnné prostředí *MQKEYRPWD* nebo atribut **SSLKeyRepositoryPassword** v konfiguračním souboru klienta.

V 9.3.0 V 9.3.0 **`public static string InitialKey {get; set;}`**

Počáteční klíč, který byl použit k zašifrování přístupové fráze úložiště klíčů uvedené v souboru `SSLKeyRepositoryPassword`.

Počáteční klíč musí být uveden, pokud byl uveden počáteční soubor s klíči, když byla přístupová fráze úložiště klíčů zašifrována pomocí obslužného programu **runmqicred** .

Vlastnosti

Test pro vyvolání `MQException` při získávání vlastností.

public static ArrayList HdrCompList {get; set;}
Seznam komprese dat záhlaví

public static int KeyResetCount {get; set;}
Označuje počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS, než je znovu vyjednán tajný klíč.

public static ArrayList MQAIRArray {get; set;}
Pole objektů `MQAuthenticationInformationRecord` .

public static ArrayList MsgCompList {get; set;}
Seznam komprese dat zpráv

public static string Password {get; set;}
Heslo, které má být ověřeno. Heslo odkazované ze struktury `MQCSP` se naplní nastavením této vlastnosti Heslo.

public static string ReceiveExit {get; set;}
Uživatelská procedura pro příjem umožňuje kontrolovat a měnit data přijatá ze správce front. Obvykle se používá s odpovídající uživatelskou procedurou pro odesílání ve správci front. Je-li volba ReceiveExit nastavena na hodnotu null, není volána žádná uživatelská procedura příjmu.

public static string ReceiveUserData {get; set;}
Uživatelská data přidružená k uživatelské proceduře pro příjem. Omezeno na 32 znaků.

public static string SecurityExit {get; set;}
Uživatelská procedura zabezpečení umožňuje upravit toky zabezpečení, které se vyskytnou při pokusu o připojení ke správci front. Je-li volba `SecurityExit` nastavena na hodnotu null, není volána žádná uživatelská procedura zabezpečení.

public static string SecurityUserData {get; set;}
Uživatelská data přidružená k uživatelské proceduře zabezpečení. Omezeno na 32 znaků.

public static string SendExit {get; set;}
Uživatelská procedura odeslání vám umožňuje zkontrolovat nebo změnit data odeslaná správci front. Obvykle se používá s odpovídající uživatelskou procedurou pro příjem ve správci front. Pokud je volba `SendExit` nastavena na hodnotu null, není volána žádná uživatelská procedura odeslání.

public static string SendUserData {get; set;}
Uživatelská data přidružená k uživatelské proceduře pro odeslání. Omezeno na 32 znaků.

public static string SharingConversations {get; set;}

Pole `SharingConversations` se používá pro připojení z aplikací .NET , když tyto aplikace nepoužívají tabulku CCDT (Client Channel Definition Table).

`SharingConversations` určuje maximální počet konverzací, které lze sdílet na soketu přidruženém k tomuto připojení.

Hodnota 0 znamená, že kanál pracuje tak, jak fungoval před produktem IBM WebSphere MQ 7.0, s ohledem na sdílení konverzace, dopředné čtení a prezenční signál.

Pole je předáno v hašovací tabulce vlastností jako `SHARING_CONVERSATIONS_PROPERTY`při vytváření instance správce front IBM MQ .

Pokud neuvedete `SharingConversations`, použije se výchozí hodnota 10.

public static string SSLCryptoHardware {get; set;}

Nastaví název řetězce parametru požadovaného pro konfiguraci šifrovacího hardwaru přítomného v systému. Parametr `SSLCryptoHardware` se ignoruje, pokud má `sslCipherSpec` hodnotu null.

public static string SSLKeyRepository {get; set;}

Nastavte úplný název souboru úložiště klíčů.

V 9.3.0 V 9.3.0 Pokud není přípona souboru dodána, předpokládá se, že je `.kdb`

Je-li `SSLKeyRepository` nastaveno na hodnotu null (výchozí), použije se k vyhledání úložiště klíčů proměnná prostředí `MQSSLKEYR` certifikátu.

`public static string UserId {get; set;}`

ID uživatele, který má být ověřen. ID uživatele odkazované ze struktury `MQCSP` se naplní nastavením `UserId`. Ověřte `UserId` pomocí rozhraní API nebo uživatelské procedury zabezpečení.

Konstruktory

`public MQEnvironment()`

Třída MQException.NET

Použijte `MQException` , abyste zjistili dokončení a kód příčiny nezdařené funkce IBM MQ . Při výskytu chyby IBM MQ dojde k výjimce `MQException` .

Třída

```
System.Object
        |
        └─ System.Exception
                |
                └─ System.ApplicationException
                        |
                        └─ IBM.WMQ.MQException
```

`public class IBM.WMQ.MQException extends System.ApplicationException;`

- “Vlastnosti” na stránce 1723
- “Konstruktory” na stránce 1723

Vlastnosti

`public int CompletionCode {get; set;}`
Kód dokončení IBM MQ přidružený k chybě. Možné hodnoty jsou:

- `MQException.MQCC_OK`
- `MQException.MQCC_WARNING`
- `MQException.MQCC_FAILED`

`public int ReasonCode {get; set;}`
Kód příčiny IBM MQ popisující chybu.

Konstruktory

`public MQException(int completionCode, int reasonCode)`

completionCode
Kód dokončení IBM MQ .

reasonCode
Kód dokončení IBM MQ .

Třída MQGetMessageOptions.NET

Pomocí volby `MQGetMessageOptions` můžete určit, jak mají být zprávy načítány. Upravuje chování `MQDestination.Get`.

Třída


```
public class IBM.WMQ.MQGetMessageOptions extends IBM.WMQ.MQBaseObject;
```

- “Vlastnosti” na stránce 1724
- “Konstruktory” na stránce 1726

Vlastnosti

Poznámka: Chování některých voleb dostupných v této třídě závisí na prostředí, ve kterém jsou použity. Tyto prvky jsou označeny hvězdičkou *.

Test pro vyvolání `MQException` při získávání vlastností.

public int GroupStatus {get;}*
`GroupStatus` označuje, zda je načtená zpráva ve skupině a zda je poslední ve skupině. Možné hodnoty jsou:

MQC.MQGS_LAST_MSG_IN_GROUP
Zpráva je poslední nebo jediná zpráva ve skupině.

MQC.MQGS_MSG_IN_GROUP
Zpráva je ve skupině, ale není poslední ve skupině.

MQC.MQGS_NOT_IN_GROUP
Zpráva není ve skupině.

public int MatchOptions {get; set;}*
`MatchOptions` určuje, jak je zpráva vybrána. Lze nastavit následující volby shody:

MQC.MQMO_MATCH_CORREL_ID
ID korelace, pro které má být nalezena shoda.

MQC.MQMO_MATCH_GROUP_ID
ID skupiny, která se má porovnat.

MQC.MQMO_MATCH_MSG_ID
ID zprávy, pro kterou má být nalezena shoda.

MQC.MQMO_MATCH_MSG_SEQ_NUMBER
Odpovídá pořadovému číslu zprávy.

MQC.MQMO_NONE
Není vyžadována žádná shoda.

public int Options {get; set;}
`Volby` řídí akci `MQQueue.get`. Lze zadat libovolnou z následujících hodnot. Je-li požadována více než jedna volba, lze hodnoty přidat nebo zkombinovat pomocí bitové operátoru OR.

MQC.MQGMO_ACCEPT_TRUNCATED_MSG
Povolit oříznutí dat zprávy.

MQC.MQGMO_ALL_MSGS_AVAILABLE*
Načíst zprávy ze skupiny pouze v případě, že jsou k dispozici všechny zprávy ve skupině.

MQC.MQGMO_ALL_SEGMENTS_AVAILABLE*
Načíst segmenty logické zprávy pouze v případě, že jsou k dispozici všechny segmenty ve skupině.

MQC.MQGMO_BROWSE_FIRST
Procházet od začátku fronty.

MQC.MQGMO_BROWSE_MSG_UNDER_CURSOR*
Procházet zprávu pod kurzorem procházení.

MQC.MQGMO_BROWSE_NEXT
Procházejte z aktuální pozice ve frontě.

MQC.MQGMO_COMPLETE_MSG*
Načíst pouze úplné logické zprávy.

MQC.MQGMO_CONVERT
Před zkopírováním dat do vyrovnávací paměti zpráv požádejte o převod dat aplikace tak, aby byla v souladu s atributy `CharacterSet` a `Encoding` produktu `MQMessage`. Protože se převod dat používá také při načítání dat z vyrovnávací paměti zpráv, aplikace tuto volbu nenastavují.

Použití této volby může způsobit problémy při převodu z jednobajtových znakových sad na dvoubajtové znakové sady. Místo toho proveďte převod pomocí metod `readString`, `readLine`a `writeString` po doručení zprávy.

MQC.MQGMO_FAIL_IF_QUIESCING
Pokud je správce front uveden do klidového stavu, dojde k selhání.

MQC.MQGMO_LOCK*
Zamknout procházenou zprávu.

MQC.MQGMO_LOGICAL_ORDER*
Vrácení zpráv ve skupinách a segmentech logických zpráv v logickém pořadí.

Pokud použijete volbu `MQGMO_LOGICAL_ORDER` v opětovně připojitelném klientu, vrátí se aplikaci kód příčiny `MQRC_RECONNECT_INCOMPATIBLE` .

MQC.MQGMO_MARK_SKIP_BACKOUT*
Povolit, aby byla jednotka práce vrácena zpět bez opětovného uvedení zprávy ve frontě.

MQC.MQGMO_MSG_UNDER_CURSOR
Získat zprávu pod kurzorem procházení.

MQC.MQGMO_NONE
Nebyly zadány žádné další volby; všechny volby předpokládají své výchozí hodnoty.

MQC.MQGMO_NO_PROPERTIES
Nejsou načteny žádné vlastnosti zprávy, kromě vlastností obsažených v deskriptoru zprávy (nebo rozšíření).

MQC.MQGMO_NO_SYNCPOINT
Získat zprávu bez řízení synchronizačního bodu.

MQC.MQGMO_NO_WAIT
Okamžitě se vraťte, pokud není žádná vhodná zpráva.

MQC.MQGMO_PROPERTIES_AS_Q_DEF
Načtěte vlastnosti zprávy, jak jsou definovány atributem `PropertyControl` `MQQueue`. Přístup k vlastnostem zprávy v deskriptoru nebo rozšíření zprávy není ovlivněn atributem `PropertyControl` .

MQC.MQGMO_PROPERTIES_COMPATIBILITY
Načtěte vlastnosti zprávy s předponou `mcd`, `jms`, `usr`nebo `mqext`v záhlaví `MQRFH2` . Ostatní vlastnosti zprávy, s výjimkou vlastností obsažených v deskriptoru zprávy nebo rozšíření, jsou vyřazeny.

MQC.MQGMO_PROPERTIES_FORCE_MQRFH2
Načtěte vlastnosti zprávy, s výjimkou vlastností obsažených v deskriptoru zprávy nebo rozšíření, v záhlavích `MQRFH2` . Volbu `MQC.MQGMO_PROPERTIES_FORCE_MQRFH2` použijte v aplikacích, které očekávají načtení vlastností, ale které nelze změnit tak, aby používaly manipulátory zpráv.

MQC.MQGMO_PROPERTIES_IN_HANDLE
Načtěte vlastnosti zprávy pomocí `MsgHandle`.

MQC.MQGMO_SYNCPOINT
Získejte zprávu pod řízením synchronizačního bodu. Zpráva je označena jako nedostupná pro jiné aplikace, ale je odstraněna z fronty pouze v případě, že je transakce potvrzena. Zpráva se znovu zpřístupní, pokud je jednotka práce vrácena zpět.

MQC.MQGMO_SYNCPOINT_IF_PERSISTENT*
Získat zprávu s řízením synchronizačního bodu, pokud je zpráva trvalá.

MQC.MQGMO_UNLOCK*
Odemknout dříve zamknutou zprávu.

MQC.MQGMO_WAIT
Počkejte na doručení zprávy.

public string ResolvedQueueName {get;}
Správce front nastaví `ResolvedQueueName` na lokální název fronty, ze které byla zpráva načtena. `ResolvedQueueNázev` se liší od názvu použitého k otevření fronty, pokud byla otevřena alias fronta nebo modelová fronta.

public char Segmentation {get;}*
`Segmentace` označuje, zda můžete povolit segmentaci pro načtenou zprávu. Možné hodnoty jsou:

MQC.MQSEG_INHIBITED
Nepovolit segmentaci.

MQC.MQSEG_ALLOWED
Povolit segmentaci

public byte SegmentStatus {get;}*
`SegmentStatus` je výstupní pole, které označuje, zda je načtená zpráva segmentem logické zprávy. Pokud je zpráva segmentem, příznak označuje, zda se jedná o poslední segment. Možné hodnoty jsou:

MQC.MQSS_LAST_SEGMENT
Zpráva je posledním nebo jediným segmentem logické zprávy.

MQC.MQSS_NOT_A_SEGMENT
Zpráva není segmentem.

MQC.MQSS_SEGMENT
Zpráva je segmentem, ale není posledním segmentem logické zprávy.

public int WaitInterval {get; set;}
`WaitInterval` je maximální doba v milisekundách, po kterou volání `MQQueue.get` čeká na doručení vhodné zprávy. Použijte `WaitInterval` s `MQC.MQGMO_WAIT`. Nastavte hodnotu `MQC.MQWI_UNLIMITED` , chcete-li čekat neomezenou dobu na zprávu.

Konstruktory

public MQGetMessageOptions()
Vytvořte nový objekt `MQGetMessageOptions` s volbou `Options` nastavenou na `MQC.MQGMO_NO_WAIT`, `WaitInterval` nastavenou na nulu a `ResolvedQueueName` nastavenou na prázdnou hodnotu.

Třída MQManagedObject.NET

Pomocí příkazu `MQManagedObject` můžete zjišťovat a nastavovat atributy položek `MQDestination`, `MQProcess`, `MQQueueManagera` `MQSubscription`. `MQManagedObject` je supertřídou těchto tříd.

Třídy

`public class IBM.WMQ.MQManagedObject extends IBM.WMQ.MQBaseObject;`

- "Vlastnosti" na stránce 1727
- "metody" na stránce 1728
- "Konstruktory" na stránce 1729

Vlastnosti

Test pro vyvolání `MQException` při získávání vlastností.

public string AlternateUserId {get; set;}
Alternativní ID uživatele, pokud existuje, nastaveno při otevření prostředku. `AlternateUserID.set` je ignorován, když je vydán pro objekt, který je otevřený. `AlternateUserId` není platné pro odběry.

public int CloseOptions {get; set;}

Tento atribut nastavte, chcete-li řídit způsob zavření prostředku. Výchozí hodnota je `MQC.MQCO_NONE`. `MQC.MQCO_NONE` je jedinou přípustnou hodnotou pro všechny prostředky kromě trvalých dynamických front, dočasných dynamických front, odběrů a témat, ke kterým přistupují objekty, které je vytvořily.

Pro fronty a témata jsou přípustné následující další hodnoty:

MQC.MQCO_DELETE
Pokud nejsou k dispozici žádné zprávy, odstraňte frontu.

MQC.MQCO_DELETE_PURGE
Odstraňte frontu a vymažte v ní všechny zprávy.

MQC.MQCO_QUIESCE
Požadujte zavření fronty, obdržíte varování, pokud zůstanou nějaké zprávy (což umožňuje jejich načtení před konečným zavřením).

U předplatného jsou přípustné následující dodatečné hodnoty:

MQC.MQCO_KEEP_SUB
Odběr nebyl odstraněn. Tato volba je platná pouze v případě, že původní odběr je trvalý. `MQC.MQCO_KEEP_SUB` je výchozí hodnota pro trvalé téma.

MQC.MQCO_REMOVE_SUB
Odběr byl odstraněn. `MQC.MQCO_REMOVE_SUB` je výchozí hodnota pro téma, které není trvalé a nespravované.

MQC.MQCO_PURGE_SUB
Odběr byl odstraněn. `MQC.MQCO_PURGE_SUB` je výchozí hodnota pro dočasné spravované téma.

public MQQueueManager ConnectionReference {get;}
Správce front, ke kterému tento prostředek patří.

public string MQDescription {get;}
Popis prostředku zadrženého správcem front. `MQDescription` vrací prázdný řetězec pro odběry a témata.

public boolean IsOpen {get;}
Označuje, zda je prostředek momentálně otevřený.

public string Name {get;}
Název prostředku. Název je buď zadaný v přístupové metodě, nebo přidělený správcem front pro dynamickou frontu.

public int OpenOptions {get; set;}
Volby OpenOptions jsou nastaveny při otevření objektu IBM MQ . Metoda `OpenOptions.set` je ignorována a nezpůsobuje chybu. Odběry nemají žádné volby `OpenOptions`.

metody

public virtual void Close();
Vyvolá výjimku `MQException`.

Zavře objekt. Po volání funkce `Close`nejsou povoleny žádné další operace pro tento prostředek. Chcete-li změnit chování metody `Close` , nastavte atribut `closeOptions` .

public string GetAttributeString(int *selector*, int *length*);
Vyvolá výjimku `MQException`.

Získá řetězec atributu.

selektor
Celé číslo označující, který atribut je dotazován.

délka
Celé číslo označující požadovanou délku řetězce.

public void Inquire(int[] *selectors*, int[] *intAttrs*, byte[] *charAttrs*);
Vyvolá výjimku `MQException`.

Vrací pole celých čísel a sadu znakových řetězců obsahujících atributy fronty, procesu nebo správce front. Atributy, které se mají dotazovat, jsou uvedeny v poli selektorů.

Poznámka: Na mnoho běžnějších atributů se lze dotazovat pomocí metod `Get` definovaných v `MQManagedObject`, `MQQueue` a `MQQueueManager`.

selektory
Celočíselné pole identifikující atributy s hodnotami, které mají být zjišťovány.

intAttrs
Pole, ve kterém jsou vráceny hodnoty celočíselných atributů. Hodnoty celočíselných atributů jsou vráceny ve stejném pořadí jako selektory celočíselných atributů v poli selektorů.

charAttrs
Vyrovnávací paměť, ve které jsou vráceny atributy znaků, zřetězená. Znakové atributy jsou vráceny ve stejném pořadí jako selektory znakových atributů v poli selektorů. Délka každého řetězce atributu je pevná pro každý atribut.

public void Set(int[] *selectors*, int[] *intAttrs*, byte[] *charAttrs*);
Vyvolá výjimku `MQException`.

Nastaví atributy definované ve vektoru selektorů. Atributy, které se mají nastavit, jsou určeny v poli selektorů.

selektory
Celočíselné pole identifikující atributy s hodnotami, které mají být nastaveny.

intAttrs
Pole hodnot celočíselných atributů, které mají být nastaveny. Tyto hodnoty musí být ve stejném pořadí jako selektory celočíselných atributů v poli selektorů.

charAttrs
Vyrovnávací paměť, ve které jsou zřetězeny atributy znaků, které mají být nastaveny. Tyto hodnoty musí být ve stejném pořadí jako selektory znakových atributů v poli selektorů. Délka každého znakového atributu je pevná.

public void SetAttributeString(int *selector*, string *value*, int *length*);
Vyvolá výjimku MQException.

Nastaví řetězec atributu.

selektor
Celé číslo označující, který atribut se nastavuje.

hodnota
Řetězec, který má být nastaven jako hodnota atributu.

délka
Celé číslo označující požadovanou délku řetězce.

Konstruktory

protected MQManagedObject()
Metoda konstruktoru. Tento objekt je abstraktní základní třída, kterou nelze převést na instanci sama.

Třída MQMessage.NET

Použijte MQMessage pro přístup k deskriptoru zprávy a datům pro zprávu IBM MQ . MQMessage zapouzdřuje zprávu IBM MQ .

Třída

```
System.Object
        |
        └─ IBM.WMQ.MQBase
                |
                └─ IBM.WMQ.MQBaseObject
                        |
                        └─ IBM.WMQ.MQMessage
```

```
public class IBM.WMQ.MQMessage extends IBM.WMQ.MQBaseObject;
```

Vytvořte objekt MQMessage a poté pomocí metod Read a Write přeneste data mezi zprávou a dalšími objekty ve vaší aplikaci. Odešlete a přijměte objekty MQMessage pomocí metod Put a Get tříd MQDestination, MQQueue a MQTopic .

Získejte a nastavte vlastnosti deskriptoru zprávy pomocí vlastností MQMessage. Nastavte a získejte rozšířené vlastnosti zprávy pomocí metod SetProperty a GetProperty .

- "Vlastnosti" na stránce 1729
- "Metody zpráv Read a Write" na stránce 1735
- "Metody vyrovnávací paměti" na stránce 1738
- "Metody vlastností" na stránce 1738
- "Konstruktory" na stránce 1741

Vlastnosti

Test pro vyvolání MQException při získávání vlastností.

public string AccountingToken {get; set;}
Část kontextu identity zprávy; pomáhá aplikaci účtovat za práci provedenou v důsledku zprávy. Výchozí hodnota je MQC.MQACT_NONE.

public string ApplicationIdData {get; set;}
Část kontextu identity zprávy. ApplicationIdData jsou informace, které jsou definovány sadou aplikací a lze je použít k poskytnutí dalších informací o zprávě nebo jejím původci. Výchozí hodnota je "".

public string ApplicationOriginData {get; set;}
Informace definované aplikací, které lze použít k poskytnutí dalších informací o původu zprávy. Výchozí hodnota je " ".

public int BackoutCount {get;}
Počet případů, kdy byla zpráva dříve vrácena a vrácena voláním funkce `MQQueue.Get` jako součást pracovní jednotky. Výchozí hodnota je 0.

public int CharacterSet {get; set;}

Identifikátor kódované znakové sady znakových dat ve zprávě.

Nastavte `CharacterSet` , abyste identifikovali znakovou sadu znakových dat ve zprávě. Získejte `CharacterSet` , abyste zjistili, která znaková sada byla použita ke kódování znakových dat ve zprávě.

Aplikace systému .NET jsou vždy spuštěny v kódování Unicode, zatímco v jiných prostředích jsou aplikace spouštěny ve stejné znakové sadě jako správce front.

Metody `ReadString` a `ReadLine` převádějí znaková data ve zprávě do kódování Unicode.

Metoda `WriteString` převádí z kódování Unicode na znakovou sadu zakódovanou v `CharacterSet`. Je-li volba `CharacterSet` nastavena na výchozí hodnotu `MQC.MQCCSI_Q_MGR`, což je 0, neprovede se žádný převod a `CharacterSet` je nastaveno na 1200. Pokud nastavíte `CharacterSet` na jinou hodnotu, `WriteString` převede z Unicode na alternativní hodnotu.

Poznámka: Jiné metody čtení a zápisu nepoužívají `CharacterSet`.

- `ReadChar` a `WriteChar` čtou a zapisují znaky Unicode do a z vyrovnávací paměti zpráv bez konverze.
- `ReadUTF` a `WriteUTF` převádějí mezi řetězcem Unicode v aplikaci a řetězcem UTF-8 , který má ve vyrovnávací paměti zpráv předponu v podobě dvoubajtového pole s délkou.
- Bajtové metody přenášejí bajty mezi aplikací a vyrovnávací pamětí zpráv bez změny.

public byte[] CorrelationId {get; set;}

- Pro volání `MQQueue.Get` se jedná o identifikátor korelace zprávy, která má být načtena. Správce front vrátí první zprávu s identifikátorem zprávy a identifikátorem korelace, který odpovídá polím deskriptoru zprávy. Výchozí hodnota `MQC.MQCI_NONE`pomáhá při porovnávání identifikátorů korelace.
- Pro volání `MQQueue.Put` se jedná o identifikátor korelace, který má být nastaven.

public int DataLength {get;}
Počet bajtů dat zprávy zbývajících ke čtení.

public int DataOffset {get; set;}
Aktuální pozice kurzoru v datech zprávy. Čtení a zápisy se projeví na aktuální pozici.

public int Encoding {get; set;}
Reprezentace použitá pro číselné hodnoty v datech zprávy aplikace. `Kódování` platí pro binární, pakovaná desetinná data a data s pohyblivou řádovou čárkou. Chování metod čtení a zápisu pro tyto číselné formáty je odpovídajícím způsobem změněno. Vytvořte hodnotu pro pole kódování přidáním jedné hodnoty z každé z těchto tří sekcí. Případně vytvořte hodnotu kombinující hodnoty z každé ze tří sekcí pomocí bitového operátoru OR.

1. Binární celé číslo

   MQC.MQENC_INTEGER_NORMAL
   Big-endian celá čísla.

   MQC.MQENC_INTEGER_REVERSED
   Celá čísla typu little-endian, použitá v architektuře Intel .

2. Pakované-desetinné

   MQC.MQENC_DECIMAL_NORMAL
   Big-endian packed-decimal, jak jej používá z/OS.

MQC.MQENC_DECIMAL_REVERSED
Malý-endian baleno-desetinné.

3. pohyblivá řádová čárka

MQC.MQENC_FLOAT_IEEE_NORMAL
Big-endian IEEE plave.

MQC.MQENC_FLOAT_IEEE_REVERSED
Little-endian IEEE floats, jak používá architekturu Intel .

MQC.MQENC_FLOAT_S390
z/OS formátovat plovoucí body.

Výchozí hodnota je:

```
MQC.MQENC_INTEGER_REVERSED |
MQC.MQENC_DECIMAL_REVERSED |
MQC.MQENC_FLOAT_IEEE_REVERSED
```

Výchozí nastavení způsobí, že `WriteInt` zapíše celé číslo typu little-endian a `ReadInt` přečte celé číslo typu little-endian. Nastavíte-li příznak `MQC.MQENC_INTEGER_NORMAL` , produkt `WriteInt` zapíše celé číslo typu big-endian a `ReadInt` přečte celé číslo typu big-endian.

Poznámka: Při převodu pohyblivých bodů formátu IEEE na pohyblivé body formátu zSeries může dojít ke ztrátě přesnosti.

public int Expiry {get; set;}
Doba vypršení platnosti vyjádřená v desetinách sekundy, nastavená aplikací, která vkládá zprávu. Po uplynutí doby vypršení platnosti zprávy je vhodné ji správce front zrušit. Pokud zpráva uvádí jeden z příznaků `MQC.MQRO_EXPIRATION` , vygeneruje se sestava, když je zpráva vyřazena. Výchozí hodnota je `MQC.MQEI_UNLIMITED`, což znamená, že platnost zprávy nikdy nevyprší.

public int Feedback {get; set;}
Pomocí volby Zpětná vazba se zprávou typu `MQC.MQMT_REPORT` můžete určit povahu sestavy. Systém definuje následující kódy zpětné vazby:

- `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`

Lze také použít hodnoty zpětné vazby definované aplikací v rozsahu `MQC.MQFB_APPL_FIRST` až `MQC.MQFB_APPL_LAST` . Výchozí hodnota tohoto pole je `MQC.MQFB_NONE`, což znamená, že není poskytnuta žádná zpětná vazba.

public string Format {get; set;}
Název formátu, který používá odesilatel zprávy k označení povahy dat ve zprávě pro příjemce. Můžete použít vlastní názvy formátů, ale názvy začínající písmeny MQ mají význam, který definuje správce front. Vestavěné formáty správce front jsou:

MQC.MQFMT_ADMIN
Zpráva požadavku/odpovědi příkazového serveru.

MQC.MQFMT_COMMAND_1
Napište 1 zprávu s odpovědí na příkaz.

MQC.MQFMT_COMMAND_2
Napište 2 zprávu s odpovědí na příkaz.

MQC.MQFMT_DEAD_LETTER_HEADER
Hlavička nedoručovací zprávy.

MQC.MQFMT_EVENT
Zpráva události.

MQC.MQFMT_NONE
Žádný název formátu.

MQC.MQFMT_PCF
Uživatelsky definovaná zpráva v programovatelném formátu příkazu.

MQC.MQFMT_STRING
Zpráva se skládá výhradně ze znaků.

MQC.MQFMT_TRIGGER
zpráva spouštěče

MQC.MQFMT_XMIT_Q_HEADER
Záhlaví přenosové fronty.

Výchozí hodnota je MQC.MQFMT_NONE.

public byte[] GroupId {get; set;}
Bajtový řetězec, který identifikuje skupinu zpráv, do které fyzická zpráva patří. Výchozí hodnota je MQC.MQGI_NONE.

public int MessageFlags {get; set;}
Příznaky řídící segmentaci a stav zprávy.

public byte[] MessageId {get; set;}

Pro volání MQQueue.Get toto pole uvádí identifikátor zprávy, která se má načíst. Správce front obvykle vrací první zprávu s identifikátorem zprávy a identifikátorem korelace, který odpovídá polím deskriptoru zprávy. Povolte, aby se jakýkoli identifikátor zprávy shodoval se speciální hodnotou MQC.MQMI_NONE.

Pro volání MQQueue.Put toto pole uvádí identifikátor zprávy, který se má použít. Je-li zadána hodnota MQC.MQMI_NONE , správce front při vložení zprávy vygeneruje jedinečný identifikátor zprávy. Hodnota této členské proměnné se aktualizuje po vložení, aby označovala použitý identifikátor zprávy. Výchozí hodnota je MQC.MQMI_NONE.

public int MessageLength {get;}
Počet bajtů dat zprávy v objektu MQMessage .

public int MessageSequenceNumber {get; set;}
Pořadové číslo logické zprávy ve skupině.

public int MessageType {get; set;}
Označuje typ zprávy. Následující hodnoty jsou momentálně definovány systémem:

- MQC.MQMT_DATAGRAM
- MQC.MQMT_REPLY
- MQC.MQMT_REPORT
- MQC.MQMT_REQUEST

Hodnoty definované aplikací lze také použít v rozsahu MQC.MQMT_APPL_FIRST až MQC.MQMT_APPL_LAST. Výchozí hodnota tohoto pole je MQC.MQMT_DATAGRAM.

public int Offset {get; set;}
V segmentované zprávě se jedná o posunutí dat ve fyzické zprávě od začátku logické zprávy.

public int OriginalLength {get; set;}
Původní délka segmentované zprávy.

public int Persistence {get; set;}

Perzistence zpráv. Jsou definovány tyto hodnoty:

- MQC.MQPER_NOT_PERSISTENT

  Nastavíte-li tuto volbu v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_NONE aplikaci při úspěšném připojení.

- MQC.MQPER_PERSISTENT

  Nastavíte-li tuto volbu v klientovi s možností opětovného připojení, po úspěšném připojení se aplikaci vrátí kód příčiny MQRC_CALL_INTERRUPTED .

- MQC.MQPER_PERSISTENCE_AS_Q_DEF

Výchozí hodnota je MQC.MQPER_PERSISTENCE_AS_Q_DEF, která vezme perzistenci pro zprávu z výchozího atributu perzistence cílové fronty.

public int Priority {get; set;}
Priorita zpráv. Speciální hodnotu MQC.MQPRI_PRIORITY_AS_Q_DEF lze také nastavit v odchozí zprávě. Priorita zprávy je pak převzata z atributu výchozí priority cílové fronty. Výchozí hodnota je MQC.MQPRI_PRIORITY_AS_Q_DEF.

public int PropertyValidation {get; set;}
Určuje, zda se má při nastavení vlastnosti zprávy provádět ověřování vlastností. Možné hodnoty jsou:

- MQCMHO_DEFAULT_VALIDATION
- MQCMHO_VALIDATE
- MQCMHO_NO_VALIDATION

Výchozí hodnota je MQCMHO_DEFAULT_VALIDATION.

public string PutApplicationName {get; set;}
Název aplikace, která vložila zprávu. Výchozí hodnota je "".

public int PutApplicationType {get; set;}
Typ aplikace vkládající zprávu. PutApplicationPutApplication může být hodnota definovaná systémem nebo uživatelem. Systém definuje následující hodnoty:

- 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

Výchozí hodnota je MQC.MQAT_NO_CONTEXT, což označuje, že ve zprávě nejsou žádné informace o kontextu.

public DateTime PutDateTime {get; set;}
Datum a čas vložení zprávy.

public string ReplyToQueueManagerName {get; set;}
Název správce front pro odesílání odpovědí nebo zpráv sestav. Výchozí hodnota je ""a správce front poskytuje ReplyToQueueManagersprávce front.

public string ReplyToQueueName {get; set;}
Název fronty zpráv, do které aplikace, která vydala požadavek na získání pro zprávu, odesílá zprávy MQC.MQMT_REPLY a MQC.MQMT_REPORT . Výchozí hodnota ReplyToQueueName je " ".

public int Report {get; set;}
Pomocí volby Sestava můžete určit volby týkající se zpráv sestav a odpovědí:

- Zda jsou vyžadovány sestavy.
- Zda mají být data zprávy aplikace zahrnuta do sestav.
- Jak nastavit identifikátory zprávy a korelace v sestavě nebo odpovědi.

Lze požadovat libovolnou kombinaci čtyř typů sestav:

- Zadejte libovolnou kombinaci čtyř typů sestav. Výběr libovolné ze tří voleb pro každý typ sestavy v závislosti na tom, zda mají být data zprávy aplikace zahrnuta do zprávy sestavy.
  1. Potvrdit při příjezdu
     - MQC.MQRO_COA
     - MQC.MQRO_COA_WITH_DATA
     - MQC.MQRO_COA_WITH_FULL_DATA **
  2. Potvrdit při doručení
     - MQC.MQRO_COD
     - MQC.MQRO_COD_WITH_DATA
     - MQC.MQRO_COD_WITH_FULL_DATA **
  3. Výjimka
     - MQC.MQRO_EXCEPTION
     - MQC.MQRO_EXCEPTION_WITH_DATA
     - MQC.MQRO_EXCEPTION_WITH_FULL_DATA **
  4. Konec platnosti
     - MQC.MQRO_EXPIRATION
     - MQC.MQRO_EXPIRATION_WITH_DATA
     - MQC.MQRO_EXPIRATION_WITH_FULL_DATA **

  Poznámka: Hodnoty označené v seznamu jako ** nejsou správci front z/OS podporovány. Nepoužívejte je, pokud je pravděpodobné, že vaše aplikace bude přistupovat ke správci front z/OS , bez ohledu na platformu, na které je aplikace spuštěna.
- Chcete-li řídit způsob generování ID zprávy pro zprávu sestavy nebo odpovědi, zadejte jednu z následujících možností:
  - MQC.MQRO_NEW_MSG_ID
  - MQC.MQRO_PASS_MSG_ID
- Chcete-li řídit, jak má být nastaveno ID korelace sestavy nebo zprávy odpovědi, zadejte jednu z následujících možností:
  - MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID
  - MQC.MQRO_PASS_CORREL_ID
- Chcete-li řídit dispozice původní zprávy v případě, že ji nelze doručit do cílové fronty, zadejte jednu z následujících možností:
  - MQC.MQRO_DEAD_LETTER_Q
  - MQC.MQRO_DISCARD_MSG **
- Pokud nejsou zadány žádné volby sestavy, předvolba je:

```
MQC.MQRO_NEW_MSG_ID |
MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID |
MQC.MQRO_DEAD_LETTER_Q
```

- Můžete uvést jednu nebo obě následující, chcete-li požadovat, aby přijímající aplikace odeslala pozitivní akci nebo negativní zprávu sestavy akcí.
  - `MQC.MQRO_PAN`
  - `MQC.MQRO_NAN`

public int TotalMessageLength {get;}
Celkový počet bajtů ve zprávě uložených ve frontě zpráv, ze které byla tato zpráva přijata.

public string UserId {get; set;}
`UserId` je součástí kontextu identity zprávy. Hodnotu obvykle poskytuje správce front. Hodnotu můžete přepsat, pokud máte oprávnění k nastavení kontextu identity.

public int Version {get; set;}
Verze používané struktury MQMD.

Metody zpráv Read a `Write`

Metody `Read` a `Write` provádějí stejné funkce jako členové tříd `BinaryReader` a `BinaryWriter` v oboru názvů .NET `System.IO` . Viz MSDN pro úplnou syntaxi jazyka a příklady použití. Metody čtení nebo zápisu z aktuální pozice ve vyrovnávací paměti zpráv. Posunou aktuální pozici dopředu o počet přečtených nebo zapsaných bajtů.

Poznámka: Pokud data zprávy obsahují záhlaví MQRFH nebo MQRFH2 , musíte ke čtení dat použít metodu `ReadBytes` .

- Všechny metody generují `IOException`.
- Metody `ReadFully` automaticky změní velikost cílového pole `byte` nebo `sbyte` tak, aby přesně odpovídala zprávě. Také se změní velikost pole s hodnotou null.
- `Read` metody throw `EndOfStreamException`.
- `WriteDecimal` metody throw `MQException`.
- Metody `ReadString`, `ReadLine` a `WriteString` se převádějí mezi kódováním Unicode a znakovou sadou zprávy; viz `CharacterSet` .
- Metody `Decimal` čtou a zapisují pakovaná desetinná čísla kódovaná ve formátu big-endian, `MQC.MQENC_DECIMAL_NORMAL`nebo little-endian `MQC.MQENC_DECIMAL_REVERSE` podle hodnoty `Encoding`. Desetinné rozsahy a odpovídající typy .NET jsou následující:

  Decimal2/short
  -999 až 999

  Decimal4/int
  -9999999 až 9999999

  Decimal8/long
  -999999999999999 až 999999999999999
- Metody `Double` a `Float` čtou a zapisují plovoucí hodnoty kódované ve formátech IEE big-endian a little-endian `MQC.MQENC_FLOAT_IEEE_NORMAL` a `MQC.MQENC_FLOAT_IEEE_REVERSED`nebo ve formátu S/390 `MQC.MQENC_FLOAT_S390`podle hodnoty `Encoding`.
- Metody `Int` čtou a zapisují celočíselné hodnoty kódované ve formátu big-endian, `MQC.MQENC_INTEGER_NORMAL`nebo little-endian, `MQC.MQENC_INTEGER_REVERSED`, podle hodnoty `Encoding`. Celá čísla jsou všechna podepsána, s výjimkou přidání dvoubajtového celočíselného typu bez znaménka. Velikosti celých čísel a typy .NET a IBM MQ jsou následující:

  2 bajty
  `short`, `Int2`, `ushort`, `UInt2`

4 bajtové
`int`, `Int4`

8 bajtů
`long`, `Int8`

- Produkt `WriteObject` přenese třídu objektu, hodnoty jeho nedočasných a nestatických polí a pole jejich supertypů do vyrovnávací paměti zpráv.
- Produkt `ReadObject` vytvoří objekt ze třídy objektu, podpisu třídy a hodnot jeho nedočasných a nestatických polí a polí jeho supertypů.

Tabulka 843. Metody čtení a zápisu zpráv

Typ cíle	Podpisy metod

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)`

Tabulka 843. Metody čtení a zápisu zpráv (pokračování)

Typ cíle	Podpisy metod
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)`

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()`

Tabulka 843. Metody čtení a zápisu zpráv (pokračování)	
Typ cíle	**Podpisy metod**

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)`

Metody vyrovnávací paměti

`public void ClearMessage();`

Vyvolá výjimku `IOException`.

Zruší všechna data ve vyrovnávací paměti zpráv a nastaví posunutí dat zpět na nulu.

`public void ResizeBuffer(int size)`

Vyvolá výjimku `IOException`.

Pokyn k objektu `MQMessage` o velikosti vyrovnávací paměti, která může být vyžadována pro následné operace získání. Pokud zpráva momentálně obsahuje data zprávy a nová velikost je menší než aktuální velikost, data zprávy se oříznou.

`public void Seek(int pos)`

Vygeneruje výjimku `IOException`, `ArgumentOutOfRangeException`, `ArgumentException`.

Přesune kurzor na absolutní pozici ve vyrovnávací paměti zpráv, kterou určuje hodnota *pos*. Následná čtení a zápisy působí na této pozici ve vyrovnávací paměti.

`public int SkipBytes(int i)`

Vygeneruje výjimku `IOException`, `EndOfStreamException`.

Přesune vpřed o n bajtů ve vyrovnávací paměti zpráv a vrátí hodnotu n, což je počet přeskočených bajtů.

Metoda `SkipBytes` blokuje, dokud nedojde k jedné z následujících událostí:

- Všechny bajty jsou vynechány
- Byl zjištěn konec vyrovnávací paměti zpráv.
- Došlo k výjimce

Metody vlastností

`public void DeleteProperty(string name);`

Vyvolá výjimku `MQException`.

Odstraní vlastnost s uvedeným názvem ze zprávy.

Název
Název vlastnosti, která se má odstranit.

public System.Collections.IEnumerator GetPropertyNames(string *name*)

Vyvolá výjimku MQException.

Vrátí IEnumerator všech názvů vlastností odpovídajících zadanému názvu. Znak procenta '%' lze použít na konci názvu jako zástupný znak pro filtrování vlastností zprávy, odpovídající nule nebo více znakům, včetně tečky.

Název
Název vlastnosti, pro kterou má být nalezena shoda.

Metody SetProperty a GetProperty

Všechny metody SetProperty a GetProperty generují MQException.

Metoda SetProperty třídy MQMessage .NET přidá novou vlastnost, pokud ještě vlastnost neexistuje. Pokud však tato vlastnost již existuje, bude zadaná hodnota vlastnosti přidána na konec seznamu. Pokud je více hodnot nastaveno na název vlastnosti pomocí SetProperty, volání GetProperty pro tento název vrátí tyto hodnoty postupně v pořadí, v jakém byly tyto hodnoty nastaveny.

Chování je stejné pro všechny metody typu Set*Property a Get*Property , například GetLongProperty, SetLongProperty, GetBooleanProperty, SetBooleanProperty, GetStringPropertya SetStringProperty.

Tabulka 844. Metody SetProperty a GetProperty

Typ	Podpisy metod

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);`

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);`

Tabulka 844. Metody SetProperty a GetProperty (pokračování)

Typ	Podpisy metod

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);`

Tabulka 844. Metody SetProperty a GetProperty (pokračování)	
Typ	**Podpisy metod**

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);`

Konstruktory

public MQMessage();

Vytvoří objekt MQMessage s výchozími informacemi o deskriptoru zpráv a prázdnou vyrovnávací pamětí zpráv.

Třída MQProcess.NET

Pomocí příkazu MQProcess se můžete dotázat na atributy procesu IBM MQ . Vytvořte objekt MQProcess pomocí konstruktoru nebo metody MQQueueManager AccessProcess .

Třída

```
System.Object
        |
        └─ IBM.WMQ.MQBase
                |
                └─ IBM.WMQ.MQBaseObject
                        |
                        └─ IBM.WMQ.MQManagedObject
                                |
                                └─ IBM.WMQ.MQProcess
```

`public class IBM.WMQ.MQProcess extends IBM.WMQ.MQManagedObject;`

- "Vlastnosti" na stránce 1741
- "Konstruktory" na stránce 1742

Vlastnosti

Test pro vyvolání MQException při získávání vlastností.

public string ApplicationId {get;}

Získá znakový řetězec, který identifikuje aplikaci, která se má spustit. ApplicationId používá aplikace monitoru spouštěčů. ApplicationId je odesláno do inicializační fronty jako součást zprávy spouštěče.

Výchozí hodnota je null.

public int ApplicationType {get;}

Identifikuje typ procesu, který má spustit aplikace monitoru spouštěčů. Standardní typy jsou definovány, ale jiné lze použít:

- MQAT_AIX
- MQAT_CICS
- MQAT_IMS
- MQAT_MVS

- MQAT_NATIVE
- MQAT_OS400
- MQAT_UNIX
- MQAT_WINDOWS
- MQAT_JAVA
- MQAT_USER_FIRST
- MQAT_USER_LAST

Výchozí hodnota je MQAT_NATIVE.

public string EnvironmentData {get;}

Získá informace o prostředí aplikace, která se má spustit.

Výchozí hodnota je null.

public string UserData {get;}

Získá informace, které uživatel poskytl o aplikaci, která se má spustit.

Výchozí hodnota je null.

Konstruktory

public MQProcess(MQQueueManager *queueManager*, string *processName*, int *openOptions*);
public MQProcess(MQQueueManager *qMgr*, string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

Vyvolá výjimku MQException.

Přístup k procesu IBM MQ ve správci front *qMgr* pro zjišťování atributů procesu.

qMgr
Správce front pro přístup.

processName
Název procesu, který se má otevřít.

openOptions
Volby, které řídí otevření procesu. Platné volby, které lze přidat nebo kombinovat pomocí bitové operace OR, jsou:

- MQC.MQOO_FAIL_IF_QUIESCING
- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

QueueManagerName
Název správce front, v němž je proces definován. Pokud je správce front stejný jako ten, ke kterému proces přistupuje, můžete ponechat prázdný název správce front nebo název správce front s hodnotou Null.

AlternateUserid

Je-li v parametru **openOptions** uveden parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY , *alternateUserId* uvádí alternativní ID uživatele použité ke kontrole autorizace pro akci. Není-li parametr MQOO_ALTERNATE_USER_AUTHORITY uveden, *alternateUserId* může být prázdný nebo mít hodnotu null.

Výchozí oprávnění uživatele se používá pro připojení ke správci front, není-li zadán parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY .

public MQProcess MQQueueManager.AccessProcess(string *processName*, int *openOptions*);
public MQProcess MQQueueManager.AccessProcess(string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

Vyvolá výjimku `MQException`.

Přístup k procesu IBM MQ v tomto správci front pro zjišťování atributů procesu.

processName
Název procesu, který se má otevřít.

openOptions
Volby, které řídí otevření procesu. Platné volby, které lze přidat nebo kombinovat pomocí bitové operace OR, jsou:

- `MQC.MQOO_FAIL_IF_QUIESCING`
- `MQC.MQOO_INQUIRE`
- `MQC.MQOO_SET`
- `MQC.MQOO_ALTERNATE_USER_AUTHORITY`

QueueManagerName
Název správce front, v němž je proces definován. Pokud je správce front stejný jako ten, ke kterému proces přistupuje, můžete ponechat prázdný název správce front nebo název správce front s hodnotou Null.

AlternateUserid

Je-li v parametru **openOptions** uveden parametr `MQC.MQOO_ALTERNATE_USER_AUTHORITY` , *`alternateUserId`* uvádí alternativní ID uživatele použité ke kontrole autorizace pro akci. Není-li parametr `MQOO_ALTERNATE_USER_AUTHORITY` uveden, *`alternateUserId`* může být prázdný nebo mít hodnotu null.

Výchozí oprávnění uživatele se používá pro připojení ke správci front, není-li zadán parametr `MQC.MQOO_ALTERNATE_USER_AUTHORITY` .

Třída MQPropertyDescriptor.NET

Použijte `MQPropertyDescriptor` jako parametr pro metody `MQMessage` `GetProperty` a `SetProperty` . `MQPropertyDescriptor` popisuje vlastnost `MQMessage` .

Třída

```
System.Object
        |
        └─ IBM.WMQ.MQPropertyDescriptor
```

`public class IBM.WMQ.MQPropertyDescriptor extends System.Object;`

- “Vlastnosti” na stránce 1743
- “Konstruktory” na stránce 1745

Vlastnosti

Test pro vyvolání `MQException` při získávání vlastností.

public int Context {get; set;}

Kontext zprávy, do kterého vlastnost patří. Možné hodnoty jsou:

MQC.MQPD_NO_CONTEXT
Vlastnost není přidružena ke kontextu zprávy.

MQC.MQPD_USER_CONTEXT
Vlastnost je přidružena ke kontextu uživatele.

Je-li uživatel autorizován, vlastnost přidružená ke kontextu uživatele se uloží při načtení zprávy. Následná metoda Put odkazující na uložený kontext může předat vlastnost do nové zprávy.

public int CopyOptions {get; set;}

CopyOptions popisuje, do kterého typu zprávy lze vlastnost zkopírovat.

Pokud správce front obdrží zprávu obsahující vlastnost definovanou v souboru IBM MQ , kterou správce front rozpozná jako nesprávnou, správce front opraví hodnotu pole CopyOptions .

Lze zadat libovolnou kombinaci následujících voleb. Zkombinujte volby přidáním hodnot nebo pomocí bitových hodnot OR.

MQC.MQCOPY_ALL
Vlastnost se zkopíruje do všech typů následných zpráv.

MQC.MQCOPY_FORWARD
Vlastnost se zkopíruje do předávané zprávy.

MQC.MQCOPY_PUBLISH
Vlastnost je zkopírována do zprávy přijaté odběratelem při publikování zprávy.

MQC.MQCOPY_REPLY
Vlastnost se zkopíruje do zprávy odpovědi.

MQC.MQCOPY_REPORT
Vlastnost se zkopíruje do zprávy sestavy.

MQC.MQCOPY_DEFAULT
Hodnota označená, že nebyly zadány žádné další volby kopírování. Mezi vlastností a následnými zprávami neexistuje žádný vztah. Pro vlastnosti deskriptoru zprávy je vždy vrácena hodnota MQC.MQCOPY_DEFAULT .

MQC.MQCOPY_NONE
Stejné jako MQC.MQCOPY_DEFAULT

public int Options { set; }
Volby standardně zobrazují CMQC.MQPD_NONE. Nelze nastavit žádnou jinou hodnotu.

public int Support { get; set; }

Nastavte volbu Podpora , chcete-li určit úroveň podpory požadovanou pro vlastnosti IBM MQ-defined message properties. Podpora všech ostatních vlastností je volitelná. Lze zadat libovolnou z následujících hodnot nebo žádnou z nich.

MQC.MQPD_SUPPORT_OPTIONAL
Vlastnost je přijata správcem front i v případě, že není podporována. Vlastnost lze zrušit tak, aby zpráva směřovala do správce front, který nepodporuje vlastnosti zprávy. Tato hodnota je také přiřazena k vlastnostem, které nejsou definovány IBM MQ .

MQC.MQPD_SUPPORT_REQUIRED
Podpora pro vlastnost je povinná. Pokud vložíte zprávu do správce front, který nepodporuje vlastnost IBM MQ-defined, metoda se nezdaří. Vrací kód dokončení MQC.MQCC_FAILED a kód příčiny MQC.MQRC_UNSUPPORTED_PROPERTY.

MQC.MQPD_SUPPORT_REQUIRED_IF_LOCAL

Podpora vlastnosti je povinná, pokud je zpráva určena pro lokální frontu. Pokud vložíte zprávu do lokální fronty ve správci front, který nepodporuje vlastnost IBM MQ-defined, metoda se nezdaří. Vrací kód dokončení MQC.MQCC_FAILED a kód příčiny MQC.MQRC_UNSUPPORTED_PROPERTY.

Pokud je zpráva vložena do vzdáleného správce front, není provedena žádná kontrola.

Konstruktory

PropertyDescriptor();
Vytvořte deskriptor vlastnosti.

Třída MQPutMessageOptions.NET

Pomocí volby MQPutMessageOptions můžete určit způsob odesílání zpráv. Upravuje chování MQDestination.Put.

Třída

```
System.Object
        │
        └─ IBM.WMQ.MQBase
                │
                └─ IBM.WMQ.MQBaseObject
                        │
                        └─ IBM.WMQ.MQPutMessageOptions
```

```
public class IBM.WMQ.MQPutMessageOptions extends IBM.WMQ.MQBaseObject;
```

- “Vlastnosti” na stránce 1745 “Konstruktory” na stránce 1747

Vlastnosti

Test pro vyvolání MQException při získávání vlastností.

Poznámka: Chování některých voleb dostupných v této třídě závisí na prostředí, ve kterém jsou použity. Tyto prvky jsou označeny hvězdičkou, *.

public MQQueue ContextReference {get; set;}

Pokud pole options obsahuje MQC.MQPMO_PASS_IDENTITY_CONTEXT nebo MQC.MQPMO_PASS_ALL_CONTEXT, nastavte toto pole tak, aby odkazovaly na MQQueue , ze kterého se mají převzít informace o kontextu.

Počáteční hodnota tohoto pole je null.

public int InvalidDestCount {get;} *

Obecně platí, že hodnota InvalidDestCount , která se používá pro distribuční seznamy, označuje počet zpráv, které nebylo možné odeslat do front v distribučním seznamu. Počet zahrnuje fronty, které se nepodařilo otevřít, a také fronty, které byly úspěšně otevřeny, ale pro které operace vložení selhala.

Produkt .NET nepodporuje distribuční seznamy, ale při otevírání jedné fronty je nastaven parametr InvalidDestCount .

public int KnownDestCount {get;} *

Obecně se používá pro distribuční seznamy, KnownDestKnownDest označuje počet zpráv, které aktuální volání úspěšně odeslalo do front, které se interpretují do lokálních front.

Produkt .NET nepodporuje distribuční seznamy, ale při otevírání jedné fronty je nastaven parametr InvalidDestCount .

public int Options {get; set;}

Volby, které řídí akci MQDestination.put a MQQueueManager.put. Lze zadat libovolnou nebo žádnou z následujících hodnot. Je-li požadována více než jedna volba, lze hodnoty přidat nebo zkombinovat pomocí bitové operátorky OR.

MQC.MQPMO_ASYNC_RESPONSE
Tato volba způsobí, že se volání MQDestination.put provede asynchronně s některými daty odezvy.

MQC.MQPMO_DEFAULT_CONTEXT
Přidružte výchozí kontext ke zprávě.

MQC.MQPMO_FAIL_IF_QUIESCING
Pokud je správce front uveden do klidového stavu, dojde k selhání.

MQC.MQPMO_LOGICAL_ORDER *
Vložte logické zprávy a segmenty ve skupinách zpráv do jejich logického pořadí.

Pokud použijete volbu MQPMO_LOGICAL_ORDER v opětovně připojitelném klientu, vrátí se aplikaci kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

MQC.MQPMO_NEW_CORREL_ID *
Vygenerujte nové ID korelace pro každou odeslanou zprávu.

MQC.MQPMO_NEW_MSG_ID *
Vygenerujte nové ID zprávy pro každou odeslanou zprávu.

MQC.MQPMO_NONE
Nejsou uvedeny žádné volby. Nepoužívejte s jinými volbami.

MQC.MQPMO_NO_CONTEXT
Ke zprávě nemá být přidružen žádný kontext.

MQC.MQPMO_NO_SYNCPOINT
Vložte zprávu bez řízení synchronizačního bodu. Není-li určena volba řízení synchronizačního bodu, předpokládá se výchozí hodnota bez synchronizačního bodu.

MQC.MQPMO_PASS_ALL_CONTEXT
Předejte veškerý kontext z manipulátoru vstupní fronty.

MQC.MQPMO_PASS_IDENTITY_CONTEXT
Předejte kontext identity z manipulátoru vstupní fronty.

MQC.MQPMO_RESPONSE_AS_Q_DEF

Pro volání MQDestination.put tato volba vezme typ odezvy vložení z atributu DEFPRESP fronty.

V případě volání MQQueueManager.put tato volba způsobí, že se volání provede synchronně.

MQC.MQPMO_RESPONSE_AS_TOPIC_DEF
MQC.MQPMO_RESPONSE_AS_TOPIC_DEF je synonymum pro MQC.MQPMO_RESPONSE_AS_Q_DEF pro použití s objekty tématu.

MQC.MQPMO_RETAIN

Odesílaná publikace má být zachována správcem front. Je-li použita tato volba a publikování nelze zachovat, zpráva se nepublikuje a volání selže s volbou MQC.MQRC_PUT_NOT_RETAINED.

Vyžádejte si kopii této publikace po jejím publikování voláním metody MQSubscription.RequestPublicationUpdate . Uložené publikování je odesláno aplikacím, které vytvoří odběr bez nastavení volby MQC.MQSO_NEW_PUBLICATIONS_ONLY . Zkontrolujte vlastnost zprávy MQIsRetained publikování, když je přijato, abyste zjistili, zda se jednalo o zachované publikování.

Pokud odběratel požaduje zachovaná publikování, může použitý odběr v řetězci tématu obsahovat zástupný znak. Pokud je ve stromu témat více zachovaných publikování, která odpovídají odběru, jsou všechna odeslána.

MQC.MQPMO_SET_ALL_CONTEXT
Nastavte celý kontext z aplikace.

MQC.MQPMO_SET_IDENTITY_CONTEXT
Nastavte kontext identity z aplikace.

MQC.MQPMO_SYNC_RESPONSE
Tato volba způsobí, že se volání MQDestination.put nebo MQQueueManager.put provede synchronně s úplnými daty odezvy.

MQC.MQPMO_SUPPRESS_REPLYTO
Odběratelům se nepředávají žádné informace vyplněné do polí ReplyToQueueName a ReplyToQueueManagersprávce front . Pokud je tato volba použita v kombinaci s volbou sestavy, která vyžaduje ReplyToQueueName, volání se nezdaří s hodnotou MQC.MQRC_MISSING_REPLY_TO_Q.

MQC.MQPMO_SYNCPOINT
Vložte zprávu s ovládacím prvkem synchronizačního bodu. Zpráva není viditelná mimo jednotku práce, dokud není jednotka práce potvrzena. Pokud je jednotka práce odvolána, zpráva se odstraní.

public int RecordFields {get; set;}*

Informace o distribučních seznamech. Distribuční seznamy nejsou v produktu .NETpodporovány.

public string ResolvedQueueManagerName {get;}

Výstupní pole nastavené správcem front na název správce front, který vlastní frontu určenou názvem vzdálené fronty. ResolvedQueueManagerName se může lišit od názvu správce front, ze kterého byla fronta zpřístupněna, pokud se jedná o vzdálenou frontu.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou. Je-li objektem distribuční seznam nebo téma, vrácená hodnota není definována.

public string ResolvedQueueName {get;}

Výstupní pole nastavené správcem front na název fronty, do které je zpráva umístěna. ResolvedQueueName se může lišit od názvu použitého k otevření fronty, pokud byla otevřená fronta aliasem nebo modelovou frontou.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou. Je-li objektem distribuční seznam nebo téma, vrácená hodnota není definována.

public int UnknownDestCount {get;}*

Obecně se používá pro distribuční seznamy, UnknownDestCount je výstupní pole nastavené správcem front. Uvádí počet zpráv, které aktuální volání úspěšně odeslalo do front, které se interpretují do vzdálených front.

Produkt .NET nepodporuje distribuční seznamy, ale při otevírání jedné fronty je nastaven parametr InvalidDestCount .

Konstruktory

public MQPutMessageOptions();

Vytvořte nový objekt MQPutMessageOptions bez nastavených voleb a prázdný objekt ResolvedQueueName a ResolvedQueueManagerName.

Třída MQQueue.NET

Pomocí produktu MQQueue můžete odesílat a přijímat zprávy a dotazovat se na atributy fronty IBM MQ . Vytvořte objekt MQQueue pomocí konstruktoru nebo metody MQQueueManager.AccessProcess .

Třída

```
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;
```


- "Vlastnosti" na stránce 1748
- "metody" na stránce 1750
- "Konstruktory" na stránce 1752

Vlastnosti

Test pro vyvolání `MQException` při získávání vlastností.

public int ClusterWorkLoadPriority {get;}
Určuje prioritu fronty. Tento parametr je platný pouze pro lokální, vzdálené a alias fronty.

public int ClusterWorkLoadRank {get;}
Určuje pořadí fronty. Tento parametr je platný pouze pro lokální, vzdálené a alias fronty.

public int ClusterWorkLoadUseQ {get;}
Určuje chování operace MQPUT v případě, že cílová fronta obsahuje lokální instanci a alespoň jednu vzdálenou instanci klastru. Tento parametr se nepoužije, pokud MQPUT pochází z kanálu klastru. Tento parametr je platný pouze pro lokální fronty.

public DateTime CreationDateTime {get;}
Datum a čas vytvoření této fronty.

public int CurrentDepth {get;}
Získá počet zpráv, které jsou momentálně ve frontě. Tato hodnota se zvýší během volání put a během zpětného volání get. Je dekrementován během non-browse get a během backout volání put.

public int DefinitionType {get;}
Jak byla fronta definována. Možné hodnoty jsou:

- `MQC.MQQDT_PREDEFINED`
- `MQC.MQQDT_PERMANENT_DYNAMIC`
- `MQC.MQQDT_TEMPORARY_DYNAMIC`

public int InhibitGet {get; set;}
Řídí, zda můžete získat zprávy v této frontě nebo pro toto téma. Možné hodnoty jsou:

- `MQC.MQQA_GET_INHIBITED`
- `MQC.MQQA_GET_ALLOWED`

public int InhibitPut {get; set;}
Určuje, zda lze do této fronty nebo pro toto téma vkládat zprávy. Možné hodnoty jsou:

- `MQQA_PUT_INHIBITED`
- `MQQA_PUT_ALLOWED`

public int MaximumDepth {get;}
Maximální počet zpráv, které mohou současně existovat ve frontě. Pokus o vložení zprávy do fronty, která již obsahuje tento počet zpráv, selže s kódem příčiny `MQC.MQRC_Q_FULL`.

public int MaximumMessageLength {get;}
Maximální délka dat aplikace, která mohou existovat v každé zprávě v této frontě. Pokus o vložení zprávy větší než tato hodnota se nezdaří s kódem příčiny `MQC.MQRC_MSG_TOO_BIG_FOR_Q`.

public int NonPersistentMessageClass {get;}
Úroveň spolehlivosti pro dočasné zprávy vložené do této fronty.

public int OpenInputCount {get;}
Počet popisovačů, které jsou momentálně platné pro odebrání zpráv z fronty. `OpenInputOpenInput` je celkový počet platných vstupních popisovačů známých lokálnímu správci front, nikoli pouze popisovačů vytvořených aplikací.

public int OpenOutputCount {get;}
Počet popisovačů, které jsou momentálně platné pro přidání zpráv do fronty. `OpenOutput` je celkový počet platných manipulátorů výstupu známých lokálnímu správci front, nikoli pouze manipulátorů vytvořených aplikací.

public int QueueAccounting {get;}
Uvádí, zda můžete povolit shromažďování informací o účtování pro frontu.

public int QueueMonitoring {get;}
Uvádí, zda můžete povolit monitorování pro frontu.

public int QueueStatistics {get;}
Určuje, zda lze povolit shromažďování statistických údajů pro frontu.

public int QueueType {get;}
Typ této fronty s jednou z následujících hodnot:

- `MQC.MQQT_ALIAS`
- `MQC.MQQT_LOCAL`
- `MQC.MQQT_REMOTE`
- `MQC.MQQT_CLUSTER`

public int Shareability {get;}
Zda lze frontu otevřít pro vstup vícekrát. Možné hodnoty jsou:

- `MQC.MQQA_SHAREABLE`
- `MQC.MQQA_NOT_SHAREABLE`

public string TPIPE {get;}
Název `TPIPE` použitý pro komunikaci s OTMA pomocí mostu IBM MQ IMS .

public int TriggerControl {get; set;}
Zda se zprávy spouštěče zapisují do inicializační fronty, aby se spustila aplikace pro obsluhu fronty. Možné hodnoty jsou:

- `MQC.MQTC_OFF`
- `MQC.MQTC_ON`

public string TriggerData {get; set;}
Data ve volném formátu, která správce front vloží do zprávy spouštěče. Vloží `TriggerData` , když zpráva přicházející do této fronty způsobí, že zpráva spouštěče bude zapsána do inicializační fronty. Maximální přípustná délka řetězce je dána hodnotou `MQC.MQ_TRIGGER_DATA_LENGTH`.

public int TriggerDepth {get; set;}
Počet zpráv, které musí být ve frontě, než se zapíše zpráva spouštěče, když je typ spouštěče nastaven na `MQC.MQTT_DEPTH`.

public int TriggerMessagePriority {get; set;}
Priorita zprávy, pod kterou zprávy nepřispívají ke generování zpráv spouštěče. To znamená, že správce front ignoruje tyto zprávy při rozhodování, zda generovat spouštěč. Hodnota nula způsobí, že všechny zprávy přispějí ke generování zpráv spouštěče.

public int TriggerType {get; set;}
Podmínky, za kterých jsou zprávy spouštěče zapisovány jako výsledek zpráv přicházejících do této fronty. Možné hodnoty jsou:

- `MQC.MQTT_NONE`
- `MQC.MQTT_FIRST`

- MQC.MQTT_EVERY
- MQC.MQTT_DEPTH

metody

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);

Vyvolá výjimku MQException.

Získá zprávu z fronty.

Pokud se příkaz get nezdaří, objekt MQMessage se nezmění. V případě úspěchu jsou deskriptor zprávy a datové části zprávy MQMessage nahrazeny deskriptorem zprávy a daty zprávy z příchozí zprávy.

Všechna volání IBM MQ z určitého MQQueueManager jsou synchronní. Proto, pokud provedete operaci get s wait, všechny ostatní podprocesy používající stejný MQQueueManager budou blokovány pro další volání IBM MQ , dokud nebude volání Get dokončeno. Potřebujete-li pro přístup k produktu IBM MQ současně více podprocesů, musí každý podproces vytvořit svůj vlastní objekt MQQueueManager .

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá pole v deskriptoru zprávy jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry MessageId a CorrelationId byly nastaveny podle potřeby.

Klient s možností opětovného připojení vrátí kód příčiny MQRC_BACKED_OUT po úspěšném opětovném připojení pro zprávy přijaté pod MQGM_SYNCPOINT.

Volby getMessage
Volby, které řídí akci získání.

Použití volby MQC.MQGMO_CONVERT může vést k výjimce s kódem příčiny MQC.MQRC_CONVERTED_STRING_TOO_BIG při převodu z jednobajtových znakových kódů na dvoubajtové kódy. V tomto případě se zpráva zkopíruje do vyrovnávací paměti bez převodu.

Není-li parametr *getMessageOptions* uveden, použije se volba zprávy MQGMO_NOWAIT.

Pokud použijete volbu MQGMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Velikost MaxMsg
Největší zpráva, kterou má tento objekt zprávy přijmout. Pokud je zpráva ve frontě větší než tato velikost, dojde k jedné ze dvou věcí:

- Je-li v objektu MQGetMessageOptions nastaven příznak MQGMO_ACCEPT_TRUNCATED_MSG , zpráva se vyplní co nejvíce dat zprávy. Dojde k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_ACCEPTED .
- Není-li příznak MQGMO_ACCEPT_TRUNCATED_MSG nastaven, zpráva zůstane ve frontě. Dojde k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_FAILED .

Není-li zadána hodnota *MaxMsgSize* , bude načtena celá zpráva.

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

Vyvolá výjimku MQException.

Vloží zprávu do fronty.

Úpravy objektu MQMessage po provedení volání Put neovlivní skutečnou zprávu ve frontě IBM MQ nebo tématu publikování.

Produkt Put aktualizuje vlastnosti MessageId a CorrelationId objektu MQMessage a nevymaže data zprávy. Další volání Put nebo Get odkazují na aktualizované informace v objektu MQMessage . Například v následujícím úseku kódu obsahuje první zpráva a a druhou ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

zpráva
Objekt MQMessage obsahující data deskriptoru zprávy a zprávu, která má být odeslána. Deskriptor zprávy může být změněn v důsledku této metody. Hodnoty v deskriptoru zprávy bezprostředně po dokončení této metody jsou hodnoty, které byly vloženy do fronty nebo publikovány do tématu.

Klientovi s možností opětovného připojení jsou vráceny následující kódy příčiny:

- MQRC_CALL_INTERRUPTED , pokud je připojení přerušeno při spuštění volání Put pro trvalou zprávu a opětovné připojení je úspěšné.
- MQRC_NONE je-li připojení úspěšné při spuštění volání Put pro dočasnou zprávu (viz Obnova aplikace).

Volby putMessage
Volby, které řídí akci vložení.

Není-li parametr *putMessageOptions* uveden, použije se výchozí instance MQPutMessageOptons .

Pokud použijete volbu MQPMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Poznámka: Chcete-li pro jednoduchost a výkon vložit jednu zprávu do fronty, použijte objekt MQQueueManager.Put . Pro tento účel byste měli mít objekt MQQueue .

public void PutForwardMessage(MQMessage *message*);
public void PutForwardMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

Vyvolá MQException

Vložte zprávu, která je předána do fronty, kde *message* je původní zpráva.

zpráva
Objekt MQMessage obsahující data deskriptoru zprávy a zprávu, která má být odeslána. Deskriptor zprávy může být změněn v důsledku této metody. Hodnoty v deskriptoru zprávy bezprostředně po dokončení této metody jsou hodnoty, které byly vloženy do fronty nebo publikovány do tématu.

Klientovi s možností opětovného připojení jsou vráceny následující kódy příčiny:

- MQRC_CALL_INTERRUPTED , pokud je připojení přerušeno při spuštění volání Put pro trvalou zprávu a opětovné připojení je úspěšné.
- MQRC_NONE je-li připojení úspěšné při spuštění volání Put pro dočasnou zprávu (viz Obnova aplikace).

Volby putMessage
Volby, které řídí akci vložení.

Není-li parametr *putMessageOptions* uveden, použije se výchozí instance MQPutMessageOptons .

Pokud použijete volbu MQPMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

public void PutReplyMessage(MQMessage *message*)
public void PutReplyMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*)

Vyvolá výjimku MQException.

Vložte zprávu s odpovědí do fronty, kde *message* je původní zpráva.

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá pole v deskriptoru zprávy jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry MessageId a CorrelationId byly nastaveny podle potřeby.

Klient s možností opětovného připojení vrátí kód příčiny MQRC_BACKED_OUT po úspěšném opětovném připojení pro zprávy přijaté pod MQGM_SYNCPOINT.

Volby putMessage
Volby, které řídí akci vložení.

Není-li parametr *putMessageOptions* uveden, použije se výchozí instance MQPutMessageOptons .

Pokud použijete volbu MQPMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

public void PutReportMessage(MQMessage *message*)
public void PutReportMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*)

Vyvolá výjimku MQException.

Vložte zprávu sestavy do fronty, kde *message* je původní zpráva.

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá pole v deskriptoru zprávy jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry MessageId a CorrelationId byly nastaveny podle potřeby.

Klient s možností opětovného připojení vrátí kód příčiny MQRC_BACKED_OUT po úspěšném opětovném připojení pro zprávy přijaté pod MQGM_SYNCPOINT.

Volby putMessage
Volby, které řídí akci vložení.

Není-li parametr *putMessageOptions* uveden, použije se výchozí instance MQPutMessageOptons .

Pokud použijete volbu MQPMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Konstruktory

public MQQueue MQQueueManager.AccessQueue(string *queueName*, int *openOptions*);
public MQQueue MQQueueManager.AccessQueue(string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

Vyvolá výjimku MQException.

Přistupuje k frontě v tomto správci front.

Můžete získat nebo procházet zprávy, vkládat zprávy, dotazovat se na atributy fronty nebo nastavit atributy fronty. Pokud je pojmenovaná fronta modelovou frontou, vytvoří se dynamická lokální fronta. Chcete-li zjistit název dynamické fronty, zadejte dotaz na atribut name výsledného objektu MQQueue .

queueName
Název fronty, která se má otevřít.

openOptions
Volby, které řídí otevření fronty.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Ověřit s uvedeným identifikátorem uživatele.

MQC.MQOO_BIND_AS_QDEF
Použít výchozí vazbu pro frontu.

MQC.MQOO_BIND_NOT_FIXED
Nepřipojujte se ke specifickému cíli.

MQC.MQOO_BIND_ON_OPEN
Svázat manipulátor s cílem při otevření fronty.

MQC.MQOO_BROWSE
Otevřít pro procházení zpráv.

MQC.MQOO_FAIL_IF_QUIESCING
Pokud je správce front uveden do klidového stavu, dojde k selhání.

MQC.MQOO_INPUT_AS_Q_DEF
Otevřít pro získání zpráv pomocí výchozího nastavení definovaného frontou.

MQC.MQOO_INPUT_SHARED
Otevřít pro získání zpráv se sdíleným přístupem.

MQC.MQOO_INPUT_EXCLUSIVE
Otevřít pro získání zpráv s výlučným přístupem.

MQC.MQOO_INQUIRE
Otevřít pro dotaz-povinné, pokud chcete dotazovat vlastnosti.

MQC.MQOO_OUTPUT
Otevřít pro vložení zpráv.

MQC.MQOO_PASS_ALL_CONTEXT
Povolit předání všech kontextů.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Povolit předání kontextu identity.

MQC.MQOO_SAVE_ALL_CONTEXT
Uložit kontext při načtení zprávy.

MQC.MQOO_SET
Otevřít pro nastavení atributů-povinné, pokud chcete nastavit vlastnosti.

MQC.MQOO_SET_ALL_CONTEXT
Umožňuje nastavení všech kontextů.

MQC.MQOO_SET_IDENTITY_CONTEXT
Umožňuje nastavení kontextu identity.

QueueManagerName
Název správce front, v němž je fronta definována. Název, který je zcela prázdný nebo má hodnotu null, označuje správce front, ke kterému je objekt `MQQueueManager` připojen.

Název dynamicQueue
Parametr *dynamicQueueName* je ignorován, pokud `queueName` neuvádí název modelové fronty. Pokud ano, *dynamicQueueName* uvádí název dynamické fronty, která se má vytvořit. Prázdný název nebo název s hodnotou null není platný, pokud `queueName` uvádí název modelové fronty. Pokud je posledním neprázdným znakem v názvu hvězdička `*`, správce front nahradí hvězdičku řetězcem znaků. Znaky zaručují, že název vygenerovaný pro frontu je v tomto správci front jedinečný.

AlternateUserid
Je-li v parametru `openOptions` uveden parametr `MQC.MQOO_ALTERNATE_USER_AUTHORITY`, *alternateUserId* uvádí alternativní identifikátor uživatele, který se používá ke kontrole autorizace pro otevření. Není-li parametr `MQC.MQOO_ALTERNATE_USER_AUTHORITY` uveden, *alternateUserId* může být ponechán prázdný, nebo může mít hodnotu null.

public MQQueue(MQQueueManager *queueManager*, string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

Vyvolá výjimku `MQException`.

Přistupuje k frontě v systému `queueManager`.

Můžete získat nebo procházet zprávy, vkládat zprávy, dotazovat se na atributy fronty nebo nastavit atributy fronty. Pokud je pojmenovaná fronta modelovou frontou, vytvoří se dynamická lokální fronta. Chcete-li zjistit název dynamické fronty, zadejte dotaz na atribut name výsledného objektu MQQueue .

queueManager
Správce front pro přístup k frontě.

queueName
Název fronty, která se má otevřít.

openOptions
Volby, které řídí otevření fronty.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Ověřit s uvedeným identifikátorem uživatele.

MQC.MQOO_BIND_AS_QDEF
Použít výchozí vazbu pro frontu.

MQC.MQOO_BIND_NOT_FIXED
Nepřipojujte se ke specifickému cíli.

MQC.MQOO_BIND_ON_OPEN
Svázat manipulátor s cílem při otevření fronty.

MQC.MQOO_BROWSE
Otevřít pro procházení zpráv.

MQC.MQOO_FAIL_IF_QUIESCING
Pokud je správce front uveden do klidového stavu, dojde k selhání.

MQC.MQOO_INPUT_AS_Q_DEF
Otevřít pro získání zpráv pomocí výchozího nastavení definovaného frontou.

MQC.MQOO_INPUT_SHARED
Otevřít pro získání zpráv se sdíleným přístupem.

MQC.MQOO_INPUT_EXCLUSIVE
Otevřít pro získání zpráv s výlučným přístupem.

MQC.MQOO_INQUIRE
Otevřít pro dotaz-povinné, pokud chcete dotazovat vlastnosti.

MQC.MQOO_OUTPUT
Otevřít pro vložení zpráv.

MQC.MQOO_PASS_ALL_CONTEXT
Povolit předání všech kontextů.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Povolit předání kontextu identity.

MQC.MQOO_SAVE_ALL_CONTEXT
Uložit kontext při načtení zprávy.

MQC.MQOO_SET
Otevřít pro nastavení atributů-povinné, pokud chcete nastavit vlastnosti.

MQC.MQOO_SET_ALL_CONTEXT
Umožňuje nastavení všech kontextů.

MQC.MQOO_SET_IDENTITY_CONTEXT
Umožňuje nastavení kontextu identity.

QueueManagerName
Název správce front, v němž je fronta definována. Název, který je zcela prázdný nebo má hodnotu null, označuje správce front, ke kterému je objekt MQQueueManager připojen.

Název dynamicQueue
Parametr *dynamicQueueName* je ignorován, pokud queueName neuvádí název modelové fronty. Pokud ano, *dynamicQueueName* uvádí název dynamické fronty, která se má vytvořit. Prázdný název nebo název s hodnotou null není platný, pokud queueName uvádí název modelové fronty.

Pokud je posledním neprázdným znakem v názvu hvězdička *, správce front nahradí hvězdičku řetězcem znaků. Znaky zaručují, že název vygenerovaný pro frontu je v tomto správci front jedinečný.

AlternateUserid
Je-li v parametru openOptions uveden parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY , *alternateUserId* uvádí alternativní identifikátor uživatele, který se používá ke kontrole autorizace pro otevření. Není-li parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY uveden, *alternateUserId* může být ponechán prázdný, nebo může mít hodnotu null.

Třída MQQueueManager.NET

Pomocí produktu MQQueueManager se můžete připojit ke správci front a přistupovat k objektům správce front. Řídí také transakce. Konstruktor MQQueueManager vytvoří připojení klienta nebo serveru.

Třída

```
System.Object
        |
        └─ IBM.WMQ.MQBase
                |
                └─ IBM.WMQ.MQBaseObject
                        |
                        └─ IBM.WMQ.ManagedObject
                                |
                                └─ IBM.WMQ.MQQueueManager
```

```
public class IBM.WMQ.MQQueueManager extends IBM.WMQ.MQManagedObject;
```

- “Vlastnosti” na stránce 1755
- “metody” na stránce 1758
- “Konstruktory” na stránce 1764

Vlastnosti

Test pro vyvolání MQException při získávání vlastností.

public int AccountingConnOverride {get;}
Zda mohou aplikace přepsat nastavení hodnot evidence MQI a evidence front.

public int AccountingInterval {get;}
Jak dlouho před zápisem mezilehlých účetních záznamů (v sekundách).

public int ActivityRecording {get;}
Ovládá generování sestav aktivity.

public int AdoptNewMCACheck {get;}
Určuje, které prvky jsou kontrolovány, aby se určilo, zda je agent MCA převzat při zjištění nového příchozího kanálu. Aby mohl být převzat, musí název agenta MCA odpovídat názvu aktivního agenta MCA.

public int AdoptNewMCAInterval {get;}
Doba v sekundách, po kterou nový kanál čeká na ukončení osiřelého kanálu.

public int AdoptNewMCAType {get;}
Zda má být osamocená instance MCA převzata (restartována), když je zjištěn nový příchozí požadavek kanálu odpovídající hodnotě AdoptNewMCACheck.

public int BridgeEvent {get;}
Zda jsou generovány události mostu IMS .

public int ChannelEvent {get;}
Zda jsou generovány události kanálu.

public int ChannelInitiatorControl {get;}
Určuje, zda se iniciátor kanálu spustí automaticky při spuštění správce front.

public int ChannelInitiatorAdapters {get;}
Počet dílčích úloh adaptéru ke zpracování volání IBM MQ .

public int ChannelInitiatorDispatchers {get;}
Počet dispečerů, které mají být použity pro inicializátor kanálu.

public int ChannelInitiatorTraceAutoStart {get;}
Určuje, zda se trasování inicializátoru kanálu spustí automaticky.

public int ChannelInitiatorTraceTableSize {get;}
Velikost prostoru trasovacích dat inicializátoru kanálu v megabajtech.

public int ChannelMonitoring {get;}
Zda se používá monitorování kanálu.

public int ChannelStatistics {get;}
Ovládá shromažďování statistických dat kanály.

public int CharacterSet {get;}
Vrací identifikátor kódované znakové sady (CCSID) správce front. `CharacterSet` je používán správcem front pro všechna pole znakových řetězců v rozhraní API.

public int ClusterSenderMonitoring {get;}
Řídí shromažďování online dat monitorování pro automaticky definované odesílací kanály klastru.

public int ClusterSenderStatistics {get;}
Řídí shromažďování statistických dat pro automaticky definované odesílací kanály klastru.

public int ClusterWorkLoadMRU {get;}
Maximální počet odchozích kanálů klastru.

public int ClusterWorkLoadUseQ {get;}
Výchozí hodnota vlastnosti `MQQueue ClusterWorkLoadUseQ`, pokud uvádí hodnotu `QMGR`.

public int CommandEvent {get;}
Uvádí, zda jsou generovány události příkazů.

public string CommandInputQueueName {get;}
Vrací název vstupní fronty příkazů definované ve správci front. Aplikace mohou odesílat příkazy do této fronty, pokud k tomu mají oprávnění.

public int CommandLevel {get;}
Označuje úroveň funkce správce front. Sada funkcí, které odpovídají konkrétní úrovni funkce, závisí na platformě. Na konkrétní platformě se můžete spolehnout na každého správce front, který podporuje funkce na nejnižší funkční úrovni společné pro všechny správce front.

public int CommandLevel {get;}
Určuje, zda se příkazový server spustí automaticky při spuštění správce front.

public string DNSGroup {get;}
Již se nepoužívá.

public int DNSWLM {get;}
Již se nepoužívá.

public int IPAddressVersion {get;}
Který protokol IP (IPv4 nebo IPv6) se má použít pro připojení kanálu.

public boolean IsConnected {get;}
Vrací hodnotu `isConnected`.

Pokud je hodnota nastavena na true, bylo vytvořeno připojení ke správci front a není známo, že by bylo přerušeno. Jakákoli volání IsConnected se aktivně nepokoušejí kontaktovat správce front, takže je možné, že se fyzická konektivita může přerušit, ale IsConnected může přesto vrátit hodnotu true. Stav IsConnected je aktualizován pouze v případě, že je ve správci front provedena aktivita, například vložení zprávy, získání zprávy.

Pokud je hodnota nastavena na false, připojení ke správci front nebylo vytvořeno, bylo přerušeno nebo bylo odpojeno.

public int KeepAlive {get;}
Uvádí, zda se má prostředek KEEPALIVE protokolu TCP použít ke kontrole, zda je druhý konec připojení stále k dispozici. Pokud není k dispozici, kanál se zavře.

public int ListenerTimer {get;}
Časový interval v sekundách mezi pokusy produktu IBM MQ o restartování modulu listener po selhání APPC nebo TCP/IP.

public int LoggerEvent {get;}
Zda jsou generovány události modulu protokolování.

public string LU62ARMSuffix {get;}
Přípona člena APPCPM SYS1.PARMLIB. Tato přípona určuje LUADD pro tento inicializátor kanálu. Když správce automatického restartu (ARM) restartuje inicializátor kanálu, je vydán příkaz z/OS SET APPC=xx.

public string LUGroupName {get; z/os}
Generické jméno LU, které má používat modul listener 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front.

public string LUName {get;}
Název LU, která má být použita pro odchozí přenosy LU 6.2 .

public int MaximumActiveChannels {get;}
Maximální počet kanálů, které mohou být současně aktivní.

public int MaximumCurrentChannels {get;}
Maximální počet kanálů, které mohou být kdykoli aktuální (včetně kanálů připojení serveru s připojenými klienty).

public int MaximumLU62Channels {get;}
Maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni a kteří používají přenosový protokol LU 6.2 .

public int MaximumMessageLength {get;}
Vrací maximální délku zprávy (v bajtech), kterou může správce front zpracovat. Nelze definovat žádnou frontu s maximální délkou zprávy větší než `MaximumMessageLength`.

public int MaximumPriority {get;}
Vrací maximální prioritu zpráv podporovanou správcem front. Priority jsou v rozsahu od nuly (nejnižší) do této hodnoty. Vyvolá metodu `MQException` , pokud tuto metodu zavoláte po odpojení od správce front.

public int MaximumTCPChannels {get;}
Maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni, kteří používají přenosový protokol TCP/IP.

public int MQIAccounting {get;}
Ovládá shromažďování informací o účtu pro data MQI.

public int MQIStatistics {get;}
Ovládá shromažďování informací o monitorování statistiky pro správce front.

public int OutboundPortMax {get;}
Maximální hodnota v rozsahu čísel portů, která má být použita při vázání odchozích kanálů.

public int OutboundPortMin {get;}
Minimální hodnota v rozsahu čísel portů, která má být použita při vázání odchozích kanálů.

public int QueueAccounting {get;}
Zda se mají data evidence třídy 3 (evidence na úrovni podprocesů a evidence na úrovni fronty) použít pro všechny fronty.

public int QueueMonitoring {get;}
Ovládá shromažďování online monitorovacích dat pro fronty.

public int QueueStatistics {get;}
Ovládá shromažďování statistických dat pro fronty.

public int ReceiveTimeout {get;}
Doba, po kterou kanál TCP/IP čeká na přijetí dat, včetně prezenčních signálů, od svého partnera před návratem do neaktivního stavu.

public int ReceiveTimeoutMin {get;}
Minimální doba, po kterou kanál TCP/IP čeká na přijetí dat, včetně prezenčních signálů, od svého partnera, než se vrátí do neaktivního stavu.

public int ReceiveTimeoutType {get;}
Kvalifikátor, který se má použít na hodnotu v poli `ReceiveTimeout`.

public int SharedQueueQueueManagerName {get;}
Určuje, jak doručit zprávy do sdílené fronty. Pokud vložení uvádí jiného správce front ze stejné skupiny sdílení front jako cílový správce front, zpráva se doručí dvěma způsoby:

MQC.MQSQQM_USE
Zprávy jsou doručeny do správce front objektů před jejich vložením do sdílené fronty.

MQCMQSQQM_IGNORE
Zprávy jsou vkládány přímo do sdílené fronty.

public int SSLEvent {get;}
Zda jsou generovány události TLS.

public int SSLFips {get;}
Zda se mají použít pouze algoritmy certifikované FIPS, pokud se šifrování provádí v produktu IBM MQ, spíše než kryptografický hardware.

public int SSLKeyResetCount {get;}
Označuje počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS, než je znovu vyjednán tajný klíč.

public int ClusterSenderStatistics {get;}
Určuje interval (v minutách) mezi následnými shromažďeními statistiky.

public int SyncpointAvailability {get;}
Označuje, zda správce front podporuje pracovní jednotky a synchronizační body pomocí metod `MQQueue.get` a `MQQueue.put` .

public string TCPName {get;}
Název jediného nebo výchozího systému TCP/IP, který se má použít, v závislosti na hodnotě TCPStackType.

public int TCPStackType {get;}
Určuje, zda iniciátor kanálu používá pouze adresní prostor TCP/IP uvedený v poli `TCPName`. Alternativně může být inicializátor kanálu svázán s libovolnou adresou TCP/IP.

public int TraceRouteRecording {get;}
Řídí záznam informací o trasování trasy.

metody

public MQProcess AccessProcess(string *processName*, int *openOptions*);
public MQProcess AccessProcess(string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

Vyvolá výjimku `MQException`.

Přístup k procesu IBM MQ v tomto správci front pro zjišťování atributů procesu.

processName
Název procesu, který se má otevřít.

openOptions
Volby, které řídí otevření procesu. Platné volby, které lze přidat nebo kombinovat pomocí bitové operace OR, jsou:

- `MQC.MQOO_FAIL_IF_QUIESCING`

- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

QueueManagerName
Název správce front, v němž je proces definován. Pokud je správce front stejný jako ten, ke kterému proces přistupuje, můžete ponechat prázdný název správce front nebo název správce front s hodnotou Null.

AlternateUserid

Je-li v parametru **openOptions** uveden parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY , *alternateUserId* uvádí alternativní ID uživatele použité ke kontrole autorizace pro akci. Není-li parametr MQOO_ALTERNATE_USER_AUTHORITY uveden, *alternateUserId* může být prázdný nebo mít hodnotu null.

Výchozí oprávnění uživatele se používá pro připojení ke správci front, není-li zadán parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY .

public MQQueue AccessQueue(string *queueName*, int *openOptions*);
public MQQueue AccessQueue(string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

Vyvolá výjimku MQException.

Přistupuje k frontě v tomto správci front.

Můžete získat nebo procházet zprávy, vkládat zprávy, dotazovat se na atributy fronty nebo nastavit atributy fronty. Pokud je pojmenovaná fronta modelovou frontou, vytvoří se dynamická lokální fronta. Chcete-li zjistit název dynamické fronty, zadejte dotaz na atribut name výsledného objektu MQQueue .

queueName
Název fronty, která se má otevřít.

openOptions
Volby, které řídí otevření fronty.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Ověřit s uvedeným identifikátorem uživatele.

MQC.MQOO_BIND_AS_QDEF
Použít výchozí vazbu pro frontu.

MQC.MQOO_BIND_NOT_FIXED
Nepřipojujte se ke specifickému cíli.

MQC.MQOO_BIND_ON_OPEN
Svázat manipulátor s cílem při otevření fronty.

MQC.MQOO_BROWSE
Otevřít pro procházení zpráv.

MQC.MQOO_FAIL_IF_QUIESCING
Pokud je správce front uveden do klidového stavu, dojde k selhání.

MQC.MQOO_INPUT_AS_Q_DEF
Otevřít pro získání zpráv pomocí výchozího nastavení definovaného frontou.

MQC.MQOO_INPUT_SHARED
Otevřít pro získání zpráv se sdíleným přístupem.

MQC.MQOO_INPUT_EXCLUSIVE
Otevřít pro získání zpráv s výlučným přístupem.

MQC.MQOO_INQUIRE
Otevřít pro dotaz-povinné, pokud chcete dotazovat vlastnosti.

MQC.MQOO_OUTPUT
Otevřít pro vložení zpráv.

MQC.MQOO_PASS_ALL_CONTEXT
Povolit předání všech kontextů.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Povolit předání kontextu identity.

MQC.MQOO_SAVE_ALL_CONTEXT
Uložit kontext při načtení zprávy.

MQC.MQOO_SET
Otevřít pro nastavení atributů-povinné, pokud chcete nastavit vlastnosti.

MQC.MQOO_SET_ALL_CONTEXT
Umožňuje nastavení všech kontextů.

MQC.MQOO_SET_IDENTITY_CONTEXT
Umožňuje nastavení kontextu identity.

QueueManagerName
Název správce front, v němž je fronta definována. Název, který je zcela prázdný nebo má hodnotu null, označuje správce front, ke kterému je objekt `MQQueueManager` připojen.

Název dynamicQueue
Parametr *dynamicQueueName* je ignorován, pokud `queueName` neuvádí název modelové fronty. Pokud ano, *dynamicQueueName* uvádí název dynamické fronty, která se má vytvořit. Prázdný název nebo název s hodnotou null není platný, pokud `queueName` uvádí název modelové fronty. Pokud je posledním neprázdným znakem v názvu hvězdička `*`, správce front nahradí hvězdičku řetězcem znaků. Znaky zaručují, že název vygenerovaný pro frontu je v tomto správci front jedinečný.

AlternateUserid
Je-li v parametru `openOptions` uveden parametr `MQC.MQOO_ALTERNATE_USER_AUTHORITY` , *alternateUserId* uvádí alternativní identifikátor uživatele, který se používá ke kontrole autorizace pro otevření. Není-li parametr `MQC.MQOO_ALTERNATE_USER_AUTHORITY` uveden, *alternateUserId* může být ponechán prázdný, nebo může mít hodnotu 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);
```

Přístup k tématu v tomto správci front.

Objekty `MQTopic` úzce souvisí s objekty administrativních témat, které se někdy nazývají objekty témat. Na vstupu `topicObject` ukazuje na objekt administrativního tématu. Konstruktor `MQTopic` získá řetězec tématu z objektu tématu a zkombinuje jej s názvem tématu `topicName` pro vytvoření názvu tématu. Buď `topicObject` , nebo `topicName` může mít hodnotu null. Název tématu odpovídá stromu témat a název nejbližšího odpovídajícího objektu administrativního tématu je vrácen v `topicObject`.

Témata, která jsou přidružena k objektu MQTopic , jsou výsledkem kombinace dvou řetězců témat. První řetězec tématu je definován objektem administrativního tématu identifikovaným pomocí *topicObject*. Druhý řetězec tématu je *topicString*. Výsledný řetězec tématu přidružený k objektu MQTopic může identifikovat více témat pomocí zástupných znaků.

V závislosti na tom, zda je téma otevřeno pro publikování nebo odběr, můžete použít metody MQTopic.Put pro publikování v tématech nebo metody MQTopic.Get pro příjem publikování v tématech. Chcete-li publikovat a přihlásit se k odběru stejného tématu, musíte k tématu přistupovat dvakrát, jednou pro publikování a jednou pro odběr.

Pokud vytvoříte objekt MQTopic pro odběr, aniž byste poskytli objekt MQDestination , předpokládá se spravovaný odběr. Pokud předáte frontu jako objekt MQDestination , předpokládá se nespravovaný odběr. Musíte se ujistit, že volby odběru, které jste nastavili, jsou konzistentní se spravovaným nebo nespravovaným odběrem.

cíl
destination je instance MQQueue . Poskytnutím *destination*se MQTopic otevře jako nespravovaný odběr. Publikování v tématu jsou doručena do fronty, ke které je přistupováno jako *destination*.

topicName
Řetězec tématu, který je druhou částí názvu tématu. *topicName* je zřetězeno s řetězcem tématu definovaným v objektu administrativního tématu *topicObject* . Parametr *topicName* můžete nastavit na hodnotu null. V takovém případě je název tématu definován řetězcem tématu v souboru *topicObject*.

topicObject
Na vstupu je *topicObject* název objektu tématu, který obsahuje řetězec tématu tvořící první část názvu tématu. Řetězec tématu v souboru *topicObject* je zřetězen s řetězcem *topicName*. Pravidla pro vytváření řetězců témat jsou definována v části Kombinování řetězců témat.

Ve výstupu obsahuje soubor *topicObject* název objektu administrativního tématu, který je nejbližší shodou ve stromu témat s tématem identifikovaným řetězcem tématu.

openAs
Přístup k tématu pro publikování nebo odběr. Parametr může obsahovat pouze jednu z těchto voleb:

- MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION
- MQC.MQTOPIC_OPEN_AS_PUBLICATION

volby

Zkombinujte volby, které řídí otevření tématu pro publikování nebo odběr. Použijte konstanty MQC.MQSO_* pro přístup k tématu pro odběr a konstanty MQC.MQOO_* pro přístup k tématu pro publikování.

Je-li požadována více než jedna volba, sečtěte hodnoty nebo zkombinujte hodnoty voleb pomocí bitových operátorů OR .

AlternateUserid
Uveďte alternativní ID uživatele, které se použije ke kontrole požadované autorizace pro dokončení operace. Musíte zadat *alternateUserId*, pokud je v parametru voleb nastavena hodnota MQC.MQOO_ALTERNATE_USER_AUTHORITY nebo MQC.MQSO_ALTERNATE_USER_AUTHORITY .

subscriptionName
Parametr *subscriptionName* je povinný, pokud jsou poskytnuty volby MQC.MQSO_DURABLE nebo MQC.MQSO_ALTER . V obou případech je produkt MQTopic implicitně otevřen pro odběr. Pokud je nastaven parametr MQC.MQSO_DURABLE a existuje odběr, nebo pokud je nastaven parametr MQC.MQSO_ALTER , dojde k výjimce a odběr neexistuje.

Vlastnosti
Nastavte všechny speciální vlastnosti odběru uvedené pomocí hašovací tabulky. Zadané položky v hašovací tabulce jsou aktualizovány výstupními hodnotami. Položky se nepřidávají do transformační tabulky pro výstupní hodnoty sestavy.

- 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();

Vyvolá MQException

Vrací objekt MQAsyncStatus , který představuje asynchronní aktivitu pro připojení správce front.

public void Backout();

Vyvolá výjimku MQException.

Vrácení všech zpráv, které byly načteny nebo zapsány v synchronizačním bodu od posledního synchronizačního bodu.

Zprávy, které byly zapsány se sadou příznaků MQC.MQPMO_SYNCPOINT , jsou odebrány z front. Zprávy přečtené s příznakem MQC.MQGMO_SYNCPOINT jsou obnoveny ve frontách, ze kterých pocházejí. Pokud jsou zprávy trvalé, změny se zaprotokolují.

V případě klientů s možností opětovného připojení je kód příčiny MQRC_NONE vrácen klientovi po úspěšném opětovném připojení.

public void Begin();

Vyvolá výjimku MQException.

Produkt Begin je podporován pouze v režimu vazeb serveru. Spustí globální jednotku práce.

public void Commit();

Vyvolá výjimku MQException.

Potvrďte všechny zprávy, které byly načteny nebo zapsány v synchronizačním bodu od posledního synchronizačního bodu.

Zprávy zapsané se sadou příznaků MQC.MQPMO_SYNCPOINT jsou zpřístupněny jiným aplikacím. Zprávy načtené se sadou příznaků MQC.MQGMO_SYNCPOINT jsou odstraněny. Pokud jsou zprávy trvalé, změny se zaprotokolují.

Klientovi s možností opětovného připojení jsou vráceny následující kódy příčiny:

- MQRC_CALL_INTERRUPTED , pokud dojde ke ztrátě připojení během provádění volání potvrzení.
- MQRC_BACKED_OUT , pokud je volání potvrzení vydáno po opětovném připojení.

Disconnect();

Vyvolá výjimku MQException.

Zavřete připojení ke správci front. Všechny objekty, ke kterým je přistupováno v tomto správci front, již nejsou pro tuto aplikaci přístupné. Chcete-li znovu přistupovat k objektům, vytvořte objekt MQQueueManager .

Obecně platí, že jakákoli práce provedená jako součást jednotky práce je potvrzena. Je-li však jednotka práce spravována produktem .NET, může být odvolána.

```
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);
```

Vyvolá výjimku `MQException`.

Umístí jednu zprávu do fronty nebo tématu bez předchozího vytvoření objektu `MQQueue` nebo `MQTopic` .

queueName
Název fronty, do které má být zpráva umístěna.

destinationName
Název cílového objektu. Jedná se buď o frontu, nebo téma v závislosti na hodnotě *type*.

typ
Typ cílového objektu. Volby nesmíte kombinovat.

MQC.MQOT_Q
Fronta

MQC.MQOT_TOPIC
Téma

QueueManagerName

Název správce front nebo alias správce front, v němž je fronta definována. Je-li zadán typ `MQC.MQOT_TOPIC` , bude tento parametr ignorován.

Pokud se jedná o modelovou frontu a interpretovaný název správce front není tento správce front, dojde k vyvolání `MQException` .

topicString

topicString se kombinuje s názvem tématu v objektu tématu *destinationName* .

Parametr *topicString* je ignorován, pokud *destinationName* je fronta.

zpráva
Zpráva, která se má odeslat. Zpráva je vstupní/výstupní objekt.

Klientovi s možností opětovného připojení jsou vráceny následující kódy příčiny:

- `MQRC_CALL_INTERRUPTED` , pokud je připojení přerušeno při provádění volání Put pro trvalou zprávu.
- `MQRC_NONE` je-li připojení úspěšné při provádění volání Put pro dočasnou zprávu (viz Obnova aplikace).

Volby putMessage

Volby, které řídí akce vložení.

Pokud vynecháte *putMessageOptions*, vytvoří se výchozí instance *putMessageOptions* . *putMessageOptions* je vstupní/výstupní objekt.

Pokud použijete volbu `MQPMO_LOGICAL_ORDER` v klientovi s možností opětovného připojení, vrátí se kód příčiny `MQRC_RECONNECT_INCOMPATIBLE` .

AlternateUserid

Uvádí alternativní identifikátor uživatele, který se používá ke kontrole autorizace při umístění zprávy do fronty.

Můžete vynechat *alternateUserId* , pokud nenastavíte MQC.MQOO_ALTERNATE_USER_AUTHORITY v *putMessageOptions*. Pokud nastavíte MQC.MQOO_ALTERNATE_USER_AUTHORITY, musíte také nastavit *alternateUserId*. *alternateUserId* se neprojeví, pokud nenastavíte také MQC.MQOO_ALTERNATE_USER_AUTHORITY.

Konstruktory

```
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);
```

Vyvolá výjimku MQException.

Vytvoří připojení ke správci front. Vyberte mezi vytvořením připojení klienta nebo připojení serveru.

Při pokusu o připojení ke správci front musíte mít oprávnění dotazovat se (inq) na správce front. Bez oprávnění k dotazování se pokus o připojení nezdaří.

Připojení klienta se vytvoří, pokud je splněna jedna z následujících podmínek:

1. *channel* nebo *connName* jsou určeny v konstruktoru.
2. *HostName*, *Port*nebo *Channel* jsou uvedeny v souboru *properties*.
3. Jsou určeny položky *MQEnvironment.HostName*, *MQEnvironment.Port*nebo *MQEnvironment.Channel* .

Hodnoty vlastností připojení jsou standardně nastaveny v zobrazeném pořadí. Hodnoty *channel* a *connName* v konstruktoru mají přednost před hodnotami vlastností v konstruktoru. Hodnoty vlastností konstruktoru mají přednost před vlastnostmi MQEnvironment .

Název hostitele, název kanálu a port jsou definovány ve třídě MQEnvironment .

QueueManagerName
Název správce front nebo skupiny správců front, ke které se chcete připojit.

Chcete-li provést výchozí výběr správce front, vynechte parametr nebo jej ponechte prázdný. Výchozí připojení správce front na serveru je k výchozímu správci front na serveru. Výchozí připojení správce front v připojení klienta je ke správci front, ke kterému je modul listener připojen.

volby
Zadejte volby připojení MQCNO . Hodnoty musí být použitelné pro typ připojení, které se provádí. Zadáte-li například následující vlastnosti připojení serveru pro připojení klienta, dojde k výjimce MQException .

- MQC.MQCNO_FASTPATH_BINDING
- MQC.MQCNO_STANDARD_BINDING

Vlastnosti

Parametr properties přebírá řadu dvojic klíč/hodnota, které potlačují vlastnosti nastavené pomocí MQEnvironment ; viz příklad “Potlačit vlastnosti MQEnvironment” na stránce 1767. Následující vlastnosti lze přepsat:

- 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

kanál
Název kanálu připojení serveru

connName
Název připojení ve formátu *HostName* (*Port*).

Jako argument konstruktoru `MQQueueManager (String queueManagerName, Hashtable properties)` můžete zadat seznam *názvů hostitelů* a *portů* pomocí vlastnosti CONNECTION_NAME_PROPERTY.

Příklad:

```
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);
```

Při pokusu o připojení se seznam názvů připojení zpracuje v uvedeném pořadí. Pokud pokus o připojení k prvnímu názvu hostitele a portu selže, provede se pokus o připojení k druhé dvojici atributů. Klient tento proces opakuje, dokud není vytvořeno úspěšné připojení nebo dokud není seznam vyčerpán. Pokud je seznam vyčerpán, vrátí se aplikaci klienta příslušný kód příčiny a kód dokončení.

Není-li pro název připojení poskytnuto číslo portu, výchozí port (konfigurovaný v adresáři `mqclient.ini`). se používá.

Nastavit seznam připojení

Seznam připojení můžete nastavit pomocí následujících metod, když jsou nastaveny volby automatického opětovného připojení klienta:

Nastavit seznam připojení prostřednictvím MQSERVER

Seznam připojení můžete nastavit pomocí příkazového řádku.

Na příkazovém řádku nastavte následující příkaz:

```
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/Hostname1(Port1),Hostname2(Por2),Hostname3(Port3)
```

Příklad:

```
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/fred.mq.com(5266),nick.mq.com(6566),jack.mq.com(8413)
```

Nastavíte-li připojení v serveru MQSERVER, nenastavujte jej v aplikaci.

Nastavíte-li seznam připojení v aplikaci, aplikace přepíše vše, co je nastaveno v proměnné prostředí MQSERVER.

Nastavení seznamu připojení prostřednictvím aplikace

Seznam připojení můžete v aplikaci nastavit zadáním názvu hostitele a vlastností portu.

```
String connName = "fred.mq.com(2344), nick.mq.com(3746), chris.mq.com(4288)";
MQQueueManager qm = new MQQueueManager("QM1", "TestChannel", connName);
```

Nastavit seznam připojení prostřednictvím `app.config`

`App.config` je soubor XML, ve kterém určíte dvojice klíč-hodnota.

V seznamu připojení zadejte

```
<app.Settings>
<add key="Connection1" value="Hostname1(Port1)"/>
<add key="Connection2" value="Hostname2(Port2)"/>
<app.Settings>
```

Příklad:

```
<app.Settings>
<add key>="Connection1" value="fred.mq.com(2966)"/>
<add key>="Connection2" value="alex.mq.com(6533)"/>
<app.Settings>
```

Seznam připojení můžete přímo změnit v souboru `app.config` .

Nastavit seznam připojení prostřednictvím `MQEnvironment`

Chcete-li nastavit seznam připojení prostřednictvím `MQEnvironment`, použijte vlastnost *ConnectionName* .

```
MQEnvironment.ConnectionName = "fred.mq.com(4288),"alex.mq.com(5211);
```

Vlastnost *ConnectionName* přepíše název hostitele a vlastnosti portu nastavené v souboru `MQEnvironment`.

Vytvořit připojení klienta

Následující příklad ukazuje, jak vytvořit připojení klienta ke správci front. Připojení klienta můžete vytvořit nastavením proměnných `MQEnvironment` před vytvořením nového objektu `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 IBM MQ port)
MQEnvironment.Channel  = "channel.name"; // the case sensitive
                                         //  name of the
                                         // SVR CONN channel on
                                         // the queue manager
MQQueueManager qMgr    = new MQQueueManager("MYQM");
```

Obrázek 11. Připojení klienta

Potlačit vlastnosti `MQEnvironment`

Následující příklad ukazuje, jak vytvořit správce front s jeho ID uživatele a heslem definovaným v hašovací tabulce.

```
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);
}
```

Obrázek 12. Potlačení vlastností `MQEnvironment`

Vytvořit znovu připojitelné připojení

Následující příklad ukazuje, jak automaticky znovu připojit klienta ke správci front.

```
Hashtable properties = new Hashtable(); // The queue manager name and the
                                 // properties how it has to be connected

properties.Add(MQC.CONNECT_OPTIONS_PROPERTY, MQC.MQCNO_RECONNECT); // Options
                                 // through which reconnection happens

properties.Add(MQC.CONNECTION_NAME_PROPERTY,"fred.mq.com(4789),nick.mq.com(4790)"); // The list
                                 // of queue managers through which reconnection happens

MQ QueueManager qmgr = new MQQueueManager("qmgrname", properties);
```

Obrázek 13. Automatické opětovné připojení klienta ke správci front

Třída `MQSubscription.NET`

Pomocí volby `MQSubscription` můžete požadovat, aby byla zachovaná publikování odesílána odběrateli. `MQSubscription` je vlastnost objektu `MQTopic` otevřeného pro odběr.

Třída

```
System.Object
        |
        └─ IBM.WMQ.MQBase
                |
                └─ IBM.WMQ.MQBaseObject
                        |
                        └─ IBM.WMQ.MQManagedObject
                                |
                                └─ IBM.WMQ.MQSubscription
```

```
public class IBM.WMQ.MQSubscription extends IBM.WMQ.MQManagedObject;
```

- “Vlastnosti” na stránce 1768
- “metody” na stránce 1768
- “Konstruktory” na stránce 1768

Vlastnosti

Přístup k vlastnostem odběru pomocí třídy `MQManagedObject` ; viz “Vlastnosti” na stránce 1727.

metody

Přístup k odběru `Inquire`, `Set` a `Get` metod pomocí třídy `MQManagedObject` ; viz “metody” na stránce 1728.

public int RequestPublicationUpdate(int *options*);

Vyvolá výjimku `MQException`.

Vyžádejte si aktualizované publikování pro aktuální téma. Pokud má správce front zachovaná publikování pro dané téma, jsou odeslána odběrateli.

Před voláním funkce `RequestPublicationUpdate`otevřete téma pro odběr a získejte objekt `MQSubscription` .

Obvykle otevřete odběr s volbou `MQC.MQSO_PUBLICATIONS_ON_REQUEST` . Pokud řetězec tématu neobsahuje žádné zástupné znaky, bude jako výsledek tohoto volání odesláno pouze jedno publikování. Pokud řetězec tématu obsahuje zástupné znaky, může být odesláno mnoho publikací. Tato metoda vrací počet zachovaných publikování, která jsou odeslána do fronty odběrů. Není zaručeno, že tento počet publikací bude přijat, zejména pokud se jedná o dočasné zprávy.

volby

MQC.MQSRO_FAIL_IF_QUIESCING
Metoda se nezdaří, pokud je správce front v klidovém stavu. V systému z/OSv případě aplikace CICS nebo IMS `MQC.MQSRO_FAIL_IF_QUIESCING` také vynutí selhání metody, pokud je připojení v klidovém stavu.

MQC.MQSRO_NONE
Nejsou zadány žádné volby.

Konstruktory

Žádný konstruktor Public .

Objekt `MQSubscription` je vrácen ve vlastnosti `SubscriptionReference` objektu `MQTopic` , který je otevřen pro odběr,

Zavolejte metodu `RequestPublicationUpdate` . `MQSubscription` je podtřídou třídy `MQManagedObject`. Použijte odkaz pro přístup k vlastnostem a metodám `MQManagedObject`.

Třída MQTopic.NET

Pomocí produktu `MQTopic` můžete publikovat nebo odebírat zprávy v tématu nebo dotazovat nebo nastavovat atributy tématu. Vytvořte objekt `MQTopic` pro publikování nebo odběr pomocí konstruktoru nebo metody `MQQueueManager.AccessTopic` .

Třída

```
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;
```

- “Vlastnosti” na stránce 1769
- “metody” na stránce 1769
- “Konstruktory” na stránce 1771

Vlastnosti

Test pro vyvolání `MQException` při získávání vlastností.

public Boolean IsDurable {get;}
Vlastnost jen pro čtení, která vrací hodnotu `True` , pokud je odběr trvalý nebo `False` jinak. Pokud bylo téma otevřeno pro publikování, vlastnost se ignoruje a vždy vrátí hodnotu `False`.

public Boolean IsManaged {get;};
Vlastnost jen pro čtení, která vrací hodnotu `True` , je-li odběr spravován správcem front, jinak hodnotu `False` . Pokud bylo téma otevřeno pro publikování, vlastnost se ignoruje a vždy vrátí hodnotu False.

public Boolean IsSubscribed {get;};
Vlastnost jen pro čtení, která vrací hodnotu `True` , pokud bylo téma otevřeno pro odběr, a hodnotu `False` , pokud bylo téma otevřeno pro publikování.

public MQSubscription SubscriptionReference {get;};
Vlastnost jen pro čtení, která vrací objekt `MQSubscription` přidružený k objektu tématu otevřenému pro odběr. Odkaz je k dispozici, pokud chcete upravit volby zavření nebo spustit některou z metod objektů.

public MQDestination UnmanagedDestinationReference {get;};
Vlastnost jen pro čtení, která vrací hodnotu `MQQueue` přidruženou k nespravovanému odběru. Jedná se o místo určení určené při vytvoření objektu tématu. Vlastnost vrací hodnotu null pro všechny objekty tématu otevřené pro publikování nebo se spravovaným odběrem.

metody

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);
Vyvolá výjimku MQException.

Publikuje zprávu do tématu.

Úpravy objektu `MQMessage` po provedení volání Put neovlivní skutečnou zprávu ve frontě IBM MQ nebo tématu publikování.

Produkt `Put` aktualizuje vlastnosti `MessageId` a `CorrelationId` objektu `MQMessage` a nevymaže data zprávy. Další volání `Put` nebo `Get` odkazují na aktualizované informace v objektu `MQMessage` . Například v následujícím úseku kódu obsahuje první zpráva a a druhou ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
```

```
msg.WriteString("b");
q.Put(msg,pmo);
```

zpráva
Objekt MQMessage obsahující data deskriptoru zprávy a zprávu, která má být odeslána. Deskriptor zprávy může být změněn v důsledku této metody. Hodnoty v deskriptoru zprávy bezprostředně po dokončení této metody jsou hodnoty, které byly vloženy do fronty nebo publikovány do tématu.

Klientovi s možností opětovného připojení jsou vráceny následující kódy příčiny:

- MQRC_CALL_INTERRUPTED , pokud je připojení přerušeno při spuštění volání Put pro trvalou zprávu a opětovné připojení je úspěšné.
- MQRC_NONE je-li připojení úspěšné při spuštění volání Put pro dočasnou zprávu (viz Obnova aplikace).

Volby putMessage
Volby, které řídí akci vložení.

Není-li parametr *putMessageOptions* uveden, použije se výchozí instance MQPutMessageOptons .

Pokud použijete volbu MQPMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Poznámka: Chcete-li pro jednoduchost a výkon vložit jednu zprávu do fronty, použijte objekt MQQueueManager.Put . Pro tento účel byste měli mít objekt MQQueue .

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);
Vyvolá výjimku MQException.

Načte zprávu z tématu.

Tato metoda používá k získání výchozí instanci MQGetMessageOptions . Použitá volba zprávy je MQGMO_NOWAIT.

Pokud se příkaz get nezdaří, objekt MQMessage se nezmění. V případě úspěchu jsou deskriptor zprávy a datové části zprávy MQMessage nahrazeny deskriptorem zprávy a daty zprávy z příchozí zprávy.

Všechna volání IBM MQ z určitého MQQueueManager jsou synchronní. Proto, pokud provedete operaci get s wait, všechny ostatní podprocesy používající stejný MQQueueManager budou blokovány pro další volání IBM MQ , dokud nebude volání Get dokončeno. Potřebujete-li pro přístup k produktu IBM MQ současně více podprocesů, musí každý podproces vytvořit svůj vlastní objekt MQQueueManager .

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá pole v deskriptoru zprávy jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry MessageId a CorrelationId byly nastaveny podle potřeby.

Klient s možností opětovného připojení vrátí kód příčiny MQRC_BACKED_OUT po úspěšném opětovném připojení pro zprávy přijaté pod MQGM_SYNCPOINT.

Volby getMessage
Volby, které řídí akci získání.

Použití volby MQC.MQGMO_CONVERT může vést k výjimce s kódem příčiny MQC.MQRC_CONVERTED_STRING_TOO_BIG při převodu z jednobajtových znakových kódů na dvoubajtové kódy. V tomto případě se zpráva zkopíruje do vyrovnávací paměti bez převodu.

Není-li parametr *getMessageOptions* uveden, použije se volba zprávy MQGMO_NOWAIT.

Pokud použijete volbu MQGMO_LOGICAL_ORDER v klientovi s možností opětovného připojení, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Velikost MaxMsg
Největší zpráva, kterou má tento objekt zprávy přijmout. Pokud je zpráva ve frontě větší než tato velikost, dojde k jedné ze dvou věcí:

- Je-li v objektu `MQGetMessageOptions` nastaven příznak `MQGMO_ACCEPT_TRUNCATED_MSG` , zpráva se vyplní co nejvíce dat zprávy. Dojde k výjimce s kódem dokončení `MQCC_WARNING` a kódem příčiny `MQRC_TRUNCATED_MSG_ACCEPTED` .
- Není-li příznak `MQGMO_ACCEPT_TRUNCATED_MSG` nastaven, zpráva zůstane ve frontě. Dojde k výjimce s kódem dokončení `MQCC_WARNING` a kódem příčiny `MQRC_TRUNCATED_MSG_FAILED` .

Není-li zadána hodnota *`MaxMsgSize`* , bude načtena celá zpráva.

Konstruktory

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*);

Přístup k tématu na webu *`queueManager`*.

Objekty `MQTopic` úzce souvisí s objekty administrativních témat, které se někdy nazývají objekty témat. Na vstupu `topicObject` ukazuje na objekt administrativního tématu. Konstruktor `MQTopic` získá řetězec tématu z objektu tématu a zkombinuje jej s názvem tématu `topicName` pro vytvoření názvu tématu. Buď `topicObject` , nebo `topicName` může mít hodnotu null. Název tématu odpovídá stromu témat a název nejbližšího odpovídajícího objektu administrativního tématu je vrácen v `topicObject`.

Témata, která jsou přidružena k objektu `MQTopic` , jsou výsledkem kombinace dvou řetězců témat. První řetězec tématu je definován objektem administrativního tématu identifikovaným pomocí *`topicObject`*. Druhý řetězec tématu je *`topicString`*. Výsledný řetězec tématu přidružený k objektu `MQTopic` může identifikovat více témat pomocí zástupných znaků.

V závislosti na tom, zda je téma otevřeno pro publikování nebo odběr, můžete použít metody `MQTopic.Put` pro publikování v tématech nebo metody `MQTopic.Get` pro příjem publikování v tématech. Chcete-li publikovat a přihlásit se k odběru stejného tématu, musíte k tématu přistupovat dvakrát, jednou pro publikování a jednou pro odběr.

Pokud vytvoříte objekt `MQTopic` pro odběr, aniž byste poskytli objekt `MQDestination` , předpokládá se spravovaný odběr. Pokud předáte frontu jako objekt `MQDestination` , předpokládá se nespravovaný odběr. Musíte se ujistit, že volby odběru, které jste nastavili, jsou konzistentní se spravovaným nebo nespravovaným odběrem.

queueManager
Správce front pro přístup k tématu.

cíl
destination je instance MQQueue . Poskytnutím *destination*se MQTopic otevře jako nespravovaný odběr. Publikování v tématu jsou doručena do fronty, ke které je přistupováno jako *destination*.

topicName
Řetězec tématu, který je druhou částí názvu tématu. *topicName* je zřetězeno s řetězcem tématu definovaným v objektu administrativního tématu *topicObject* . Parametr *topicName* můžete nastavit na hodnotu null. V takovém případě je název tématu definován řetězcem tématu v souboru *topicObject*.

topicObject
Na vstupu je *topicObject* název objektu tématu, který obsahuje řetězec tématu tvořící první část názvu tématu. Řetězec tématu v souboru *topicObject* je zřetězen s řetězcem *topicName*. Pravidla pro vytváření řetězců témat jsou definována v části Kombinování řetězců témat.

Ve výstupu obsahuje soubor *topicObject* název objektu administrativního tématu, který je nejbližší shodou ve stromu témat s tématem identifikovaným řetězcem tématu.

openAs
Přístup k tématu pro publikování nebo odběr. Parametr může obsahovat pouze jednu z těchto voleb:

- MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION
- MQC.MQTOPIC_OPEN_AS_PUBLICATION

volby

Zkombinujte volby, které řídí otevření tématu pro publikování nebo odběr. Použijte konstanty MQC.MQSO_* pro přístup k tématu pro odběr a konstanty MQC.MQOO_* pro přístup k tématu pro publikování.

Je-li požadována více než jedna volba, sečtěte hodnoty nebo zkombinujte hodnoty voleb pomocí bitových operátorů OR .

AlternateUserid
Uveďte alternativní ID uživatele, které se použije ke kontrole požadované autorizace pro dokončení operace. Musíte zadat *alternateUserId*, pokud je v parametru voleb nastavena hodnota MQC.MQOO_ALTERNATE_USER_AUTHORITY nebo MQC.MQSO_ALTERNATE_USER_AUTHORITY .

subscriptionName
Parametr *subscriptionName* je povinný, pokud jsou poskytnuty volby MQC.MQSO_DURABLE nebo MQC.MQSO_ALTER . V obou případech je produkt MQTopic implicitně otevřen pro odběr. Pokud je nastaven parametr MQC.MQSO_DURABLE a existuje odběr, nebo pokud je nastaven parametr MQC.MQSO_ALTER , dojde k výjimce a odběr neexistuje.

Vlastnosti
Nastavte všechny speciální vlastnosti odběru uvedené pomocí hašovací tabulky. Zadané položky v hašovací tabulce jsou aktualizovány výstupními hodnotami. Položky se nepřidávají do transformační tabulky pro výstupní hodnoty sestavy.

- 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*);

Přístup k tématu v tomto správci front.

Objekty `MQTopic` úzce souvisí s objekty administrativních témat, které se někdy nazývají objekty témat. Na vstupu `topicObject` ukazuje na objekt administrativního tématu. Konstruktor `MQTopic` získá řetězec tématu z objektu tématu a zkombinuje jej s názvem tématu `topicName` pro vytvoření názvu tématu. Buď `topicObject` , nebo `topicName` může mít hodnotu null. Název tématu odpovídá stromu témat a název nejbližšího odpovídajícího objektu administrativního tématu je vrácen v `topicObject`.

Témata, která jsou přidružena k objektu `MQTopic` , jsou výsledkem kombinace dvou řetězců témat. První řetězec tématu je definován objektem administrativního tématu identifikovaným pomocí *`topicObject`*. Druhý řetězec tématu je *`topicString`*. Výsledný řetězec tématu přidružený k objektu `MQTopic` může identifikovat více témat pomocí zástupných znaků.

V závislosti na tom, zda je téma otevřeno pro publikování nebo odběr, můžete použít metody `MQTopic.Put` pro publikování v tématech nebo metody `MQTopic.Get` pro příjem publikování v tématech. Chcete-li publikovat a přihlásit se k odběru stejného tématu, musíte k tématu přistupovat dvakrát, jednou pro publikování a jednou pro odběr.

Pokud vytvoříte objekt `MQTopic` pro odběr, aniž byste poskytli objekt `MQDestination` , předpokládá se spravovaný odběr. Pokud předáte frontu jako objekt `MQDestination` , předpokládá se nespravovaný odběr. Musíte se ujistit, že volby odběru, které jste nastavili, jsou konzistentní se spravovaným nebo nespravovaným odběrem.

cíl
`destination` je instance `MQQueue` . Poskytnutím *`destination`*se `MQTopic` otevře jako nespravovaný odběr. Publikování v tématu jsou doručena do fronty, ke které je přistupováno jako *`destination`*.

topicName
Řetězec tématu, který je druhou částí názvu tématu. *`topicName`* je zřetězeno s řetězcem tématu definovaným v objektu administrativního tématu *`topicObject`* . Parametr *`topicName`* můžete nastavit na hodnotu null. V takovém případě je název tématu definován řetězcem tématu v souboru *`topicObject`*.

topicObject
Na vstupu je *`topicObject`* název objektu tématu, který obsahuje řetězec tématu tvořící první část názvu tématu. Řetězec tématu v souboru *`topicObject`* je zřetězen s řetězcem *`topicName`*. Pravidla pro vytváření řetězců témat jsou definována v části Kombinování řetězců témat.

Ve výstupu obsahuje soubor *`topicObject`* název objektu administrativního tématu, který je nejbližší shodou ve stromu témat s tématem identifikovaným řetězcem tématu.

openAs
Přístup k tématu pro publikování nebo odběr. Parametr může obsahovat pouze jednu z těchto voleb:

- `MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION`
- `MQC.MQTOPIC_OPEN_AS_PUBLICATION`

volby

Zkombinujte volby, které řídí otevření tématu pro publikování nebo odběr. Použijte konstanty `MQC.MQSO_*` pro přístup k tématu pro odběr a konstanty `MQC.MQOO_*` pro přístup k tématu pro publikování.

Je-li požadována více než jedna volba, sečtěte hodnoty nebo zkombinujte hodnoty voleb pomocí bitových operátorů `OR` .

AlternateUserid
Uveďte alternativní ID uživatele, které se použije ke kontrole požadované autorizace pro dokončení operace. Musíte zadat *`alternateUserId`*, pokud je v parametru voleb nastavena hodnota `MQC.MQOO_ALTERNATE_USER_AUTHORITY` nebo `MQC.MQSO_ALTERNATE_USER_AUTHORITY` .

subscriptionName
Parametr *`subscriptionName`* je povinný, pokud jsou poskytnuty volby `MQC.MQSO_DURABLE` nebo `MQC.MQSO_ALTER` . V obou případech je produkt `MQTopic` implicitně otevřen pro odběr. Pokud je nastaven parametr `MQC.MQSO_DURABLE` a existuje odběr, nebo pokud je nastaven parametr `MQC.MQSO_ALTER` , dojde k výjimce a odběr neexistuje.

Vlastnosti
Nastavte všechny speciální vlastnosti odběru uvedené pomocí hašovací tabulky. Zadané položky v hašovací tabulce jsou aktualizovány výstupními hodnotami. Položky se nepřidávají do transformační tabulky pro výstupní hodnoty sestavy.

- `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`

Rozhraní `IMQObjectTrigger.NET`

Implementujte `IMQObjectTrigger` pro zpracování zpráv předaných monitorem **runmqdnm**.NET .

Rozhraní

```
public interface IBM.WMQMonitor.IMQObjectTrigger();
```

V závislosti na tom, zda je v příkazu **runmqdnm** zadáno řízení synchronizačního bodu, je zpráva odebrána z fronty před návratem metody `Execute` nebo po něm.

metody

void Execute (MQQueueManager *queueManager*, MQQueue *queue*, MQMessage *message*, string *param*);

queueManager
Správce front, který je hostitelem monitorované fronty.

fronta
Fronta je monitorována.

zpráva
Zpráva přečtená z fronty.

Parametr
Data předaná z `UserParameter`.

Rozhraní MQC.NET

Na konstantu `MQI` odkazujte předponou názvu konstanty s předponou `MQC.`. `MQC` definuje všechny konstanty používané `MQI`.

Rozhraní

```
System.Object
        │
        └─ IBM.WMQ.MQC
```

```
public interface IBM.WMQ.MQC extends System.Object;
```

Příklad

```
MQQueue queue;
queue.closeOptions = MQC.MQCO_DELETE;
```

Identifikátory znakové sady pro aplikace .NET

Popisy znakových sad, které můžete vybrat pro kódování zpráv .NET IBM MQ

Znaková sada	Popis
37	ibm037
437	ibm437 /Původní počítač PC
500	ibm500
819	iso-8859-1 / latin1 / ibm819
1200	Unicode
1208	UTF-8
273	ibm273
277	ibm277
278	ibm278
280	ibm280
284	ibm284
285	ibm285
297	ibm297
420	ibm420
424	ibm424
737	ibm737 /PC řečtina

Znaková sada	Popis
775	ibm775 /PC v pobaltském prostředí
813	iso-8859-7 /řecký/ ibm813
838	ibm838
850	ibm850 /PC Latin 1
852	ibm852 /PC Latin 2
855	ibm855 /Cyrilice PC
856	ibm856
857	ibm857 /PC turecká
860	ibm860 /portugalština (PC)
861	ibm861 /PC islandština
862	ibm862 /PC-hebrejština
863	ibm863 /PC-kanadská francouzština
864	ibm864 /arabština pro PC
865	ibm865 /PC Nordic
866	ibm866 /PC ruština
868	ibm868
869	ibm869 /PC Moderní řečtina
870	ibm870
871	ibm871
874	ibm874
875	ibm875
912	iso-8859-2 / latin2 / ibm912
913	iso-8859-3 / latin3 / ibm913
914	iso-8859-4 / latin4 / ibm914
915	iso-8859-5 /cyrilice/ ibm915
916	iso-8859-8 /hebrejský/ ibm916
918	ibm918
920	iso-8859-9 / latin5 / ibm920
921	ibm921
922	ibm922
930	ibm930
932	PC japonština
933	ibm933
935	ibm935
937	ibm937

Znaková sada	Popis
939	ibm939
942	ibm942
943	ibm943
948	ibm948
949	ibm949
950	ibm950 /Velká 5 tradiční čínština
954	EUCJIS-informační systém
964	ibm964 /CNS 11643 Tradiční čínština
970	ibm970
1006	ibm1006
1025	ibm1025
1026	ibm1026
1089	iso-8859-6 /arabština/ ibm1089
1097	ibm1097
1098	ibm1098
1112	ibm1112
1122	ibm1122
1123	ibm1123
1124	ibm1124
1250	Windows Latinská 2
1251	Windows Cyrilice
1252	Windows Latinská 1
1253	Windows Řečtina
1254	Windows Turečtina
1255	Windows Hebrejský
1256	Windows Arabština
1257	Windows Baltské jazyky
1258	Windows Vietnamština
1381	ibm1381
1383	ibm1383
2022	JIS
5601	ksc-5601 Korejština
33722	ibm33722

IBM MQ třídy C++

Třídy IBM MQ C++ zapouzdřují rozhraní MQI (IBM MQ Message Queue Interface). Existuje jeden soubor záhlaví C++, **imqi.hpp**, který pokrývá všechny tyto třídy.

Pro každou třídu se zobrazí následující informace:

Diagram hierarchie tříd
Diagram třídy zobrazující třídu v jejím vztahu dědičnosti k bezprostředním nadřízeným třídám, pokud existují.

Ostatní příslušné třídy
Odkazy na dokumenty s dalšími relevantními třídami, například s nadřízenými třídami, a třídami objektů používaných v podpisech metod.

Atributy objektů
Atributy třídy. Jedná se o doplněk k atributům definovaným pro všechny nadřízené třídy. Mnoho atributů odráží členy datové struktury IBM MQ (viz “Křížový odkaz C++ a MQI” na stránce 1779). Podrobné popisy viz “Atributy objektů” na stránce 799.

Konstruktory
Podpisy speciálních metod použitých k vytvoření objektu třídy.

Metody objektů (veřejné)
Podpisy metod, které vyžadují instanci třídy pro svou činnost a které nemají žádná omezení použití.

V případě, že se použije, jsou uvedeny také tyto informace:

Metody třídy (veřejné)
Podpisy metod, které nevyžadují instanci třídy pro svou činnost a které nemají žádná omezení použití.

Přetížené metody (nadřízená třída)
Podpisy těch virtuálních metod, které jsou definovány v nadřízených třídách, ale vykazují odlišné, polymorfní chování pro tuto třídu.

Metody objektů (chráněné)
Podpisy metod, které vyžadují instanci třídy pro svou operaci a jsou vyhrazeny pro použití implementacemi odvozených tříd. Tato sekce je zajímavá pouze pro spisovatele tříd, na rozdíl od uživatelů tříd.

Data objektu (chráněná)
Podrobnosti implementace pro data instance objektu dostupná pro implementace odvozených tříd. Tato sekce je zajímavá pouze pro spisovatele tříd, na rozdíl od uživatelů tříd.

Kódy příčin
Hodnoty MQRC_ * (viz kód příčiny a dokončení rozhraní API), které lze očekávat od metod, které selžou. Úplný seznam kódů příčiny, které se mohou vyskytnout pro objekt třídy, naleznete v dokumentaci nadřízené třídy. Zdokumentovaný seznam kódů příčiny pro třídu neobsahuje kódy příčiny pro nadřízené třídy.

Poznámka:

1. Objekty těchto tříd nezabezpečují podprocesy. To zajišťuje optimální výkon, ale dávejte pozor, abyste neměli přístup k žádnému objektu z více než jednoho podprocesu.
2. Doporučuje se, aby pro program s podporou podprocesů byl pro každý podproces použit samostatný objekt ImqQueueManager. Každý objekt správce musí mít svou vlastní nezávislou kolekci jiných objektů, která zajistí, že objekty v různých podprocesech budou vzájemně izolovány.

Třídy jsou:

- “ImqAuthenticationZáznam třídy C++” na stránce 1795
- “ImqBinary Třída C + +” na stránce 1797
- “Třída ImqCache C++” na stránce 1799
- “Třída ImqChannel C++” na stránce 1802
- “ImqCICSBridgeHeader třída jazyka C++” na stránce 1807

- “ImqDeadLetterHeader třída C++” na stránce 1814
- “ImqDistributionVypsat třídu C++” na stránce 1816
- “ImqError Třída C + +” na stránce 1817
- “ImqGetMessageOptions třída C++” na stránce 1818
- “Třída ImqHeader C++” na stránce 1822
- “ImqIMSBridgeHeader třída jazyka C++” na stránce 1823
- “Třída ImqItem C++” na stránce 1826
- “Třída ImqMessage C++” na stránce 1828
- “Třída ImqMessageTracker C++” na stránce 1834
- “Třída ImqNamelist C++” na stránce 1837
- “Třída ImqObject C++” na stránce 1839
- “Třída ImqProcess C++” na stránce 1844
- “ImqPutMessageOptions třída C++” na stránce 1846
- “Třída ImqQueue C++” na stránce 1848
- “Třída ImqQueueManager C++” na stránce 1858
- “Třída ImqReferenceHeader C++” na stránce 1874
- “ImqString Třída C + +” na stránce 1877
- “Třída ImqTrigger C++” na stránce 1882
- “Třída ImqWorkHeader C++” na stránce 1885

Křížový odkaz C++ a MQI

Tato kolekce témat obsahuje informace týkající se jazyka C++ pro rozhraní MQI.

Přečtěte si tyto informace společně s “Datové typy použité v rozhraní MQI” na stránce 235.

Tato tabulka spojuje datové struktury MQI s třídami C++ a soubory začlenění. Následující témata zobrazují informace o křížových odkazech pro každou třídu C + +. Tyto křížové odkazy se týkají použití základních procedurálních rozhraní IBM MQ . Třídy ImqBinary, ImqDistributionList a ImqString nemají žádné atributy, které spadají do této kategorie a jsou vyloučeny.

Tabulka 845. Datová struktura, třída a křížový odkaz souboru začlenění

datová struktura	**Třída**	**Zahrnout soubor**
MQAIR	Záznam ImqAuthentication	imqair.hpp
	ImqBinary	imqbin.hpp
	ImqCache	imqcac.hpp
MQCD	ImqChannel	imqchl.hpp
MQCIH.	ImqCICSBridgeHeader	imqcih.hpp
MQDLH	ImqDeadLetterHeader	imqdlh.hpp
MQOR	Seznam ImqDistribution	imqdst.hpp
	ImqError	imqerr.hpp
MQGMO	ImqGetMessageOptions	imqgmo.hpp
	ImqHeader	imqhdr.hpp
MQIIH.	ImqIMSBridgeHeader	imqiih.hpp
	ImqItem	imqitm.hpp

Tabulka 845. Datová struktura, třída a křížový odkaz souboru začlenění (pokračování)		
datová struktura	**Třída**	**Zahrnout soubor**
MQMD	ImqMessage	imqmsg.hpp
	Sledování 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	Správce ImqQueue	imqmgr.hpp
MQRMH	Záhlaví ImqReference	imqrfh.hpp
	ImqString	imqstr.hpp
MQTM	ImqTrigger	imqtrg.hpp
MQTMC		
MQTMC2	ImqTrigger	imqtrg.hpp
MQXQH		
MQWIH	Záhlaví ImqWork	imqwih.hpp

Křížový odkaz záznamu ImqAuthentication

Křížový odkaz atributů, datových struktur, polí a volání pro třídu ImqAuthenticationRecord C++.

Tabulka 846. Atributy, datové struktury, pole a volání			
Atribut	**datová struktura**	**Pole**	**Volání**
Název připojení	MQAIR	AuthInfoConnName	MQCONNX
heslo	MQAIR	LDAPPassword	MQCONNX
typ	MQAIR	AuthInfoType	MQCONNX
jméno uživatele	MQAIR	LDAPUserNamePtr	MQCONNX
	MQAIR	Odsazení LDAPUserName	MQCONNX
	MQAIR	LDAPUserNameDélka	MQCONNX

Křížový odkaz ImqCache

Křížový odkaz atributů a volání pro třídu ImqCache C + +.

Tabulka 847. Atributy a volání	
Atribut	**Volání**
automatická vyrovnávací paměť	MQGET
Délka vyrovnávací paměti	MQGET
ukazatel vyrovnávací paměti	MQGET, MQPUT

Tabulka 847. Atributy a volání (pokračování)	
Atribut	**Volání**
Délka dat	MQGET
Posun dat	MQGET
ukazatel na data	MQGET
délka zprávy	MQGET, MQPUT

Křížový odkaz ImqChannel

Křížový odkaz atributů, datových struktur, polí a volání pro třídu ImqChannel C + + +.

Tabulka 848. Atributy, datové struktury, pole a volání			
Atribut	**datová struktura**	**Pole**	**Volání**
dávkový prezenční signál	MQCD	BatchHeartbeat	MQCONNX
Název kanálu	MQCD	ChannelName	MQCONNX
Název připojení	MQCD	ConnectionName	MQCONNX
	MQCD	ShortConnectionnázev	MQCONNX
Komprese záhlaví	MQCD	Seznam HdrComp	MQCONNX
interval prezenčního signálu	MQCD	HeartbeatInterval	MQCONNX
Interval udržení aktivity	MQCD	KeepAliveInterval	MQCONNX
Lokální adresa	MQCD	LocalAddress	MQCONNX
Maximální délka zprávy	MQCD	MaxMsgLength	MQCONNX
Komprese zpráv	MQCD	Seznam MsgComp	MQCONNX
Název režimu	MQCD	ModeName	MQCONNX
heslo	MQCD	Heslo	MQCONNX
počet uživatelských procedur pro příjem	MQCD		MQCONNX
názvy uživatelských procedur pro příjem	MQCD	ReceiveExit	MQCONNX
	MQCD	ReceiveExitsDefinováno	MQCONNX
	MQCD	ReceiveExitPtr	MQCONNX
příjem uživatelských dat	MQCD	ReceiveUserData	MQCONNX
	MQCD	ReceiveUserDataPtr	MQCONNX
Název uživatelské procedury zabezpečení zprávy	MQCD	SecurityExit	MQCONNX
zabezpečení uživatelských dat	MQCD	Data SecurityUser	MQCONNX
počet uživatelských procedur pro odeslání	MQCD		MQCONNX
názvy uživatelských procedur pro odeslání	MQCD	SendExit	MQCONNX
	MQCD	SendExitsDefinováno	MQCONNX
	MQCD	SendExitPtr	MQCONNX
odeslání uživatelských dat	MQCD	Data SendUser	MQCONNX

Tabulka 848. Atributy, datové struktury, pole a volání (pokračování)			
Atribut	**datová struktura**	**Pole**	**Volání**
	MQCD	SendUserDataPtr	MQCONNX
SSL CipherSpec	MQCD	Specifikace sslCipher	MQCONNX
Typ ověřování klienta SSL	MQCD	Ověření sslClient	MQCONNX
Název partnera SSL	MQCD	SSLPEERNAME	MQCONNX
Jméno programu transakce	MQCD	TpName	MQCONNX
Typ přenosu	MQCD	TransportType	MQCONNX
Jméno uživatele	MQCD	UserIdentifier	MQCONNX

ImqCICSBridgeHeader křížový odkaz

Křížový odkaz atributů, datových struktur a polí pro třídu C++ ImqCICSBridgeHeader .

Tabulka 849. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
kód nestandardního ukončení mostu	MQCIH.	AbendCode
Deskriptor ADS	MQCIH.	AdsDescriptor
identifikátor upozornění	MQCIH.	AttentionId
ověřovatel	MQCIH.	Ověřovatel
kód dokončení mostu	MQCIH.	Kód BridgeCompletion
posunutí chyby mostu	MQCIH.	ErrorOffset
kód příčiny mostu	MQCIH.	BridgeReason
kód zrušení mostu	MQCIH.	CancelCode
úloha konverzace	MQCIH.	ConversationalTask
pozice kurzoru	MQCIH.	CursorPosition
token zařízení	MQCIH.	Poskytovaná služba
čas uchování zařízení	MQCIH.	Čas FacilityKeep
zařízení jako	MQCIH.	FacilityLike
funkce	MQCIH.	Funkce
interval čekání na získání	MQCIH.	GetWaitInterval
Typ odkazu	MQCIH.	LinkType
další identifikátor transakce	MQCIH.	ID NextTransaction
délka výstupních dat	MQCIH.	OutputDataDélka
formát odpovědi	MQCIH.	ReplyToFormát
návratový kód mostu	MQCIH.	ReturnCode
kód spuštění	MQCIH.	StartCode
stav ukončení úlohy	MQCIH.	Stav TaskEnd

Tabulka 849. Mapování atributů, datových struktur a polí (pokračování)		
Atribut	**datová struktura**	**Pole**
Identifikátor transakce	MQCIH.	TransactionId
kontrola uow	MQCIH.	UowControl
verze	MQCIH.	Verze

ImqDeadLetterHeader křížový odkaz

Křížový odkaz atributů, datových struktur a polí pro třídu C++ ImqDeadLetterHeader .

Tabulka 850. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
kód příčiny nedoručených dopisů	MQDLH	Příčina
Název správce cílových front	MQDLH	DestQMgrName
název cílové fronty	MQDLH	DestQName
Název vkládající aplikace	MQDLH	PutApplName
Typ vkládající aplikace	MQDLH	PutApplType
Datum vložení	MQDLH	PutDate
Čas vložení	MQDLH	PutTime

Křížový odkaz ImqError

Křížový odkaz atributů a volání třídy ImqError C + + +.

Tabulka 851. Atributy a volání	
Atribut	**Volání**
kód dokončení	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET
kód příčiny	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET

Křížový odkaz ImqGetMessageOptions

Křížový odkaz atributů, datových struktur a polí pro třídu ImqGetMessageOptions C++.

Tabulka 852. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
stav skupiny	MQGMO	GroupStatus
volby shody	MQGMO	MatchOptions
token zprávy	MQGMO	MessageToken
volby	MQGMO	Volby
vyřešený název fronty	MQGMO	ResolvedQName
vrácená délka	MQGMO	ReturnedLength
segmentace	MQGMO	Segmentace

Tabulka 852. Mapování atributů, datových struktur a polí (pokračování)		
Atribut	**datová struktura**	**Pole**
stav segmentu	MQGMO	SegmentStatus
	MQGMO	Signal1
	MQGMO	Signal2
účast synchronizačního bodu	MQGMO	Volby
Interval čekání	MQGMO	WaitInterval

Křížový odkaz ImqHeader

Křížový odkaz atributů, datových struktur a polí pro třídu ImqHeader C++.

Tabulka 853. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
znaková sada	MQDLH, MQIIH	CodedCharSetId
kódování	MQDLH, MQIIH	Kódování
formát	MQDLH, MQIIH	Formát
příznaky záhlaví	MQIIH, MQRMH	Příznaky

ImqIMSBridgeHeader křížový odkaz

Křížový odkaz atributů, datových struktur a polí pro třídu ImqAuthenticationRecord C++.

Tabulka 854. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
ověřovatel	MQIIH.	Ověřovatel
režim potvrzení	MQIIH.	CommitMode
potlačení logického terminálu	MQIIH.	LTermOverride
název mapy služeb formátu zpráv	MQIIH.	MFSMapName
formát odpovědi	MQIIH.	ReplyToFormát
rozsah zabezpečení	MQIIH.	SecurityScope
id instance transakce	MQIIH.	ID TranInstance
Stav transakce	MQIIH.	TranState

Křížový odkaz ImqItem

Křížový odkaz atributů a volání pro třídu ImqItem C++.

Tabulka 855. Atributy a volání	
Atribut	**Volání**
id struktury	MQGET

Křížový odkaz ImqMessage

Křížový odkaz atributů, datových struktur, polí a volání třídy ImqMessage C + +.

Tabulka 856. Atributy, datové struktury, pole a volání			
Atribut	**datová struktura**	**Pole**	**Volání**
data ID aplikace	MQMD	ApplIdentityData	
Data původu aplikace	MQMD	ApplOriginData	
Počet vrácení	MQMD	BackoutCount	
znaková sada	MQMD	CodedCharSetId	
kódování	MQMD	Kódování	
Vypršení	MQMD	Vypršení	
formát	MQMD	Formát	
Příznaky zprávy	MQMD	MsgFlags	
typ zprávy	MQMD	MsgType	
posunutí	MQMD	Offset	
Původní délka	MQMD	OriginalLength	
trvání, perzistence	MQMD	Trvání	
priorita	MQMD	Priorita	
Název vkládající aplikace	MQMD	PutApplName	
Typ vkládající aplikace	MQMD	PutApplType	
Datum vložení	MQMD	PutDate	
Čas vložení	MQMD	PutTime	
název správce front pro odpověď	MQMD	ReplyToQMgr	
název fronty pro odpověď	MQMD	ReplyToQ	
sestava	MQMD	Sestava	
pořadové číslo	MQMD	MsgSeqNumber	
celková délka zprávy		DataLength	MQGET
Jméno uživatele	MQMD	UserIdentifier	

Křížový odkaz sledování ImqMessage

Křížový odkaz atributů, datových struktur a polí pro třídu ImqMessageTracker C + +.

Tabulka 857. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
Token evidence	MQMD	AccountingToken
ID korelace	MQMD	CorrelId
Zpětná vazba	MQMD	Zpětná vazba
ID skupiny	MQMD	GroupId
ID zprávy	MQMD	MsgId

Křížový odkaz ImqNamelist

Křížový odkaz atributů, dotazů a volání pro třídu ImqNamelist C + + +.

Tabulka 858. Atributy, dotazy a volání		
Atribut	**Dotaz**	**Volání**
Počet názvů	MQIA_NAME_COUNT	MQINQ
Název seznamu názvů	MQCA_NAMELIST_NAME	MQINQ

Křížový odkaz ImqObject

Křížový odkaz atributů, datových struktur, polí, dotazů a volání pro třídu ImqObject C++.

Tabulka 859. Atributy, datové struktury, pole, dotazy a volání				
Atribut	**datová struktura**	**Pole**	**Dotaz**	**Volání**
Datum změny			MQCA_ALTERATION_DATE	MQINQ
Čas změny			MQCA_ALTERATION_TIME	MQINQ
Jméno alternativního uživatele	MQOD	AlternateUserid		
alternativní ID zabezpečení				
volby zavření				MQCLOSE
description			MQCA_Q_DESC, MQCA_Q_MGR_DESC, MQCA_PROCESS_DESC	MQINQ
Název	MQOD	ObjectName	MQCA_Q_MGR_NAME, MQCQ_Q_NAME, MQCA_PROCESS_NAME	MQINQ
Volby otevření				MQOPEN
stav otevření				MQOPEN, MQCLOSE
identifikátor správce front	identifikátor správce front		MQCA_Q_MGR_IDENTIFIER	MQINQ

Křížový odkaz ImqProcess

Křížový odkaz na atributy, dotazy a volání třídy ImqAuthenticationRecord C++.

Tabulka 860. Atributy, dotazy a volání		
Atribut	**Dotaz**	**Volání**
ID aplikace	MQCA_APPL_ID	MQINQ
Typ aplikace	MQIA_APPL_TYPE	MQINQ
Data prostředí	MQCA_ENV_DATA	MQINQ
Data uživatele	MQCA_USER_DATA	MQINQ

Křížový odkaz ImqPutMessageOptions

Křížový odkaz atributů, datových struktur a polí pro třídu ImqAuthenticationRecord C++.

Tabulka 861. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
odkaz na kontext	MQPMO	Kontext
	MQPMO	InvalidDestCount
	MQPMO	KnownDestCount
volby	MQPMO	Volby
pole záznamu	MQPMO	PutMsgRecFields
název vyřešeného správce front	MQPMO	Název ResolvedQMgr
vyřešený název fronty	MQPMO	ResolvedQName
	MQPMO	Časový limit
	MQPMO	UnknownDestCount
účast synchronizačního bodu	MQPMO	Volby

Křížový odkaz ImqQueue

Křížový odkaz atributů, datových struktur, polí, dotazů a volání pro třídu ImqQueue C + +.

Tabulka 862. Křížový odkaz ImqQueue				
Atribut	**datová struktura**	**Pole**	**Dotaz**	**Volání**
Zpětné jméno přefrontování			MQCA_BACKOUT_REQ_Q_NAME	MQINQ
Práh vrácení			MQIA_BACKOUT_THRESHOLD	MQINQ
název základní fronty			MQCA_BASE_Q_NAME	MQINQ
název klastru			MQCA_CLUSTER_NAME	MQINQ
Název seznamu názvů klastru			MQCA_CLUSTER_NAMELIST	MQINQ
Rozsah vytíľení klastru			MQIA_CLWL_Q_RANK	MQINQ
Priorita vytíľení klastru			MQIA_CLWL_Q_PRIORITY	MQINQ
Pracovní zátěž klastru - použitá fronta			MQIA_CLWL_USEQ	MQINQ
Datum vytvoření			MQCA_CREATION_DATE	MQINQ
Čas vytvoření			MQCA_CREATION_TIME	MQINQ
Aktuální délka			MQIA_CURRENT_Q_DEPTH	MQINQ
výchozí vazba			MQIA_DEF_BIND	MQINQ
Výchozí volba otevření pro vstup			MQIA_DEF_INPUT_OPEN_OPTION	MQINQ

Tabulka 862. Křížový odkaz ImqQueue (pokračování)

Atribut	datová struktura	Pole	Dotaz	Volání
Výchozí trvání			PERZISTENCE MQIA_DEF_PERSISTENCE	MQINQ
Výchozí priorita			MQIA_DEF_PRIORITY	MQINQ
Typ definice			MQIA_DEFINITION_TYPE	MQINQ
událost s vysokou hloubkou			MQIA_Q_DEPTH_HIGH_EVENT	MQINQ
horní mez hloubky			MQIA_Q_DEPTH_HIGH_LIMIT	MQINQ
hloubka nízká událost			MQIA_Q_DEPTH_LOW_EVENT	MQINQ
dolní mez hloubky			MQIA_Q_DEPTH_LOW_LIMIT	MQINQ
událost maxima hloubky			MQIA_Q_DEPTH_MAX_LIMIT	MQINQ
distribuční seznamy			MQIA_DIST_LISTS	MQINQ, MQSET
název dynamické fronty	MQOD	DynamicQName		
Uloľení počtu vrácení			MQIA_HARDEN_GET_BACKOUT	MQINQ
Typ indexu			MQIA_INDEX_TYPE	MQINQ
inhibovat získání			MQIA_INHIBIT_GET	MQINQ, MQSET
blokování vložení			MQIA_INHIBIT_PUT	MQINQ, MQSET
Název inicializační fronty			MQCA_INITIATION_Q_NAME	MQINQ
Maximální hloubka			MQIA_MAX_Q_HLOUBKA	MQINQ
Maximální délka zprávy			MQIA_MAX_MSG_LENGTH	MQINQ
Pořadí doručení zpráv			MQIA_MSG_DELIVERY_SEQUENCE	MQINQ
další distribuovaná fronta				
Třída netrvalých zpráv			MQIA_NPM_CLASS	MQINQ
Otevření pro vstup - počet			MQIA_OPEN_INPUT_COUNT	MQINQ
Otevření pro výstup - počet			MQIA_OPEN_OUTPUT_COUNT	MQINQ
předchozí distribuovaná fronta				
Název procesu			MQCA_PROCESS_NAME	MQINQ
Účtování fronty			MQIA_ACCOUNTING_Q	MQINQ

Tabulka 862. Křížový odkaz ImqQueue (pokračování)				
Atribut	**datová struktura**	**Pole**	**Dotaz**	**Volání**
Název správce front	MQOD	ObjectQMgrName		
Monitorování fronty			MQIA_MONITORING_Q	MQINQ
Statistiky fronty			MQIA_STATISTICS_Q	MQINQ
Typ fronty			MQIA_Q_TYPE	MQINQ
Název vzdáleného správce front			MQCA_REMOTE_Q_MGR_NAME	MQINQ
Název vzdálené fronty			MQCA_REMOTE_Q_NAME	MQINQ
název vyřešeného správce front	MQOD	Název ResolvedQMgr		
vyřešený název fronty	MQOD	ResolvedQName		
Interval uchování			MQIA_RETENTION_INTERVAL	MQINQ
rozsah			MQIA_SCOPE	MQINQ
interval služeb			MQIA_Q_SERVICE_INTERVAL	MQINQ
událost intervalu služeb			MQIA_Q_SERVICE_INTERVAL_EVENT	MQINQ
Možnost sdílení			MQIA_SHAREABILITY	MQINQ
paměťová třída			MQCA_STORAGE_CLASS	MQINQ
Jméno přenosové fronty			MQCA_XMIT_Q_NAME	MQINQ
Řízení spouštěče			MQIA_TRIGGER_CONTROL	MQINQ, MQSET
Data spouątěče			MQCA_TRIGGER_DATA	MQINQ, MQSET
Hloubka spouątěče			MQIA_TRIGGER_HLOUBKA	MQINQ, MQSET
Priorita zpráv spouątěče			MQIA_TRIGGER_MSG_PRIORITY	MQINQ, MQSET
typ spouštěče			MQIA_TRIGGER_TYPE	MQINQ, MQSET
Využití			MQIA_USAGE	MQINQ

Křížový odkaz správce ImqQueue

Křížový odkaz atributů, datových struktur, polí, dotazů a volání pro třídu ImqQueueManager C + +.

Tabulka 863. Atributy, datové struktury, pole, dotazy a volání

Atribut	datová struktura	Pole	Dotaz	Volání
přepsání evidenčních připojení			MQIA_ACCOUNTING_CONN_OVERRIDE	MQINQ
Interval evidence			MQIA_ACCOUNTING_INTERVAL	MQINQ
Záznam činnosti			MQIA_ACTIVITY_RECORDING	MQINQ
Převzetí nového agenta MCA - kontrola			MQIA_ADOPTNEWMCA_CHECK	MQINQ
Převzetí nového agenta MCA - typ			MQIA_ADOPTNEWMCA_TYPE	MQINQ
Typ ověřování	MQCSP.	AuthenticationType		MQCONNX
událost oprávnění			MQIA_AUTHORITY_EVENT	MQINQ
volby začátku	MQBO	Volby		MQBEGIN
událost mostu			MQIA_BRIDGE_EVENT	MQINQ
Automatická definice kanálů			MQIA_CHANNEL_AUTO_DEF	MQINQ
Událost automatické definice kanálu			MQIA_CHANNEL_AUTO_EVENT	MQIA
Uživatelská procedura automatické definice kanálů			MQIA_CHANNEL_AUTO_EXIT	MQIA
událost kanálu			MQIA_CHANNEL_EVENT	MQINQ
Adaptéry inicializátoru kanálu			MQIA_CHINIT_ADAPTERS	MQINQ
Řízení iniciátoru kanálu			MQIA_CHINIT_CONTROL	MQINQ
Dispečeři inicializátoru kanálu			Dispečeři MQIA_CHINIT_DISPATCHERS	MQINQ
Automatické spuštění trasování inicializátoru kanálu			MQIA_CHINIT_TRACE_AUTO_START	MQINQ
Velikost tabulky trasování inicializátoru kanálu			MQIA_CHINIT_TRACE_TABLE_SIZE	MQINQ

Tabulka 863. Atributy, datové struktury, pole, dotazy a volání (pokračování)

Atribut	datová struktura	Pole	Dotaz	Volání
Monitorování kanálů			MQIA_MONITORING_CHANNEL	MQINQ
odkaz na kanál	MQCD	ChannelType		MQCONNX
Statistika kanálů			MQIA_STATISTICS_CHANNEL	MQINQ
znaková sada			MQIA_CODED_CHAR_SET_ID	MQINQ
Monitorování odesílatele klastru			MQIA_MONITORING_AUTO_CLUSSDR	MQINQ
Statistiky odesílatele klastru			MQIA_STATISTICS_AUTO_CLUSSDR	MQINQ
Data pracovní zátěže klastru			MQCA_CLUSTER_WORKLOAD_DATA	MQINQ
Uživatelská procedura pracovní zátěže klastru			MQCA_CLUSTER_WORKLOAD_EXIT	MQINQ
Délka pracovní zátěže klastru			MQIA_CLUSTER_WORKLOAD_LENGTH	MQINQ
mru pracovní zátěže klastru			MQIA_CLWL_MRU_CHANNELS	MQINQ
Pracovní zátěž klastru - použitá fronta			MQIA_CLWL_USEQ	MQINQ
událost příkazu			Událost MQIA_COMMAND_EVENT	MQINQ
Název fronty vstupu příkazů			MQCA_COMMAND_INPUT_Q_NAME	MQINQ
Úroveň příkazů			Úroveň_příkazu MQIA_COMMAND_LEVEL	MQINQ
Řízení příkazového serveru			MQIA_CMD_SERVER_CONTROL	MQINQ
Volby připojení	MQCNO	Volby		MQCONN, MQCONNX
ID připojení	MQCNO	ConnectionId		MQCONNX
Stav připojení				MQCONN, MQCONNX, MQDISC
Značka připojení	MQCD	ConnTag		MQCONNX
Kryptografický hardware	MQSCO	CryptoHardware		MQCONNX
název fronty nedoručených zpráv			MQCA_DEAD_LETTER_Q_NAME	MQINQ

Tabulka 863. Atributy, datové struktury, pole, dotazy a volání (pokračování)

Atribut	datová struktura	Pole	Dotaz	Volání
název výchozí přenosové fronty			MQCA_DEF_XMIT_Q_NAME	MQINQ
distribuční seznamy			MQIA_DIST_LISTS	MQINQ
skupina dns			MQCA_DNS_GROUP	MQINQ
dns wlm			MQIA_DNS_WLM	MQINQ
první záznam ověření	MQSCO	AuthInfoRecOffset		MQCONNX
	MQSCO	AuthInfoRecPtr		MQCONNX
blokovat událost			Událost MQIA_INHIBITORY	MQINQ
Verze adresy IP			MQIA_IP_ADDRESS_VERSION	MQINQ
úložiště klíčů	MQSCO	KeyRepository		MQCONNX
počet resetů klíče	MQSCO	Počet KeyReset		MQCONNX
Časovač modulu listener			Časovač MQIA_LISTENER_TIMER	MQINQ
lokální událost			MQIA_LOKÁLNÍ_UDÁLOST	MQINQ
Událost modulu protokolování			Událost MQIA_LOGGER_EVENT	MQINQ
Název skupiny LU			MQCA_LU_NÁZEV_SKUPINY	MQINQ
Název jednotky LU			MQCA_LU_NAME	MQINQ
lu62 přípona ramene			MQCA_LU62_ARM_SUFFIX	MQINQ
lu62 kanálů			MQIA_LU62_CHANNELS	MQINQ
maximální počet aktivních kanálů			MQIA_ACTIVE_CHANNELS	MQINQ
Maximální počet kanálů			MQIA_MAX_CHANNELS	MQINQ
maximální počet manipulátorů			MQIA_MAX_MANIPULÁTORY	MQINQ
Maximální délka zprávy			MQIA_MAX_MSG_LENGTH	MQINQ
maximální priorita			MQIA_MAX_PRIORITY	MQINQ
Maximum nepotvrzených zpráv			MQIA_MAX_UNCOMMITTED_MSGS	MQINQ
Evidence MQI			MQIA_ACCOUNTING_MQI	MQINQ
Statistika MQI			MQIA_STATISTICS_MQI	MQINQ

Tabulka 863. Atributy, datové struktury, pole, dotazy a volání (pokračování)

Atribut	datová struktura	Pole	Dotaz	Volání
maximální počet odchozích portů			MQIA_OUTBOUND_PORT_MAX	MQINQ
minimum odchozího portu			MQIA_OUTBOUND_PORT_MIN	MQINQ
heslo	MQCSP.	CSPPasswordPtr		MQCONNX
	MQCSP.	CSPPasswordOffset		MQCONNX
	MQCSP.	CSPPasswordLength		MQCONNX
událost výkonu			Událost MQIA_PERFORMANCE_EVENT	MQINQ
platforma			MQIA_PLATFORM	MQINQ
Účtování fronty			MQIA_ACCOUNTING_Q	MQINQ
Monitorování fronty			MQIA_MONITORING_Q	MQINQ
Statistiky fronty			MQIA_STATISTICS_Q	MQINQ
Časový limit pro příjem			MQIA_RECEIVE_TIMEOUT	MQINQ
minimum časového limitu příjmu			MQIA_RECEIVE_TIMEOUT_MIN	MQINQ
Typ časového limitu pro příjem			MQIA_RECEIVE_TIMEOUT_TYPE	MQINQ
vzdálená událost			MQIA_REMOTE_EVENT	MQINQ
REPOSITORY NAME			MQCA_REPOSITORY_NAME	MQINQ
Seznam názvů úložiště			MQCA_REPOSITORY_NAMELIST	MQINQ
název správce front sdílené fronty			MQIA_SHARED_Q_Q_MGR_NAME	MQINQ
Událost ssl			MQIA_SSL_EVENT	MQINQ
ssl fips			MQIA_SSL_FIPS_REQUIRED	MQINQ
Počet resetování klíče SSL			MQIA_SSL_RESET_COUNT	MQINQ
událost start-stop			MQIA_START_STOP_EVENT	MQINQ
Interval statistiky			MQIA_STATISTICS_INTERVAL	MQINQ
Dostupnost synchronizačního bodu			MQIA_SYNCPOINT	MQINQ
kanály tcp			MQIA_TCP_CHANNELS	MQINQ

Tabulka 863. Atributy, datové struktury, pole, dotazy a volání (pokračování)				
Atribut	**datová struktura**	**Pole**	**Dotaz**	**Volání**
Udržení připojení TCP			MQIA_TCP_KEEP_ALIVE	MQINQ
Název TCP			MQCA_TCP_NAME	MQINQ
Typ sady protokolů TCP			MQIA_TCP_STACK_TYPE	MQINQ
Záznam přenosových tras			Záznam MQIA_TRACE_ROUTE_RECORDING	MQINQ
Interval spouštěče			MQIA_TRIGGER_INTERVAL	MQINQ
Jméno uživatele	MQCSP.	CSPUserIdPtr		MQCONNX
	MQCSP.	Odsazení CSPUserId		MQCONNX
	MQCSP.	CSPUserIdDélka		MQCONNX

Křížový odkaz záhlaví ImqReference

Křížový odkaz atributů, datových struktur a polí pro třídu ImqAuthenticationRecord C++.

Tabulka 864. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
cílové prostředí	MQRMH	DestEnvDélka, DestEnvOdchylka
Název místa určení	MQRMH	DestNameDélka, DestNameOffset
ID instance	MQRMH	ID ObjectInstance
logická délka	MQRMH	DataLogicalDélka
logické posunutí	MQRMH	DataLogicalOffset
logický posun 2	MQRMH	DataLogicalOffset2
Typ odkazu	MQRMH	ObjectType
Zdrojové prostředí	MQRMH	SrcEnvDélka, SrcEnvOffset
Zdrojový název	MQRMH	SrcNameDélka, SrcNameOffset

Křížový odkaz ImqTrigger

Křížový odkaz atributů, datových struktur a polí pro třídu ImqAuthenticationRecord C++.

Tabulka 865. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
ID aplikace	MQTM	ApplId
Typ aplikace	MQTM	ApplType
Data prostředí	MQTM	EnvData
Název procesu	MQTM	ProcessName

Tabulka 865. Mapování atributů, datových struktur a polí (pokračování)		
Atribut	**datová struktura**	**Pole**
Název fronty	MQTM	QName
Data spouątěče	MQTM	TriggerData
Data uživatele	MQTM	UserData

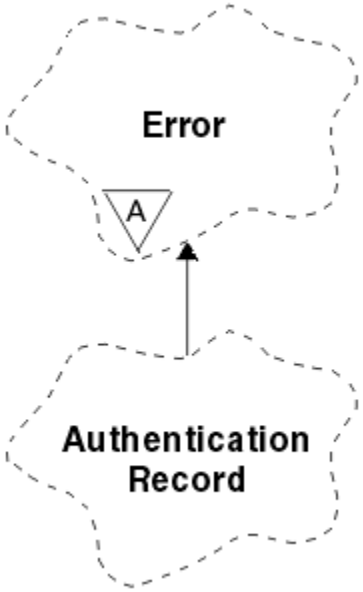
Křížový odkaz záhlaví ImqWork

Křížový odkaz atributů, datových struktur a polí pro třídu ImqAuthenticationRecord C++.

Tabulka 866. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
token zprávy	MQWIH	MessageToken
Název služby	MQWIH	ServiceName
servisní krok	MQWIH	ServiceStep

ImqAuthenticationZáznam třídy C++

Tato třída zapouzdřuje záznam ověřovacích informací (MQAIR) pro použití během provádění metody ImqQueueManager: :connect pro vlastní připojení klienta TLS.



Obrázek 14. Třída záznamu ImqAuthentication

Další podrobnosti naleznete v popisu metody ImqQueueManager: :connect. Tato třída není k dispozici na platformě z/OS .

- “Atributy objektů” na stránce 1796
- “Konstruktory” na stránce 1796
- “Metody objektů (veřejné)” na stránce 1796
- “Metody objektů (chráněné)” na stránce 1797

Atributy objektů

Název připojení
Název připojení k serveru LDAP CRL. Jedná se o adresu IP nebo název DNS, za nímž volitelně následuje číslo portu v závorkách.

odkaz na připojení
Odkaz na objekt ImqQueueManager, který poskytuje požadované připojení k (lokálnímu) správci front. Počáteční hodnota je nula. Nepleťte si to s názvem správce front, který identifikuje správce front (pravděpodobně vzdáleného) pro pojmenovanou frontu.

další záznam ověření
Další objekt této třídy, bez konkrétního pořadí, mající stejný **odkaz na připojení** jako tento objekt. Počáteční hodnota je nula.

heslo
Heslo dodané pro ověření připojení k serveru LDAP CRL.

předchozí záznam ověření
Předchozí objekt této třídy, bez konkrétního pořadí, mající stejný **odkaz na připojení** jako tento objekt. Počáteční hodnota je nula.

typ
Typ ověřovacích informací obsažených v záznamu.

jméno uživatele
Identifikátor uživatele dodaný pro autorizaci k serveru LDAP CRL.

Konstruktory

ImqAuthenticationRecord ();
Výchozí konstruktor.

Metody objektů (veřejné)

void operator = (const ImqAuthenticationRecord & *air*);
Zkopíruje data instance ze *air*a nahradí existující data instance.

const ImqString & connectionName () const;
Vrací **název připojení**.

void setConnectionName (const ImqString & *name*);
Nastaví **název připojení**.

void setConnectionName (const char * *name* = 0);
Nastaví **název připojení**.

ImqQueueManager * connectionReference () const;
Vrací **odkaz na připojení**.

void setConnectionReference (ImqQueueManager & *manager*);
Nastaví **odkaz na připojení**.

void setConnectionReference (ImqQueueManager * *manager* = 0);
Nastaví **odkaz na připojení**.

void copyOut (MQAIR * *pAir*);
Zkopíruje data instance do *pAir*a nahradí existující data instance. To může zahrnovat přidělení závislého úložiště.

void clear (MQAIR * *pAir*);
Vymaže strukturu a uvolní závislé úložiště, na které odkazuje *pAir*.

ImqAuthenticationRecord * nextAuthenticationRecord () const;
Vrátí **další záznam ověření**.

const ImqString & password () const;
Vrátí **heslo**.

void setPassword (const ImqString & *password*);
Nastaví **heslo**.

void setPassword (const char * *password* = 0);
Nastaví **heslo**.

ImqAuthenticationRecord * previousAuthenticationRecord () const;
Vrátí **předchozí záznam ověření**.

Typ MQLONG () const;
Vrátí **typ**.

void setType (const MQLONG *type*);
Nastaví **typ**.

const ImqString & userName () const;
Vrací **jméno uživatele**.

void setUserName (const ImqString & *name*);
Nastaví **jméno uživatele**.

void setUserName (const char * *name* = 0);
Nastaví **jméno uživatele**.

Metody objektů (chráněné)

void setNextAuthenticationRecord (ImqAuthenticationRecord * *pAir* = 0);
Nastaví **další záznam ověření**.

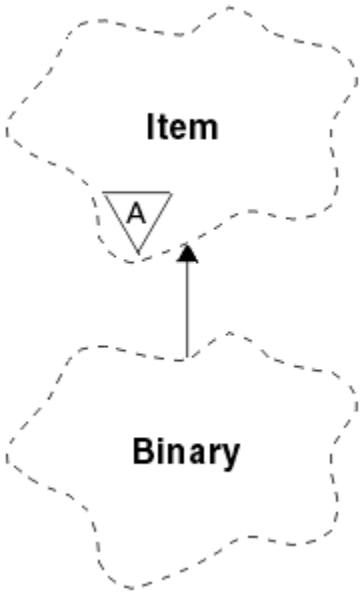
Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nepřeruší seznam ověřovacích záznamů.

void setPreviousAuthenticationRecord (ImqAuthenticationRecord * *pAir* = 0);
Nastaví **předchozí záznam ověření**.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nepřeruší seznam ověřovacích záznamů.

ImqBinary Třída C + +

Tato třída zapouzdřuje binární bajtové pole, které lze použít pro hodnoty ImqMessage **token evidence**, **ID korelace**a **ID zprávy** . Umožňuje snadné přiřazení, kopírování a porovnání.



Obrázek 15. Třída ImqBinary

Atributy objektů

data
Pole bajtů binárních dat. Počáteční hodnota je null.

Délka dat
Počet bajtů. Počáteční hodnota je nula.

ukazatel na data
Adresa prvního bajtu **dat**. Počáteční hodnota je nula.

Konstruktory

ImqBinary( );
Výchozí konstruktor.

ImqBinary(const ImqBinary & *binární*);
Konstruktor kopírování.

ImqBinary(const void * *data*, const size_t *length*);
Kopíruje *délku* bajtů z *dat*.

Přetížené metody ImqItem

virtuální ImqBoolean copyOut (ImqMessage & *msg*);
Zkopíruje **data** do vyrovnávací paměti zpráv a nahradí veškerý existující obsah. Nastaví *msg* **formát** na MQFMT_NONE.

Další podrobnosti viz popis metody třídy ImqItem .

virtuální ImqBoolean pasteIn (ImqMessage & *msg*);
Nastaví **data** přenosem zbývajících dat z vyrovnávací paměti zpráv a nahradí existující **data**.

Aby byla zpráva ImqMessage **ve formátu** úspěšná, musí být MQFMT_NONE.

Další podrobnosti viz popis metody třídy ImqItem .

Metody objektů (veřejné)

void operator = (const ImqBinary & *binární*);
Zkopíruje bajty z *binárního*souboru.

ImqBoolean operátor == (const ImqBinary & *binární*);
Porovná tento objekt s *binárním*. Vrací FALSE, pokud není rovno a TRUE jinak. Objekty se rovnají, pokud mají stejnou **délku dat** a bajty se shodují.

ImqBoolean copyOut (void * *buffer*, const size_t *length*, const char *pad* = 0);
Zkopíruje až *délku* bajtů z **datového ukazatele** do *vyrovnávací paměti*. Pokud je **délka dat** nedostatečná, zbývající prostor ve *vyrovnávací paměti* se naplní *pad* bajty. Hodnota *buffer* může být nula, pokud *length* je také nula. *length* nesmí být záporná. V případě úspěchu vrátí hodnotu TRUE.

size_t dataLength () const ;
Vrací **délku dat**.

ImqBoolean setDataDélka (const size_t *délka*);
Nastaví **délku dat**. Pokud se v důsledku této metody změní **délka dat** , data v objektu se neinicializují. V případě úspěchu vrátí hodnotu TRUE.

void * dataPointer () const ;
Vrací **datový ukazatel**.

ImqBoolean isNull () const ;
Vrátí hodnotu TRUE, pokud je **délka dat** nula nebo pokud jsou všechny **datové** bajty nula. Jinak vrátí hodnotu FALSE.

ImqBoolean set (const void * *buffer*, const size_t *length*);
Kopíruje *délku* bajtů z *vyrovnávací paměti*. V případě úspěchu vrátí hodnotu TRUE.

Metody objektů (chráněné)

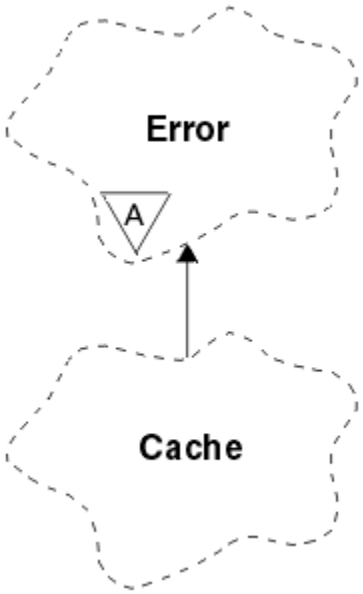
void clear ();
Zmenší **délku dat** na nulu.

Kódy příčin

- MQRC_NO_BUFFER
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_NEKONZISTENTNÍ_FORMÁT

Třída ImqCache C++

Pomocí této třídy můžete zadržet nebo zařadit data do paměti.



Obrázek 16. Třída ImqCache

Pomocí této třídy můžete zadržet nebo zařadit data do paměti. Můžete nominovat vyrovnávací paměť s pevnou velikostí, nebo může systém automaticky poskytnout flexibilní množství paměti. Tato třída souvisí s voláními MQI uvedenými v seznamu “Křížový odkaz ImqCache” na stránce 1780.

- “Atributy objektů” na stránce 1800
- “Konstruktory” na stránce 1800
- “Metody objektů (veřejné)” na stránce 1800
- “Kódy příčin” na stránce 1802

Atributy objektů

automatická vyrovnávací paměť
Označuje, zda je vyrovnávací paměť spravována automaticky systémem (TRUE) nebo je dodána uživatelem (FALSE). Na počátku je nastaven na hodnotu TRUE.

Tento atribut není nastaven přímo. Je nastaven nepřímo buď pomocí metody **useEmptyVyrovnávací paměť** , nebo pomocí metody **useFullVyrovnávací paměť** .

Pokud je zadáno uživatelské úložiště, tento atribut je FALSE, vyrovnávací paměť nemůže růst a může dojít k chybám přetečení vyrovnávací paměti. Adresa a délka vyrovnávací paměti zůstávají konstantní.

Není-li uživatelské úložiště zadáno, má tento atribut hodnotu TRUE a paměť vyrovnávací paměti může přírůstkově narůstat, aby pojmula libovolné množství dat zprávy. Když však vyrovnávací paměť roste, adresa vyrovnávací paměti se může změnit, takže při použití **ukazatele vyrovnávací paměti** a **ukazatele dat**buďte opatrní.

Délka vyrovnávací paměti
Počet bajtů paměti ve vyrovnávací paměti. Počáteční hodnota je nula.

ukazatel vyrovnávací paměti
Adresa vyrovnávací paměti. Počáteční hodnota je null.

Délka dat
Počet bajtů, které jsou za **datovým ukazatelem**. Tato hodnota musí být menší nebo rovna **délce zprávy**. Počáteční hodnota je nula.

Posun dat
Počet bajtů předcházejících **ukazateli dat**. Tato hodnota musí být menší nebo rovna **délce zprávy**. Počáteční hodnota je nula.

ukazatel na data
Adresa části vyrovnávací paměti, do které se má zapisovat nebo která se má číst od dalšího. Počáteční hodnota je null.

délka zprávy
Počet bajtů významných dat ve vyrovnávací paměti. Počáteční hodnota je nula.

Konstruktory

ImqCache( );
Výchozí konstruktor.

ImqCache(const ImqCache & ***mezipaměť*****);**
Konstruktor kopírování.

Metody objektů (veřejné)

void operator = (const ImqCache & ***cache*** **);**
Kopíruje až **délku zprávy** bajtů dat z objektu *mezipaměti* do objektu. Má-li **automatická vyrovnávací paměť** hodnotu FALSE, **délka vyrovnávací paměti** již musí být dostatečná pro uložení zkopírovaných dat.

ImqBoolean automaticBuffer () const ;
Vrátí hodnotu **automatické vyrovnávací paměti** .

size_t bufferLength () const ;
Vrací **délku vyrovnávací paměti**.

char * bufferPointer () const ;
Vrací **ukazatel vyrovnávací paměti**.

void clearMessage ();
Nastaví **délku zprávy** a **posunutí dat** na nulu.

size_t dataLength () const ;
Vrací **délku dat**.

size_t dataOffset () const ;
Vrátí hodnotu **posunutí dat**.

ImqBoolean setDataOffset (const size_t *offset*);
Nastaví **posunutí dat**. **Délka zprávy** se v případě potřeby zvýší, aby se zajistilo, že nebude menší než **posunutí dat**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

char * dataPointer () const ;
Vrátí kopii **datového ukazatele**.

size_t messageLength () const ;
Vrací **délku zprávy**.

ImqBoolean setMessageDélka (const size_t *délka*);
Nastaví **délku zprávy**. Zvýší **délku vyrovnávací paměti** , je-li to nutné, aby **délka zprávy** nebyla větší než **délka vyrovnávací paměti**. V případě potřeby sníží **posunutí dat** , aby se zajistilo, že nebude větší než **délka zprávy**. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean moreBytes (const size_t *bajtů-povinné*);
Zajišťuje, že *požadované bajty* jsou k dispozici více bajtů (pro zápis) mezi **datovým ukazatelem** a koncem vyrovnávací paměti. V případě úspěchu vrátí hodnotu TRUE.

Má-li **automatická vyrovnávací paměť** hodnotu TRUE, je podle potřeby získáno více paměti; jinak již musí být **délka vyrovnávací paměti** dostatečná.

ImqBoolean čtení (const size_t *délka*, char * & *externí vyrovnávací paměť*);
Kopíruje *délku* bajtů z vyrovnávací paměti začínající na pozici **ukazatel dat** do *externí vyrovnávací paměti*. Po zkopírování dat se **posunutí dat** zvýší o *délku*. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean resizeBuffer (const size_t *délka*);
Liší **délku vyrovnávací paměti**za předpokladu, že **automatická vyrovnávací paměť** má hodnotu TRUE. Toho je dosaženo opětovným přidělením vyrovnávací paměti. Do nové vyrovnávací paměti se zkopírují až **délka zprávy** bajtů dat z existující vyrovnávací paměti. Maximální počet zkopírovaných bajtů je *length* bajtů. **Ukazatel vyrovnávací paměti** se změní. **Délka zprávy** a **posunutí dat** jsou zachovány co nejpřesněji v mezích nové vyrovnávací paměti. Vrátí hodnotu TRUE, je-li úspěšná, a hodnotu FALSE, je-li **automatická vyrovnávací paměť** FALSE.

Poznámka: Tato metoda může selhat s MQRC_STORAGE_NOT_AVAILABLE, pokud dojde k problému se systémovými prostředky.

ImqBoolean useEmptyBuffer (const char * *externí-buffer*, const size_t *length*);
Identifikuje prázdnou uživatelskou vyrovnávací paměť a nastaví **ukazatel vyrovnávací paměti** tak, aby ukazoval na *externí vyrovnávací paměť*, **délku vyrovnávací paměti** na *délku*a **délku zprávy** na nulu. Provede **clearMessage**. Pokud je vyrovnávací paměť plně naplněna daty, použijte místo toho metodu **useFullBuffer** . Pokud je vyrovnávací paměť částečně naplněna daty, použijte metodu **setMessageLength** k označení správného množství. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

Tuto metodu lze použít k identifikaci pevného množství paměti, jak bylo popsáno výše (*externí vyrovnávací paměť* nemá hodnotu null a *délka* je nenulová). V takovém případě je **automatická vyrovnávací paměť** nastavena na hodnotu FALSE, nebo ji lze použít k návratu do flexibilní paměti spravované systémem (*externí vyrovnávací paměť* má hodnotu null a *délka* je nula). V takovém případě je volba **automatická vyrovnávací paměť** nastavena na hodnotu TRUE.

ImqBoolean useFullBuffer (const char * *externalBuffer*, const size_t *length*);
Stejně jako u **useEmptyBuffer**, kromě toho, že **délka zprávy** je nastavena na *length*. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean write (const size_t *length*, const char * *external-buffer*);
Kopíruje *délku* bajtů z *externí vyrovnávací paměti*do vyrovnávací paměti začínající na pozici **datového ukazatele** . Po zkopírování dat se **offset dat** zvýší o *délku*a **délka zprávy** se v případě potřeby zvýší, aby se zajistilo, že nebude menší než nová hodnota **offset dat** . Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

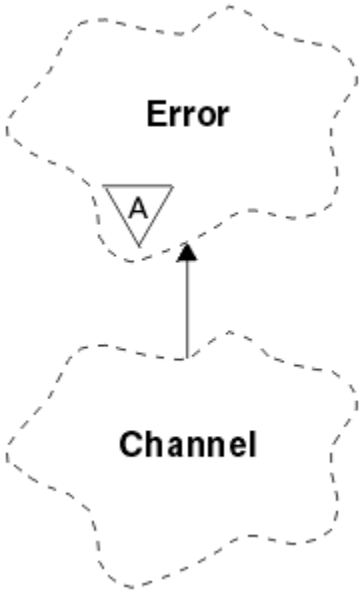
Má-li **automatická vyrovnávací paměť** hodnotu TRUE, je zaručeno dostatečné množství paměti; jinak nesmí konečný **offset dat** překročit **délku vyrovnávací paměti**.

Kódy příčin

- MQRC_BUFFER_NOT_AUTOMATIC
- MQRC_DATA_TRUNCATED
- MQRC_INNEDOSTATEČNÁ_VYROVNÁVACÍ paměť
- MQRC_INDOSTATEČNÁ_DATA
- Ukazatel MQRC_NULL_POINTER
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_ZERO_LENGTH

Třída ImqChannel C++

Tato třída zapouzdřuje definici kanálu (MQCD) pro použití během provádění metody Manager: :connect pro vlastní připojení klienta.



Obrázek 17. Třída ImqChannel

Další podrobnosti viz popis metody Manager: :connect a Ukázkový program HELLO WORLD (imqwrld.cpp).

Ne všechny uvedené metody jsou použitelné pro všechny platformy. Další informace naleznete v popisech příkazů DEFINE CHANNEL a ALTER CHANNEL .

Třída ImqChannel není v systému z/OSpodporována.

- "Atributy objektů" na stránce 1802
- "Konstruktory" na stránce 1804
- "Metody objektů (veřejné)" na stránce 1804
- "Kódy příčin" na stránce 1807

Atributy objektů

dávkový prezenční signál
Počet milisekund mezi kontrolami, zda je vzdálený kanál aktivní. Počáteční hodnota je 0.

Název kanálu
Název kanálu. Počáteční hodnota je null.

Název připojení
Název připojení. Například adresa IP hostitelského počítače. Počáteční hodnota je null.

Komprese záhlaví
Seznam technik komprese dat hlavičky podporovaných kanálem. Počáteční hodnoty jsou všechny nastaveny na MQCOMPRESS_NOT_AVAILABLE.

interval prezenčního signálu
Počet sekund mezi kontrolami, zda připojení stále funguje. Počáteční hodnota je 300.

Interval udržení aktivity
Počet sekund předaných do zásobníku komunikací, který určuje časování udržení aktivity pro kanál. Počáteční hodnota je MQKAI_AUTO.

Lokální adresa
Lokální komunikační adresa kanálu.

Maximální délka zprávy
Maximální délka zprávy podporované kanálem v jedné komunikaci. Počáteční hodnota je 4 194 304.

Komprese zpráv
Seznam technik komprese dat zprávy podporovaných kanálem. Počáteční hodnoty jsou všechny nastaveny na MQCOMPRESS_NOT_AVAILABLE.

Název režimu
Název režimu. Počáteční hodnota je null.

heslo
Heslo dodané pro ověření připojení. Počáteční hodnota je null.

počet uživatelských procedur pro příjem
Počet uživatelských procedur příjmu. Počáteční hodnota je nula. Tento atribut je určen jen pro čtení.

názvy uživatelských procedur pro příjem
Jména uživatelských procedur pro příjem.

příjem uživatelských dat
Data přidružená k uživatelské procedury příjmu.

Název uživatelské procedury zabezpečení zprávy
Název uživatelské procedury zabezpečení, která má být vyvolána na straně serveru připojení. Počáteční hodnota je null.

zabezpečení uživatelských dat
Data, která mají být předána uživatelské proceduře pro zabezpečení zprávy. Počáteční hodnota je null.

počet uživatelských procedur pro odeslání
Počet uživatelských procedur odeslání. Počáteční hodnota je nula. Tento atribut je určen jen pro čtení.

názvy uživatelských procedur pro odeslání
Názvy uživatelských procedur odeslání.

odeslání uživatelských dat
Data přidružená k uživatelské procedury odeslání.

SSL CipherSpec
CipherSpec pro použití s protokolem TLS.

Typ ověřování klienta SSL
Typ ověřování klienta pro použití s protokolem TLS.

Název partnera SSL
Název partnera pro použití s protokolem TLS.

Jméno programu transakce
Název transakčního programu. Počáteční hodnota je null.

Typ přenosu
Typ přenosu připojení. Počáteční hodnota je MQXPT_LU62.

Jméno uživatele
Identifikátor uživatele dodaný pro autorizaci. Počáteční hodnota je null.

Konstruktory

ImqChannel( ) ;
Výchozí konstruktor.

ImqChannel(const ImqChannel & ***channel*** **);**
Konstruktor kopírování.

Metody objektů (veřejné)

void operator = (const ImqChannel & ***channel*** **);**
Zkopíruje data instance z *kanálu*a nahradí veškerá existující data instance.

MQLONG batchHeartBeat () const;
Vrátí **prezenční signál dávky**.

ImqBoolean setBatchHeartBeat(const MQLONG ***prezenční signál*** **= 0L);**
Nastaví **prezenční signál dávky**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString channelName() const;
Vrací **název kanálu**.

ImqBoolean setChannelNázev (const char * ***name*** **= 0);**
Nastaví **název kanálu**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString connectionName() const;
Vrací **název připojení**.

ImqBoolean setConnectionNázev (const char * ***name*** **= 0);**
Nastaví **název připojení**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

size_t headerCompressionPočet () const;
Vrací počet podporovaných technik komprese dat záhlaví.

ImqBoolean headerCompression(počet const size_t, komprese MQLONG []) const;
Vrací kopie podporovaných technik komprese dat záhlaví v **kompresi**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setHeaderKomprese (const size_t count, const MQLONG compress []);

Nastaví podporované techniky komprese dat záhlaví na **compress**.

Nastaví počet podporovaných technik komprese dat záhlaví na **count**.

Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

MQLONG heartBeatInterval () const;
Vrátí **interval prezenčního signálu**.

ImqBoolean setHeartBeatInterval(const MQLONG ***interval*** **= 300L);**
Nastaví **interval prezenčního signálu**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

MQLONG keepAliveInterval () const;
Vrací **interval udržení aktivity**.

ImqBoolean setKeepAliveInterval(const MQLONG ***interval*** **= MQKAI_AUTO);**
Nastaví **interval udržení aktivity**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString localAddress() const;
Vrátí **lokální adresu**.

ImqBoolean setLocalAdresa (const char * ***adresa*** **= 0);**
Nastaví **lokální adresu**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

Délka volání MQLONG maximumMessage() const;
Vrací **maximální délku zprávy**.

ImqBoolean setMaximumMessageLength(const MQLONG *délka* = 4194304L);
Nastaví **maximální délku zprávy**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

size_t messageCompressionPočet () const;
Vrací počet podporovaných technik komprese dat zpráv.

ImqBoolean messageCompression(počet const size_t, komprimace MQLONG []) const;
Vrací kopie podporovaných technik komprese dat zpráv v **kompresi**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setMessageKomprese (const size_t count, const MQLONG compress []);

Nastaví podporované metody komprese dat zpráv na kompresi.

Nastaví počet podporovaných technik komprese dat zpráv na počet.

Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString modeName() const;
Vrátí **název režimu**.

ImqBoolean setModeNázev (const char * *name* = 0);
Nastaví **název režimu**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString password () const;
Vrátí **heslo**.

ImqBoolean setPassword(const char * *heslo* = 0);
Nastaví **heslo**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

size_t receiveExitCount () const;
Vrátí **počet uživatelských procedur pro příjem**.

ImqString receiveExitName ();
Vrátí první z **názvů uživatelských procedur pro příjem**, pokud existují. Je-li **počet uživatelských procedur pro příjem** nula, vrátí prázdný řetězec.

ImqBoolean receiveExitNames (const size_t *count*, ImqString * *names* []);
Vrátí kopie **názvů uživatelských procedur pro příjem** v *názvech*. Nastaví jakékoli *názvy* přesahující **počet uživatelských procedur pro příjem** na řetězce s hodnotou null. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setReceiveExitName(const char * *name* = 0);
Nastaví **názvy uživatelských procedur pro příjem** na jeden *název*. *název* může být prázdný nebo mít hodnotu null. Nastaví **počet uživatelských procedur pro příjem** na hodnotu 1 nebo nula. Vymaže **přijmout uživatelská data**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setReceiveExitNames(const size_t *count*, const char * *names* []);
Nastaví **názvy uživatelských procedur pro příjem** na *názvy*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet uživatelských procedur pro příjem** na *počet*. Vymaže **přijmout uživatelská data**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setReceiveExitNames(const size_t *count*, const ImqString * *names* []);
Nastaví **názvy uživatelských procedur pro příjem** na *názvy*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet uživatelských procedur pro příjem** na *počet*. Vymaže **přijmout uživatelská data**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString receiveUserData ();
Vrátí první z položek **přijmout uživatelská data** , pokud existují. Pokud je **počet uživatelských procedur pro příjem** nula, vrátí prázdný řetězec.

ImqBoolean receiveUserData (const size_t *count*, ImqString * *data* []);
Vrací kopie položek **přijmout uživatelská data** v *datech*. Nastaví jakákoli *data* , která přesahují **počet uživatelských procedur pro příjem** , na řetězce s hodnotou null. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setReceiveUserData(const char * *data* = 0);
Nastaví **přijmout uživatelská data** na jedinou položku *data*. Pokud *data* nemají hodnotu null, **počet uživatelských procedur pro příjem** musí být alespoň 1. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setReceiveUserData(const size_t *count*, const char * *data* []);
Nastaví **přijmout uživatelská data** na *data*. Hodnota *count* nesmí být větší než hodnota **receive exit count**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setReceiveUserData(const size_t *count*, const ImqString * *data* []);
Nastaví **přijmout uživatelská data** na *data*. Hodnota *count* nesmí být větší než hodnota **receive exit count**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString securityExitName () const;
Vrací **název uživatelské procedury zabezpečení**.

ImqBoolean setSecurityExitName(znak připojení * *název* = 0);
Nastaví **název uživatelské procedury zabezpečení**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString securityUserData () const;
Vrací **uživatelská data zabezpečení**.

ImqBoolean setSecurityUserData(const char * *data* = 0);
Nastaví **uživatelská data zabezpečení**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

size_t sendExitCount () const;
Vrátí **počet uživatelských procedur pro odeslání**.

ImqString sendExitName ();
Vrátí první z **názvů uživatelských procedur pro odeslání**, pokud existují. Vrátí prázdný řetězec, pokud je **počet uživatelských procedur pro odeslání** nula.

ImqBoolean sendExitNames (const size_t *count*, ImqString * *names* []);
Vrátí kopie **názvů uživatelských procedur pro odeslání** v *názvech*. Nastaví jakékoli *názvy* , které přesahují **počet uživatelských procedur pro odeslání** , na řetězce s hodnotou null. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setSendExitName(znak const * *název* = 0);
Nastaví **názvy uživatelských procedur pro odeslání** na jeden *název*. *název* může být prázdný nebo mít hodnotu null. Nastaví **počet uživatelských procedur pro odeslání** na hodnotu 1 nebo nula. Vymaže **odeslání uživatelských dat**. Tato metoda vrátí hodnotu TRUE, je-li úspěšná

ImqBoolean setSendExitNames(const size_t *count*, const char * *names* []);
Nastaví **názvy uživatelských procedur pro odeslání** na *jména*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet uživatelských procedur pro odeslání** na *count*. Vymaže **odeslání uživatelských dat**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setSendExitNames(const size_t *count*, const ImqString * *names* []);
Nastaví **názvy uživatelských procedur pro odeslání** na *jména*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet uživatelských procedur pro odeslání** na *count*. Vymaže **odeslání uživatelských dat**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString sendUserData ();
Vrátí první z položek **send user data** (pokud existují). Vrátí prázdný řetězec, pokud je **počet uživatelských procedur pro odeslání** nula.

ImqBoolean sendUserData (const size_t *count*, ImqString * *data* []);
Vrací kopie položek **odeslat uživatelská data** v *datech*. Nastaví jakákoli *data* , která přesahují **počet uživatelských procedur pro odeslání** , na řetězce s hodnotou null. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setSendUserData(znak const * *data* = 0);
Nastaví **odeslání uživatelských dat** na jednotlivou položku *data*. Pokud *data* nemají hodnotu null, **počet uživatelských procedur pro odeslání** musí být alespoň 1. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setSendUserData(const size_t *count*, const char * *data* []);
Nastaví **odeslání uživatelských dat** na *data*. *count* nesmí být větší než **počet uživatelských procedur pro odeslání**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setSendUserData(const size_t *počet*, const ImqString * *data* []);
Nastaví **odeslání uživatelských dat** na *data*. *count* nesmí být větší než **počet uživatelských procedur pro odeslání**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString sslCipherSpecifikace () const;
Vrací specifikaci šifrování TLS.

ImqBoolean setSslCipherSpecification(znak const * *název* = 0);
Nastaví specifikaci šifrování TLS. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

MQLONG sslClientAuthentication () const;
Vrací typ ověřování klienta TLS.

ImqBoolean setSslClientAuthentication(const MQLONG *auth* = MQSCA_REQUIRED);
Nastaví typ ověřování klienta TLS. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString sslPeerName () const;
Vrací název partnera TLS.

ImqBoolean setSslPeerName(const char * *name* = 0);
Nastaví název partnera TLS. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString transactionProgramName () const;
Vrátí **název transakčního programu**.

ImqBoolean setTransactionProgramName(const char * *name* = 0);
Nastaví **název transakčního programu**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

MQLONG transportType() const;
Vrátí **typ přenosu**.

ImqBoolean setTransportTyp (const MQLONG *typ* = MQXPT_LU62);
Nastaví **typ přenosu**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString userId() const;
Vrací **ID uživatele**.

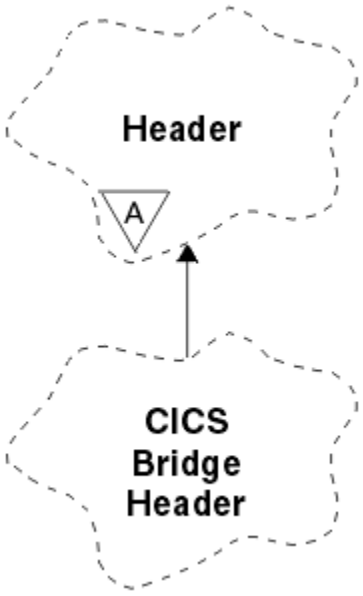
ImqBoolean setUserId (const char * *id* = 0);
Nastaví **ID uživatele**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

Kódy příčin

- MQRC_DATA_LENGTH_ERROR
- MQRC_ITEM_COUNT_ERROR
- Ukazatel MQRC_NULL_POINTER
- MQRC_SOURCE_BUFFER_ERROR

ImqCICSBridgeHeader třída jazyka C++

Tato třída zapouzdřuje specifické funkce datové struktury MQCIH.



Obrázek 18. ImqCICSBridgeHeader

Objekty této třídy používají aplikace, které odesílají zprávy do CICS bridge až IBM MQ for z/OS.

- "Atributy objektů" na stránce 1808
- "Konstruktory" na stránce 1810
- "Přetížené metody ImqItem" na stránce 1810
- "Metody objektů (veřejné)" na stránce 1811
- "Data objektu (chráněná)" na stránce 1813
- "Kódy příčin" na stránce 1813
- "Návratové kódy" na stránce 1813

Atributy objektů

Deskriptor ADS
Deskriptor ADS pro odeslání/příjem. Toto je nastaveno pomocí MQCADSD_NONE. Počáteční hodnota je MQCADSD_NONE. Jsou možné následující další hodnoty:

- MQCADSD_NONE
- MQCADSD_SEND
- MQCADSD_RECV
- MQCADSD_MSGFORMAT

identifikátor upozornění
Klíč AID. Pole musí mít délku MQ_ATTENTION_ID_LENGTH.

ověřovatel
RACF heslo nebo přístupový prvek. Počáteční hodnota obsahuje mezery o délce MQ_AUTHENTICATOR_LENGTH.

kód nestandardního ukončení mostu
Kód nestandardního ukončení mostu o délce MQ_ABEND_CODE_LENGTH. Počáteční hodnota je čtyři prázdné znaky. Hodnota vrácená v tomto poli závisí na návratovém kódu. Další podrobnosti naleznete v části Tabulka 867 na stránce 1813.

kód zrušení mostu
Kód transakce nestandardního ukončení mostu. Pole je vyhrazené, musí obsahovat mezery a musí mít délku MQ_CANCEL_CODE_LENGTH.

kód dokončení mostu
Kód dokončení, který může obsahovat buď kód dokončení IBM MQ , nebo hodnotu CICS EIBRESP. Pole má počáteční hodnotu MQCC_OK. Hodnota vrácená v tomto poli závisí na návratovém kódu. Další podrobnosti naleznete v části Tabulka 867 na stránce 1813.

posunutí chyby mostu
Posunutí chyby mostu. Počáteční hodnota je nula. Tento atribut je určen jen pro čtení.

kód příčiny mostu
Kód příčiny. Toto pole může obsahovat buď hodnotu IBM MQ reason, nebo hodnotu CICS EIBRESP2 . Pole má počáteční hodnotu MQRC_NONE. Hodnota vrácená v tomto poli závisí na návratovém kódu. Další podrobnosti naleznete v části Tabulka 867 na stránce 1813.

návratový kód mostu
Návratový kód ze souboru CICS bridge. Počáteční hodnota je MQCRC_OK.

úloha konverzace
Zda může být úloha konverzační. Počáteční hodnota je MQCCT_NO. Jsou možné následující další hodnoty:

- MQCCT_YES
- MQCCT_NO

pozice kurzoru
Pozice kurzoru. Počáteční hodnota je nula.

čas uchování zařízení
CICS bridge čas uvolnění zařízení.

zařízení jako
Atribut emulovaný terminálem. Pole musí mít délku MQ_FACILITY_LIKE_LENGTH.

token zařízení
Hodnota tokenu BVT. Pole musí mít délku MQ_FACILITY_LENGTH. Počáteční hodnota je MQCFAC_NONE.

funkce
Funkce, která může obsahovat buď název volání IBM MQ , nebo funkci CICS EIBFN. Pole má počáteční hodnotu MQCFUNC_NONE s délkou MQ_FUNCTION_LENGTH. Hodnota vrácená v tomto poli závisí na návratovém kódu. Další podrobnosti naleznete v části Tabulka 867 na stránce 1813.

Následující další hodnoty jsou možné, když **funkce** obsahuje název volání IBM MQ :

- MQCFUNC_MQCONN
- MQCFUNC_MQGET
- MQCFUNC_MQINQ
- MQCFUNC_NONE
- MQCFUNC_MQOPEN
- MQCFUNC_PUT
- MQCFUNC_MQPUT1

interval čekání na získání
Interval čekání na volání MQGET vydané úlohou CICS bridge . Počáteční hodnota je MQCGWI_DEFAULT. Pole se použije pouze v případě, že ovládací prvek **uow control** má hodnotu MQCUOWC_FIRST. Jsou možné následující další hodnoty:

- VÝCHOZÍ-MQCGWI_DEFAULT
- MQWI_UNLIMITED

Typ odkazu
Typ odkazu. Počáteční hodnota je MQCLT_PROGRAM. Jsou možné následující další hodnoty:

- MQCLT_PROGRAM
- MQCLT_TRANSACTION

další identifikátor transakce
ID další transakce, která se má připojit. Pole musí mít délku MQ_TRANSACTION_ID_LENGTH.

délka výstupních dat
Délka dat COMMAREA. Počáteční hodnota je MQCODL_AS_INPUT.

formát odpovědi
Název formátu zprávy odpovědi. Počáteční hodnota je MQFMT_NONE s délkou MQ_FORMAT_LENGTH.

kód spuštění
Kód spuštění transakce. Pole musí mít délku MQ_START_CODE_LENGTH. Počáteční hodnota je MQCSC_NONE. Jsou možné následující další hodnoty:

- MQCSC_START
- MQCSC_STARTDATA
- MQCSC_TERMINPUT
- MQCSC_NONE

stav ukončení úlohy
Stav ukončení úlohy. Počáteční hodnota je MQCTES_NOSYNC. Jsou možné následující další hodnoty:

- MQCTES_COMMIT
- MQCTES_BACKOUT
- MQCTES_ENDTASK
- MQCTES_NOSYNC

Identifikátor transakce
ID transakce, která se má připojit. Počáteční hodnota musí obsahovat mezery a musí mít délku MQ_TRANSACTION_ID_LENGTH. Pole se použije pouze v případě, že ovládací prvek **uow** má hodnotu MQCUOWC_FIRST nebo MQCUOWC_ONLY.

Ovládací prvek UOW
Ovládací prvek UOW. Počáteční hodnota je MQCUOWC_ONLY. Jsou možné následující další hodnoty:

- MQCUOWC_FIRST
- MQCUOWC_MIDDLE
- MQCUOWC_LAST
- MQCUOWC_ONLY
- MQCUOWC_COMMIT
- MQCUOWC_BACKOUT
- MQCUOWC_CONTINUE

verze
Číslo verze MQCIH. Počáteční hodnota je MQCIH_VERSION_2. Jedinou další podporovanou hodnotou je hodnota MQCIH_VERSION_1.

Konstruktory

ImqCICSBridgeHeader();
Výchozí konstruktor.

ImqCICSBridgeHeader(const ImqCICSBridgeHeader & *záhlaví*);
Konstruktor kopírování.

Přetížené metody ImqItem

virtuální ImqBoolean copyOut(ImqMessage & *msg*);
Vloží datovou strukturu MQCIH do vyrovnávací paměti zpráv na začátku, přesune existující data zprávy dále a nastaví formát zprávy na MQFMT_CICS.

Další podrobnosti viz popis metody nadřízené třídy.

virtuální ImqBoolean pasteIn(ImqMessage & *msg*);
Čte datovou strukturu MQCIH z vyrovnávací paměti zpráv. Aby bylo kódování objektu *msg* úspěšné, musí být MQENC_NATIVE. Načíst zprávy s parametrem MQGMO_CONVERT na MQENC_NATIVE. Chcete-li být úspěšní, formát ImqMessage musí být MQFMT_CICS.

Další podrobnosti viz popis metody nadřízené třídy.

Metody objektů (veřejné)

void operator = (const ImqCICSBridgeHeader & *záhlaví*);
Zkopíruje data instance ze *záhlaví*a nahradí existující data instance.

MQLONG ADSDescriptor () const;
Vrací kopii **deskriptoru ADS**.

void setADSDescriptor(const MQLONG *descriptor* = MQCADSD_NONE);
Nastaví **deskriptor ADS**.

ImqString attentionIdentifier() const;
Vrací kopii **identifikátoru upozornění**vyplněnou koncovými mezerami na délku MQ_ATTENTION_ID_LENGTH.

void setAttentionIdentifier (const char * *data* = 0);
Nastaví **identifikátor upozornění**doplněný koncovými mezerami na délku MQ_ATTENTION_ID_LENGTH. Pokud nejsou dodána žádná *data* , nastaví **identifikátor upozornění** na počáteční hodnotu.

ImqString authenticator () const;
Vrací kopii **ověřovatele**vyplněnou koncovými mezerami na délku MQ_AUTHENTICATOR_LENGTH.

void setAuthenticator(const char * *data* = 0);
Nastaví **ověřovatele**vyplněný koncovými mezerami na délku MQ_AUTHENTICATOR_LENGTH. Pokud nejsou dodána žádná *data* , resetuje **ověřovatele** na počáteční hodnotu.

ImqString bridgeAbendKód () const;
Vrací kopii **kódu nestandardního ukončení mostu**vyplněnou koncovými mezerami na délku MQ_ABEND_CODE_LENGTH.

ImqString bridgeCancelKód () const;
Vrací kopii **kódu zrušení mostu**vyplněnou koncovými mezerami na délku MQ_CANCEL_CODE_LENGTH.

void setBridgeCancelCode(const char * *data* = 0);
Nastaví **kód zrušení mostu**doplněný koncovými mezerami na délku MQ_CANCEL_CODE_LENGTH. Pokud nejsou dodána žádná *data* , resetuje **kód zrušení mostu** na počáteční hodnotu.

Kód MQLONG bridgeCompletion() const;
Vrátí kopii **kódu dokončení mostu**.

Odsazení () funkce MQLONG bridgeError const;
Vrací kopii **offsetu chyb mostu**.

MQLONG bridgeReasonKód () const;
Vrací kopii **kódu příčiny mostu**.

MQLONG bridgeReturnKód () const;
Vrátí **návratový kód mostu**.

MQLONG conversationalTask() const;
Vrací kopii **konverzační úlohy**.

void setConversationalTask (const MQLONG *task* = MQCCT_NO);
Nastaví **úlohu konverzace**.

MQLONG cursorPosition() const;
Vrátí kopii **pozice kurzoru**.

void setCursorPozice (const MQLONG *pozice* = 0);
Nastaví **pozici kurzoru**.

Čas operace MQLONG facilityKeep() const;
Vrátí kopii **času uchování zařízení**.

void setFacilityKeepTime(const MQLONG *time* = 0);
Nastaví **dobu uchování zařízení**.

ImqString facilityLike() const;
Vrací kopii zařízení **, jako je**, vyplněnou koncovými mezerami na délku MQ_FACILITY_LIKE_LENGTH.

void setFacilityLike (const char * *name* = 0);
Nastaví vlastnost **jako**doplněnou koncovými mezerami na délku MQ_FACILITY_LIKE_LENGTH. Není-li zadán žádný *název* , resetuje **zařízení jako** počáteční hodnotu.

ImqBinary facilityToken() const;
Vrací kopii **tokenu zařízení**.

ImqBoolean setFacility(const ImqBinary & *token*);
Nastaví **token prostředku**. **Délka dat** *tokenu* musí být buď nula, nebo MQ_FACILITY_LENGTH.
V případě úspěchu vrátí hodnotu TRUE.

void setFacilityToken (const MQBYTE8 *token* = 0);
Nastaví **token prostředku**. *token* může být nula, což je stejné jako určení MQCFAC_NONE.
Pokud je *token* nenulový, musí adresovat bajty MQ_FACILITY_LENGTH binárních dat. Při použití předdefinovaných hodnot, jako např. MQCFAC_NONE, budete možná muset provést přetypování, abyste zajistili shodu podpisu. Například (MQBYTE *) MQCFAC_NONE.

ImqString funkce () const;
Vrací kopii **funkce**vyplněnou koncovými mezerami na délku MQ_FUNCTION_LENGTH.

MQLONG getWaitInterval () const;
Vrací kopii **intervalu čekání na získání**.

void setGetWaitInterval(const MQLONG *interval* = MQCGWI_DEFA
Nastaví **interval čekání na získání**.

MQLONG linkType() const;
Vrací kopii **typu odkazu**.

void setLinkType (const MQLONG *type* = MQCLT_PROGRAM);
Nastaví **typ odkazu**.

ImqString nextTransactionIdentifier () const;
Vrací kopii dat **identifikátoru další transakce** vyplněnou koncovými mezerami na délku MQ_TRANSACTION_ID_LENGTH.

MQLONG outputDataDélka () const;
Vrací kopii **délky výstupních dat**.

void setOutputDataLength(const MQLONG *length* = MQCODL_AS_INPUT);
Nastaví **délku výstupních dat**.

ImqString replyToFormat () const;
Vrací kopii názvu **formátu odpovědi** vyplněnou koncovými mezerami na délku MQ_FORMAT_LENGTH.

void setReplyToFormat(const char * *název* = 0);
Nastaví **formát odpovědi**doplněný koncovými mezerami na délku MQ_FORMAT_LENGTH. Není-li zadán žádný *název* , nastaví **formát odpovědi** na počáteční hodnotu.

ImqString startCode() const;
Vrací kopii **počátečního kódu**doplněnou koncovými mezerami na délku MQ_START_CODE_LENGTH.

void setStartCode (const char * *data* = 0);
Nastaví data **počátečního kódu** doplněná koncovými mezerami na délku MQ_START_CODE_LENGTH.
Pokud nejsou dodána žádná *data* , resetuje **počáteční kód** na počáteční hodnotu.

Stav MQLONG taskEnd() const;
Vrátí kopii **stavu ukončení úlohy**.

ImqString transactionIdentifier() const;
Vrací kopii dat **identifikátoru transakce** vyplněnou koncovými mezerami na délku MQ_TRANSACTION_ID_LENGTH.

void setTransactionIdentifikátor (const char * *data* = 0);
Nastaví **identifikátor transakce**doplněný koncovými mezerami na délku MQ_TRANSACTION_ID_LENGTH. Pokud nejsou dodána žádná *data* , resetuje **identifikátor transakce** na počáteční hodnotu.

MQLONG UOWControl () const;
Vrací kopii **ovládacího prvku UOW**.

void setUOWControl(const MQLONG *control* = MQCUOWC_ONLY);
Nastaví **ovládací prvek UOW**.

MQLONG verze () const;
Vrátí číslo **verze** .

ImqBoolean setVersion(const MQLONG *verze* = MQCIH_VERSION_2);
Nastaví číslo **verze** . V případě úspěchu vrátí hodnotu TRUE.

Data objektu (chráněná)

MQLONG *olVersion*
Maximální číslo verze MQCIH, které lze umístit do úložiště přiděleného pro *opcih*.

PMQCIH *opcih*
Adresa datové struktury MQCIH. Množství přiděleného úložiště je označeno hodnotou *olVersion*.

Kódy příčin

- MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_WRONG_VERSION

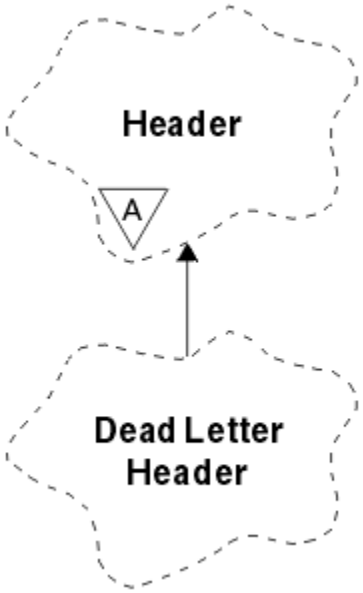
Návratové kódy

Tabulka 867. Návratové kódy třídy ImqCICSBridgeHeader

Návratový kód	Funkce	CompCode	Příčina	Kód nestandardního ukončení
MQCRC_OK				
MQCRC_BRIDGE_ERROR			MQFB_CICS:	
MQCRC_MQ_API_ERROR	IBM MQ název volání	IBM MQ CompCode	IBM MQ Příčina	
MQCRC_BRIDGE_TIMEOUT	IBM MQ název volání	IBM MQ CompCode	IBM MQ Příčina	
MQCRC_CICS_EXEC_ERROR	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_SECURITY_ERROR	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_PROGRAM_NENÍ k dispozici	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_TRANSID_NOT_AVAILABLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_BRIDGE_ABEND				CICS ABCODE
MQCRC_APPLICATION_ABEND				CICS ABCODE

ImqDeadLetterHeader třída C++

Tato třída zapouzdřuje funkce datové struktury MQDLH.



Obrázek 19. ImqDeadLetterHeader

Objekty této třídy jsou obvykle používány aplikací, která zjistí zprávu, kterou nelze zpracovat. Nová zpráva obsahující záhlaví nedoručených zpráv a obsah zprávy je umístěn do fronty nedoručených zpráv a zpráva je vyřazena.

- "Atributy objektů" na stránce 1814
- "Konstruktory" na stránce 1815
- "Přetížené metody ImqItem" na stránce 1815
- "Metody objektů (veřejné)" na stránce 1815
- "Data objektu (chráněná)" na stránce 1816
- "Kódy příčin" na stránce 1816

Atributy objektů

kód příčiny nedoručených dopisů
Příčina, proč zpráva dorazila do fronty nedoručených zpráv. Počáteční hodnota je MQRC_NONE.

Název správce cílových front
Název původního cílového správce front. Název je řetězec délky MQ_Q_MGR_NAME_LENGTH. Jeho počáteční hodnota je null.

název cílové fronty
Název původní cílové fronty. Název je řetězec délky MQ_Q_NAME_LENGTH. Jeho počáteční hodnota je null.

Název vkládající aplikace
Název aplikace, která vložila zprávu do fronty nedoručených zpráv. Název je řetězec délky MQ_PUT_APPL_NAME_LENGTH. Jeho počáteční hodnota je null.

Typ vkládající aplikace
Typ aplikace, která vložila zprávu do fronty nedoručených zpráv. Počáteční hodnota je nula.

Datum vložení
Datum, kdy byla zpráva vložena do fronty nedoručených zpráv. Datum je řetězec délky MQ_PUT_DATE_LENGTH. Jeho počáteční hodnota je řetězec s hodnotou null.

Čas vložení
Čas, kdy byla zpráva vložena do fronty nedoručených zpráv. Čas je řetězec délky MQ_PUT_TIME_LENGTH. Jeho počáteční hodnota je řetězec s hodnotou null.

Konstruktory

ImqDeadLetterHeader( );
Výchozí konstruktor.

ImqDeadLetterHeader(const ImqDeadLetterHeader & *záhlaví*);
Konstruktor kopírování.

Přetížené metody ImqItem

virtuální ImqBoolean copyOut (ImqMessage & *msg*);
Vloží datovou strukturu MQDLH do vyrovnávací paměti zpráv na začátku, přičemž stávající data zpráv přesunou dále. Nastaví formát *msg* na hodnotu MQFMT_DEAD_LETTER_HEADER.

Další podrobnosti viz popis metody třídy ImqHeader na stránce "Třída ImqHeader C++" na stránce 1822 .

virtuální ImqBoolean pasteIn (ImqMessage & *msg*);
Čte datovou strukturu MQDLH z vyrovnávací paměti zpráv.

Chcete-li být úspěšní, formát ImqMessage musí být MQFMT_DEAD_LETTER_HEADER.

Další podrobnosti viz popis metody třídy ImqHeader na stránce "Třída ImqHeader C++" na stránce 1822 .

Metody objektů (veřejné)

void operator = (const ImqDeadLetterHeader & *záhlaví*);
Zkopíruje data instance z *záhlaví*a nahradí existující data instance.

MQLONG deadLetterReasonCode () const;
Vrátí kód příčiny nedoručených písmen.

void setDeadLetterReasonKód (const MQLONG *příčina*);
Nastaví kód příčiny nedoručených písmen.

ImqString destinationQueueManagerName () const;
Vrací název cílového správce front, zbavený všech koncových mezer.

void setDestinationQueueManagerName (const char * *name*);
Nastaví název správce cílové fronty. Ořízne data delší než MQ_Q_MGR_NAME_LENGTH (48 znaků).

ImqString destinationQueueName () const;
Vrací kopii názvu cílové fronty, zbavenou všech koncových mezer.

void setDestinationQueueName (const char * *name*);
Nastaví název cílové fronty. Ořízne data delší než MQ_Q_NAME_LENGTH (48 znaků).

ImqString putApplicationNázev () const;
Vrací kopii názvu vkládající aplikace, zbavenou všech koncových mezer.

void setPutApplicationName (const char * *name* = 0);
Nastaví název vkládné aplikace. Ořízne data delší než MQ_PUT_APPL_NAME_LENGTH (28 znaků).

MQLONG putApplicationTyp () const;
Vrací typ vkládací aplikace.

void setPutApplicationType (const MQLONG *typ* = MQAT_NO_CONTEXT);
Nastaví typ aplikace vložení.

ImqString putDate () const;
Vrací kopii data vložení, zbavenou všech koncových mezer.

void setPutDate (const char * *date* = 0);
Nastaví datum vložení. Ořízne data delší než MQ_PUT_DATE_LENGTH (8 znaků).

ImqString putTime () const;
Vrací kopii času vložení, zbavenou všech koncových mezer.

void setPutČas (const char * *čas* = 0);
Nastaví čas vložení. Ořízne data delší než MQ_PUT_TIME_LENGTH (8 znaků).

Data objektu (chráněná)

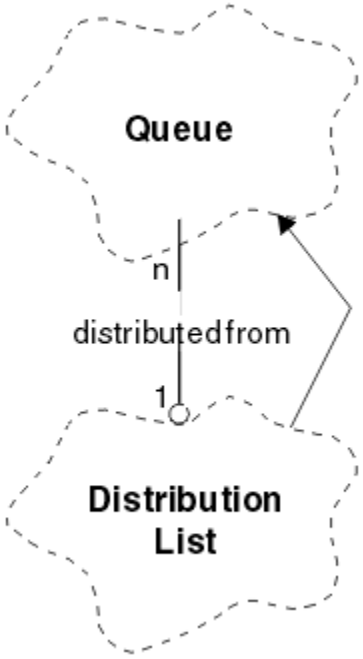
MQDLH *omqdlh*
Datová struktura MQDLH.

Kódy příčin

- MQRC_NEKONZISTENTNÍ_FORMÁT
- MQRC_STRUC_ID_ERROR
- MQRC_ENCODING_ERROR

ImqDistributionVypsat třídu C++

Tato třída zapouzdřuje dynamický distribuční seznam, který odkazuje na jednu nebo více front za účelem odeslání zprávy nebo zpráv do více míst určení.



Obrázek 20. ImqDistributionVypsat třídu

- "Atributy objektů" na stránce 1816
- "Konstruktory" na stránce 1817
- "Metody objektů (veřejné)" na stránce 1817
- "Metody objektů (chráněné)" na stránce 1817

Atributy objektů

první distribuovaná fronta
První z jednoho nebo více objektů třídy, bez konkrétního pořadí, ve kterém **odkaz na rozdělovník** adresuje tento objekt.

Zpočátku neexistují žádné takové objekty. Chcete-li úspěšně otevřít seznam ImqDistribution, musí existovat alespoň jeden takový objekt.

Poznámka: Když se otevře objekt seznamu ImqDistribution, všechny otevřené objekty, které na něj odkazují, se automaticky zavřou.

Konstruktory

ImqDistributionList ();
Výchozí konstruktor.

ImqDistributionList (const ImqDistributionList & *list*);
Konstruktor kopírování.

Metody objektů (veřejné)

void operator = (const ImqDistributionList & *list*);
Všechny objekty, které odkazují na **tento** objekt, jsou před kopírováním vyhodnoceny. Po vyvolání této metody nebudou na **tento** objekt odkazovat žádné objekty.

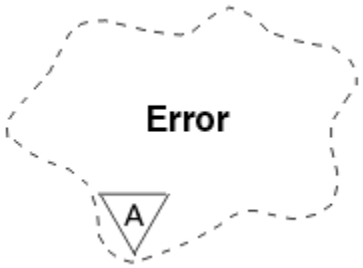
*** firstDistributedFronta () const ;**
Vrací **první distribuovanou frontu**.

Metody objektů (chráněné)

void setFirstDistributedQueue (* *queue* = 0);
Nastaví **první distribuovanou frontu**.

ImqError Třída C + +

Tato abstraktní třída poskytuje informace o chybách přidružených k objektu.



Obrázek 21. Třída ImqError

- "Atributy objektů" na stránce 1817
- "Konstruktory" na stránce 1818
- "Metody objektů (veřejné)" na stránce 1818
- "Metody objektů (chráněné)" na stránce 1818
- "Kódy příčin" na stránce 1818

Atributy objektů

kód dokončení
Nejnovější kód dokončení. Počáteční hodnota je nula. Jsou možné následující další hodnoty:

- MQCC_OK
- MQCC_VAROVÁNÍ
- MQCC_FAILED

kód příčiny
Nejnovější kód příčiny. Počáteční hodnota je nula.

Konstruktory

ImqError( );
Výchozí konstruktor.

ImqError(const ImqError & ***chyba*** **);**
Konstruktor kopírování.

Metody objektů (veřejné)

void operator = (const ImqError & ***error*** **);**
Zkopíruje data instance z *error*a nahradí existující data instance.

void clearErrorKódy ();
Nastaví **kód dokončení** a **kód příčiny** na nulu.

MQLONG completionCode () const ;
Vrátí **kód dokončení**.

MQLONG reasonCode () const ;
Vrátí **kód příčiny**.

Metody objektů (chráněné)

ImqBoolean checkReadPointer (const void * ***ukazatel*****, const size_t** ***délka*** **);**
Ověří, zda je kombinace ukazatele a délky platná pro přístup jen pro čtení, a v případě úspěchu vrátí hodnotu TRUE.

ImqBoolean checkWritePointer (const void * ***ukazatel*****, const size_t** ***délka*** **);**
Ověří, zda je kombinace ukazatele a délky platná pro přístup pro čtení a zápis, a v případě úspěchu vrátí hodnotu TRUE.

void setCompletionCode (const MQLONG ***code*** **= 0);**
Nastaví **kód dokončení**.

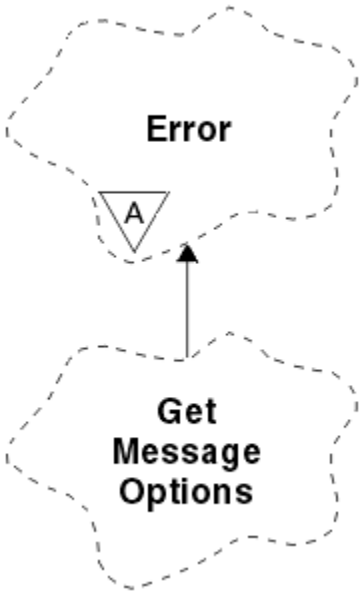
void setReasonKód (const MQLONG ***kód*** **= 0);**
Nastaví **kód příčiny**.

Kódy příčin

- MQRC_BUFFER_ERROR

ImqGetMessageOptions třída C++

Tato třída zapouzdřuje datovou strukturu MQGMO



Obrázek 22. Třída ImqGetMessageOptions

Atributy objektů

stav skupiny
Stav zprávy pro skupinu zpráv. Počáteční hodnota je MQGS_NOT_IN_GROUP. Jsou možné následující další hodnoty:

- MQGS_MSG_IN_GROUP
- MQGS_LAST_MSG_IN_GROUP

volby shody
Volby pro výběr příchozích zpráv. Počáteční hodnota je MQMO_MATCH_MSG_ID | MQMO_MATCH_CORREL_ID. Jsou možné následující další hodnoty:

- MQMO_GROUP_ID
- MQMO_MATCH_MSG_SEQ_NUMBER
- MQMO_MATCH_OFFSET
- MQMO_MSG_TOKEN
- MQMO_NONE

token zprávy
Token zprávy. Binární hodnota (MQBYTE16) délky MQ_MSG_TOKEN_LENGTH. Počáteční hodnota je MQMTOK_NONE.

volby
Volby použitelné pro zprávu. Počáteční hodnota je MQGMO_NO_WAIT. Jsou možné následující další hodnoty:

- 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

vyřešený název fronty
Vyřešený název fronty. Tento atribut je určen jen pro čtení. Názvy nejsou nikdy delší než 48 znaků a mohou být doplněny do této délky nulami. Počáteční hodnota je řetězec s hodnotou Null.

vrácená délka
Vrácená délka. Počáteční hodnota je MQRL_UNDEFINED. Tento atribut je určen jen pro čtení.

segmentace
Schopnost segmentovat zprávu. Počáteční hodnota je MQSEG_INHIBITED. Je možná další hodnota MQSEG_ALLOWED.

stav segmentu
Stav segmentace zprávy. Počáteční hodnota je MQSS_NOT_A_SEGMENT. Jsou možné následující další hodnoty:

- MQSS_SEGMENT
- MQSS_LAST_SEGMENT

účast synchronizačního bodu
TRUE, když jsou zprávy načteny pod řízením synchronizačního bodu.

Interval čekání
Doba, po kterou se metoda get třídy pozastaví při čekání na příchod vhodné zprávy, pokud již není k dispozici. Počáteční hodnota je nula, což má vliv na neurčité čekání. Je možná další hodnota MQWI_UNLIMITED. Tento atribut je ignorován, pokud volby nezahrnují MQGMO_WAIT.

Konstruktory

ImqGetMessageOptions( );
Výchozí konstruktor.

ImqGetMessageOptions(const ImqGetMessageOptions & ***gmo*** **);**
Konstruktor kopírování.

Metody objektů (veřejné)

void operator = (const ImqGetMessageOptions & ***gmo*** **);**
Zkopíruje data instance z *gmo*a nahradí existující data instance.

MQCHAR groupStatus () const;
Vrátí stav skupiny.

void setGroupStatus (const MQCHAR *status*);
Nastaví stav skupiny.

MQLONG matchOptions () const;
Vrátí volby shody.

void setMatchOptions (const MQLONG *options*);
Nastaví volby shody.

ImqBinary messageToken() const;
Vrátí token zprávy.

ImqBoolean setMessage(const ImqBinary & *token*);
Nastaví token zprávy. Délka dat *tokenu* musí být buď nula, nebo MQ_MSG_TOKEN_LENGTH. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

void setMessageToken (const MQBYTE16 *token* = 0);
Nastaví token zprávy. *token* může být nula, což je stejné jako zadání MQMTOK_NONE. Je-li *token* nenulový, musí adresovat MQ_MSG_TOKEN_LENGTH bajtů binárních dat.

Při použití předdefinovaných hodnot, jako např. MQMTOK_NONE, možná nebudete muset provést přetypování, abyste zajistili shodu podpisu, například (MQBYTE *) MQMTOK_NONE.

Volby MQLONG () const;
Vrátí volby.

void setOptions (const MQLONG *options*);
Nastaví volby včetně hodnoty účasti synchronizačního bodu.

ImqString resolvedQueueName () const;
Vrátí kopii rozlišeného názvu fronty.

MQLONG returnedLength() const;
Vrátí vrácenou délku.

MQCHAR segmentace () const;
Vrátí segmentaci.

void setSegmentation (const MQCHAR *value*);
Nastaví segmentaci.

MQCHAR segmentStatus () const;
Vrátí stav segmentu.

void setSegmentStatus (const MQCHAR *status*);
Nastaví stav segmentu.

ImqBoolean syncPointParticipace () const;
Vrací hodnotu účasti synchronizačního bodu, která je TRUE, pokud volby zahrnují buď MQGMO_SYNCPOINT, nebo MQGMO_SYNCPOINT_IF_PERSISTENT.

void setSyncPointParticipation (const ImqBoolean *sync*);
Nastaví hodnotu účasti synchronizačního bodu. Má-li parametr *sync* hodnotu TRUE, změní volby tak, aby zahrnovaly MQGMO_SYNCPOINT a vyloučily jak MQGMO_NO_SYNCPOINT, tak MQGMO_SYNCPOINT_IF_PERSISTENT. Je-li hodnota *sync* FALSE, změní volby tak, aby zahrnovaly MQGMO_NO_SYNCPOINT a vyloučily jak MQGMO_SYNCPOINT, tak MQGMO_SYNCPOINT_IF_PERSISTENT.

MQLONG waitInterval () const;
Vrátí interval čekání.

void setWaitInterval (const MQLONG *interval*);
Nastaví interval čekání.

Metody objektů (chráněné)

statická void setVersionSupported (const MQLONG);
Nastaví verzi MQGMO. Výchozí hodnota je MQGMO_VERSION_3.

Data objektu (chráněná)

MQGMO *omqgmo*
Datová struktura MQGMO verze 2. Přístup k polím MQGMO podporovaným pouze pro MQGMO_VERSION_2 .

PMQGMO *opgmo*
Adresa datové struktury MQGMO. Číslo verze pro tuto adresu je označeno v *olVersion*. Před přístupem k polím MQGMO zkontrolujte číslo verze, abyste se ujistili, že jsou přítomny.

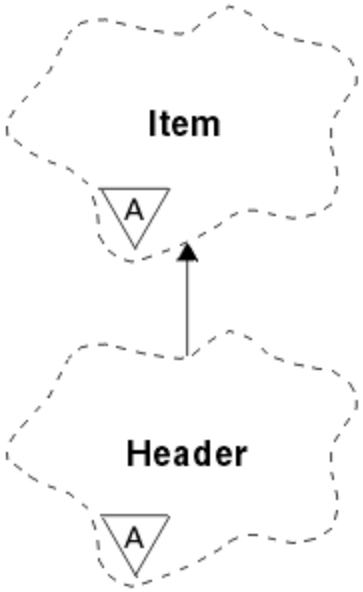
MQLONG *olVersion*
Číslo verze datové struktury MQGMO adresované pomocí *opgmo*.

Kódy příčin

- MQRC_BINARY_DATA_LENGTH_ERROR

Třída ImqHeader C++

Tato abstraktní třída zapouzdřuje společné vlastnosti datové struktury MQDLH.



Obrázek 23. Třída ImqHeader

- "Atributy objektů" na stránce 1822
- "Konstruktory" na stránce 1823
- "Metody objektů (veřejné)" na stránce 1823

Atributy objektů

znaková sada
Původní identifikátor kódované znakové sady. Na počátku MQCCSI_Q_MGR.

kódování
Původní kódování. Zpočátku MQENC_NATIVE.

formát
Původní formát. Počáteční hodnota MQFMT_NONE.

příznaky záhlaví
Počáteční hodnoty jsou:

- Nula pro objekty třídy ImqDeadLetterHeader
- MQIIH_NONE pro objekty třídy ImqIMSBridgeHeader
- MQRMHF_LAST pro objekty třídy záhlaví ImqReference
- MQCIH_NONE pro objekty třídy ImqCICSBridgeHeader
- MQWIH_NONE pro objekty třídy záhlaví ImqWork

Konstruktory

ImqHeader( );
Výchozí konstruktor.

ImqHeader(const ImqHeader & ***záhlaví*** **);**
Konstruktor kopírování.

Metody objektů (veřejné)

void operator = (const ImqHeader & ***záhlaví*** **);**
Zkopíruje data instance ze záhlaví *header*a nahradí existující data instance.

virtuální MQLONG characterSet () const ;
Vrátí **znakovou sadu**.

virtuální void setCharacterSet (const MQLONG ***ccsid*** **= MQCCSI_Q_MGR);**
Nastaví **znakovou sadu**.

virtuální MQLONG kódování () const ;
Vrací **kódování**.

virtuální void setEncoding (const MQLONG ***encoding*** **= MQENC_NATIVE);**
Nastaví **kódování**.

virtuální ImqString formát () const ;
Vrací kopii **formátu**, včetně koncových mezer.

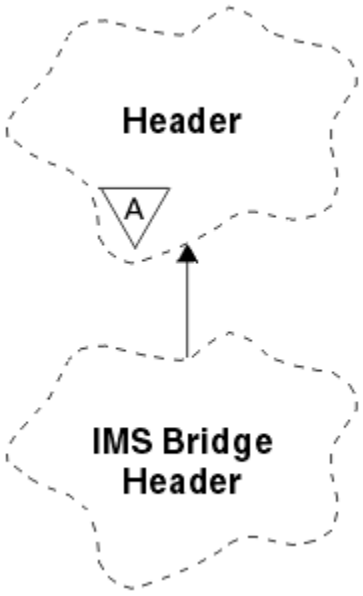
virtuální void setFormat (const char * ***name*** **= 0);**
Nastaví **formát**, vyplněný na 8 znaků s koncovými mezerami.

virtuální MQLONG headerFlags () const ;
Vrací **příznaky záhlaví**.

virtuální void setHeaderPříznaky (const MQLONG ***flags*** **= 0);**
Nastaví **příznaky záhlaví**.

ImqIMSBridgeHeader třída jazyka C++

Tato třída zapouzdřuje funkce datové struktury MQIIH.



Obrázek 24. ImqIMSBridgeHeader

Objekty této třídy používají aplikace, které odesílají zprávy na most IMS přes IBM MQ for z/OS.

Poznámka: Znaková sada a kódování ImqHeader musí mít výchozí hodnoty a nesmí být nastaveny na žádné jiné hodnoty.

- "Atributy objektů" na stránce 1824
- "Konstruktory" na stránce 1825
- "Přetížené metody ImqItem" na stránce 1825
- "Metody objektů (veřejné)" na stránce 1825
- "Data objektu (chráněná)" na stránce 1826
- "Kódy příčin" na stránce 1826

Atributy objektů

ověřovatel
RACF heslo nebo přístupový prvek o délce MQ_AUTHENTICATOR_LENGTH. Počáteční hodnota je MQIAUT_NONE.

režim potvrzení
Režim potvrzení. Další informace o režimech potvrzení IMS naleznete v příručce *OTMA User's Guide* . Počáteční hodnota je MQICM_COMMIT_THEN_SEND. Je možná další hodnota MQICM_SEND_THEN_COMMIT.

potlačení logického terminálu
Přepis logického terminálu o délce MQ_LTERM_OVERRIDE_LENGTH. Počáteční hodnota je řetězec s hodnotou Null.

název mapy služeb formátu zpráv
Název mapy MFS o délce MQ_MFS_MAP_NAME_LENGTH. Počáteční hodnota je řetězec s hodnotou Null.

formát odpovědi
Formát libovolné odpovědi o délce MQ_FORMAT_LENGTH. Počáteční hodnota je MQFMT_NONE.

rozsah zabezpečení
Rozsah zpracování zabezpečení IMS . Počáteční hodnota je MQISS_CHECK. Je možná další hodnota MQISS_FULL.

id instance transakce
Identita instance transakce, binární (MQBYTE16) hodnota délky MQ_TRAN_INSTANCE_ID_LENGTH. Počáteční hodnota je MQITII_NONE.

Stav transakce
Stav konverzace IMS . Počáteční hodnota je MQITS_NOT_IN_CONVERSATION. Je možná další hodnota MQITS_IN_CONVERSATION.

Konstruktory

ImqIMSBridgeHeader();
Výchozí konstruktor.

ImqIMSBridgeHeader(const ImqIMSBridgeHeader & *záhlaví*);
Konstruktor kopírování.

Přetížené metody ImqItem

virtuální ImqBoolean copyOut (ImqMessage & *msg*);
Na začátku vloží datovou strukturu MQIIH do vyrovnávací paměti zpráv a dále přesouvá existující data zpráv. Nastaví formát *msg* na MQFMT_IMS.

Další podrobnosti viz popis metody nadřízené třídy.

virtuální ImqBoolean pasteIn (ImqMessage & *msg*);
Čte datovou strukturu MQIIH z vyrovnávací paměti zpráv.

Aby bylo kódování objektu *msg* úspěšné, musí být MQENC_NATIVE. Načíst zprávy s parametrem MQGMO_CONVERT na MQENC_NATIVE.

Chcete-li být úspěšní, formát ImqMessage musí být MQFMT_IMS.

Další podrobnosti viz popis metody nadřízené třídy.

Metody objektů (veřejné)

void operator = (const ImqIMSBridgeHeader & *záhlaví*);
Zkopíruje data instance ze záhlaví *header*a nahradí existující data instance.

ImqString authenticator () const;
Vrací kopii ověřovatele vyplněnou koncovými mezerami na délku MQ_AUTHENTICATOR_LENGTH.

void setAuthenticator (const char * *name*);
Nastaví ověřovatele.

MQCHAR commitMode () const;
Vrátí režim potvrzení.

void setCommitMode (const MQCHAR *mode*);
Nastaví režim potvrzení.

ImqString logicalTerminalOverride (potlačení () const;
Vrátí kopii přepisu logického terminálu.

void setLogicalTerminalOverride (const char * *override*);
Nastaví přepis logického terminálu.

ImqString messageFormatServicesMapName () const;
Vrátí kopii názvu mapy služeb formátu zpráv.

void setMessageFormatServicesMapName (const char * *name*);
Nastaví název mapy služeb formátu zpráv.

ImqString replyToFormat () const;
Vrací kopii formátu odpovědi doplněnou koncovými mezerami o délku MQ_FORMAT_LENGTH.

void setReplyToFormat (const char * *format*);
Nastaví formát odpovědi na, doplněný koncovými mezerami na délku MQ_FORMAT_LENGTH.

MQCHAR securityScope () const;
Vrátí rozsah zabezpečení.

void setSecurityScope (const MQCHAR *scope*);
Nastaví rozsah zabezpečení.

ImqBinary transactionInstanceId () const;
Vrátí kopii ID instance transakce.

ImqBoolean setTransactionInstanceId (const ImqBinary & *id*);
Nastaví ID instance transakce. Délka dat *tokenu* musí být buď nula, nebo MQ_TRAN_INSTANCE_ID_LENGTH. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

void setTransactionInstanceId (const MQBYTE16 *id* = 0);
Nastaví ID instance transakce. *id* může být nula, což je stejné jako určení MQITII_NONE. Je-li *id* nenulový, musí adresovat MQ_TRAN_INSTANCE_ID_LENGTH bajtů binárních dat. Používáte-li předdefinované hodnoty, jako např. MQITII_NONE, budete možná muset provést přetypování, abyste zajistili shodu podpisu, například (MQBYTE *) MQITII_NONE.

MQCHAR transactionState () const;
Vrátí stav transakce.

void setTransactionState (const MQCHAR *state*);
Nastaví stav transakce.

Data objektu (chráněná)

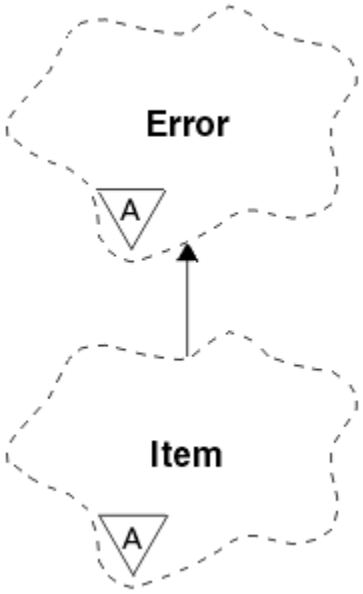
MQIIH *omqiih*
Datová struktura MQIIH.

Kódy příčin

- MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_NEKONZISTENTNÍ_FORMÁT
- MQRC_ENCODING_ERROR
- MQRC_STRUC_ID_ERROR

Třída ImqItem C++

Tato abstraktní třída představuje položku, možná jednu z několika, v rámci zprávy.



Obrázek 25. Třída ImqItem

Položky jsou zřetězeny do vyrovnávací paměti zpráv. Každá specializace je přidružena ke konkrétní datové struktuře, která začíná ID struktury.

Polymorfní metody v této abstraktní třídě umožňují kopírovat položky do zpráv a ze zpráv. Třídy ImqMessage **readItem** a **writeItem** poskytují další styl vyvolání těchto polymorfních metod, které jsou pro aplikační programy přirozenější.

- "Atributy objektů" na stránce 1827
- "Konstruktory" na stránce 1827
- "Metody třídy (veřejné)" na stránce 1827
- "Metody objektů (veřejné)" na stránce 1827
- "Kódy příčin" na stránce 1828

Atributy objektů

id struktury
Řetězec čtyř znaků na začátku datové struktury. Tento atribut je určen jen pro čtení. Zvažte tento atribut pro odvozené třídy. Není zahrnut automaticky.

Konstruktory

ImqItem( );
Výchozí konstruktor.

ImqItem(const ImqItem & ***položka*** **);**
Konstruktor kopírování.

Metody třídy (veřejné)

static ImqBoolean structureIdIs (const char * ***structure-id-to-test*****, const ImqMessage &** ***msg*** **);**
Vrátí hodnotu TRUE, pokud je **ID struktury** další položky ImqItem v příchozí *msg* stejné jako *structure-id-to-test*. Další položka je identifikována jako ta část vyrovnávací paměti zpráv, kterou momentálně adresuje ImqCache **ukazatel na data**. Tato metoda spoléhá na **id struktury** , a proto není zaručeno, že bude fungovat pro všechny odvozené třídy ImqItem .

Metody objektů (veřejné)

void operator = (const ImqItem & ***item*** **);**
Zkopíruje data instance z *položky*a nahradí existující data instance.

virtuální ImqBoolean copyOut (ImqMessage & ***msg*** **) = 0;**
Zapíše tento objekt jako další položku do vyrovnávací paměti odchozích zpráv a připojí jej k existujícím položkám. Pokud je operace zápisu úspěšná, zvýší se ImqCache **délka dat**. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

Chcete-li pracovat se specifickou podtřídou, potlačte tuto metodu.

virtuální ImqBoolean pasteIn (ImqMessage & ***msg*** **) = 0;**
Přečte tento objekt *destruktivně* z vyrovnávací paměti příchozích zpráv. Čtení je destruktivní v tom, že ImqCache **ukazatel na data** je přesunut. Obsah vyrovnávací paměti však zůstává stejný, takže data lze znovu načíst resetováním konzoly ImqCache **ukazatel na data**.

(sub) třída tohoto objektu musí být konzistentní s **ID struktury** nalezeným dále ve vyrovnávací paměti zpráv objektu *msg* .

kódování objektu *msg* by mělo být MQENC_NATIVE. Doporučuje se načítat zprávy s parametrem ImqMessage **kódování** nastaveným na hodnotu MQENC_NATIVE a s parametrem ImqGetMessageOptions **volby** včetně hodnoty MQGMO_CONVERT.

Pokud je operace čtení úspěšná, sníží se ImqCache **délka dat** . Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

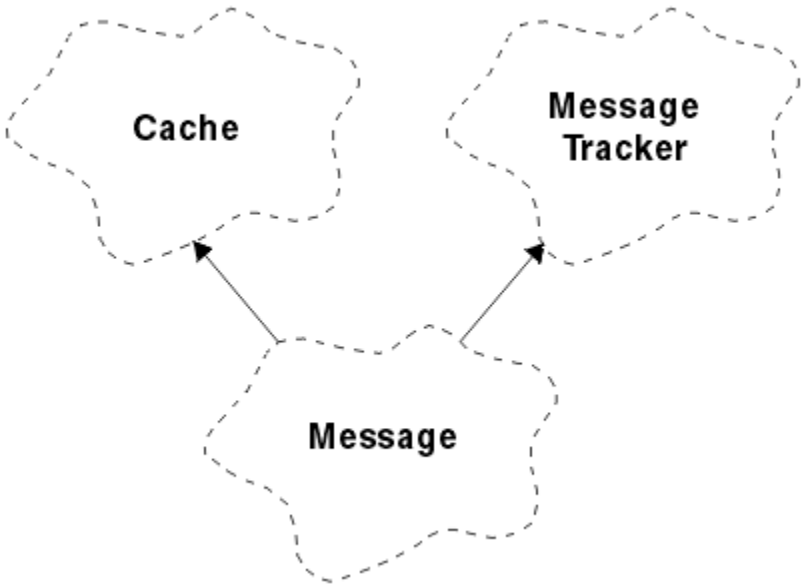
Chcete-li pracovat se specifickou podtřídou, potlačte tuto metodu.

Kódy příčin

- MQRC_ENCODING_ERROR
- MQRC_STRUC_ID_ERROR
- MQRC_NEKONZISTENTNÍ_FORMÁT
- MQRC_INNEDOSTATEČNÁ_VYROVNÁVACÍ paměť
- MQRC_INDOSTATEČNÁ_DATA

Třída ImqMessage C++

Tato třída zapouzdřuje datovou strukturu MQMD a také zpracovává konstrukci a rekonstrukci dat zpráv.



Obrázek 26. Třída ImqMessage

- "Atributy objektů" na stránce 1828
- "Konstruktory" na stránce 1832
- "Metody objektů (veřejné)" na stránce 1832
- "Metody objektů (chráněné)" na stránce 1834
- "Data objektu (chráněná)" na stránce 1834

Atributy objektů

data ID aplikace
Informace o identitě přidružené ke zprávě. Počáteční hodnota je řetězec s hodnotou Null.

Data původu aplikace
Informace o původu přidružené ke zprávě. Počáteční hodnota je řetězec s hodnotou Null.

Počet vrácení
Počet případů, kdy byla zpráva předběžně načtena a následně vrácena zpět. Počáteční hodnota je nula. Tento atribut je určen jen pro čtení.

znaková sada
ID kódované znakové sady. Počáteční hodnota je MQCCSI_Q_MGR. Jsou možné následující další hodnoty:

- MQCCSI_INHERIT

- MQCCSI_EMBEDDED

Můžete také použít kódované ID znakové sady dle vašeho výběru. Informace o tomto tématu viz “Převod kódové stránky” na stránce 936.

kódování
Kódování počítače dat zprávy. Počáteční hodnota je MQENC_NATIVE.

Vypršení
Časově závislé množství, které řídí, jak dlouho produkt IBM MQ uchová nenačtenou zprávu, než ji vyřadí. Počáteční hodnota je MQEI_UNLIMITED.

formát
Název formátu (šablony), který popisuje rozvržení dat ve vyrovnávací paměti. Názvy delší než osm znaků jsou zkráceny na osm znaků. Názvy jsou vždy vyplněny mezerami na osm znaků. Počáteční hodnota konstanty je MQFMT_NONE. Jsou možné následující další konstanty:

- 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

Můžete také použít vámi zvolený řetězec specifický pro aplikaci. Další informace naleznete v poli “Formát (MQCHAR8) pro MQMD” na stránce 449 deskriptoru zprávy (MQMD).

Příznaky zprávy
Informace o řízení segmentace. Počáteční hodnota je MQMF_SEGMENTATION_INHIBITED. Jsou možné následující další hodnoty:

- MQMF_SEGMENTATION_ALLOWED
- MQMF_MSG_IN_GROUP
- MQMF_LAST_MSG_IN_GROUP
- MQMF_SEGMENT
- MQMF_LAST_SEGMENT
- MQMF_NONE

typ zprávy
Široká kategorizace zprávy. Počáteční hodnota je MQMT_DATAGRAM. Jsou možné následující další hodnoty:

- MQMT_SYSTEM_FIRST
- MQMT_SYSTEM_LAST

- MQMT_DATAGRAM
- MQMT_REQUEST
- MQMT_REPLY
- MQMT_REPORT
- MQMT_APPL_FIRST
- MQMT_APPL_LAST

Můžete také použít hodnotu specifickou pro aplikaci dle vašeho výběru. Další informace naleznete v poli "MsgType (MQLONG) pro MQMD" na stránce 439 deskriptoru zprávy (MQMD).

posunutí
Informace o offsetu. Počáteční hodnota je nula.

Původní délka
Původní délka segmentované zprávy. Počáteční hodnota je MQOL_UNDEFINED.

trvání, perzistence
Označuje, že zpráva je důležitá a musí být vždy zálohována pomocí trvalého úložiště. Tato volba znamená pokutu za výkon. Počáteční hodnota je MQPER_PERSISTENCE_AS_Q_DEF. Jsou možné následující další hodnoty:

- MQPER_PERSISTENT
- MQPER_NOT_PERSISTENT

priorita
Relativní priorita přenosu a doručení. Zprávy se stejnou prioritou jsou obvykle doručovány ve stejném pořadí, v jakém byly dodány (i když existuje několik kritérií, která musí být splněna, aby to bylo zaručeno). Počáteční hodnota je MQPRI_PRIORITY_AS_Q_DEF.

ověření vlastnosti
Určuje, zda má při nastavení vlastnosti zprávy dojít k ověření vlastností. Počáteční hodnota je **MQCMHO_DEFAULT_VALIDATION**. Jsou možné následující další hodnoty:

- MQCMHO_VALIDATE
- MQCMHO_NO_VALIDATION

Následující metody fungují na **ověřování vlastností**:

MQLONG propertyValidation() const;
Vrátí volbu **ověření vlastnosti** .

void setPropertyValidation (const MQLONG *volba*);
Nastaví volbu **ověření vlastnosti** .

Název vkládající aplikace
Název aplikace, která vložila zprávu. Počáteční hodnota je řetězec s hodnotou Null.

Typ vkládající aplikace
Typ aplikace, která vložila zprávu. Počáteční hodnota je MQAT_NO_CONTEXT. Jsou možné následující další hodnoty:

- MQAT_AIX-operační systém
- 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
- VÝCHOZÍ
- MQAT_UNKNOWN (neznámý)
- MQAT_USER_FIRST
- MQAT_USER_LAST

Můžete také použít vámi zvolený řetězec specifický pro aplikaci. Další informace naleznete v poli "PutApplTyp (MQLONG) pro MQMD" na stránce 463 deskriptoru zprávy (MQMD).

Datum vložení
Datum, kdy byla zpráva vložena. Počáteční hodnota je řetězec s hodnotou Null.

Čas vložení
Čas, kdy byla zpráva vložena. Počáteční hodnota je řetězec s hodnotou Null.

název správce front pro odpověď
Název správce front, kterému má být odeslána odpověď. Počáteční hodnota je řetězec s hodnotou Null.

název fronty pro odpověď
Název fronty, do které má být odeslána jakákoli odpověď. Počáteční hodnota je řetězec s hodnotou Null.

sestava
Informace o zpětné vazbě přidružené ke zprávě. Počáteční hodnota je MQRO_NONE. Jsou možné následující další hodnoty:

- MQRO_EXCEPTION
- MQRO_EXCEPTION_WITH_DATA
- MQRO_EXCEPTION_WITH_FULL_DATA *
- MQRO_EXPIRATION
- MQRO_EXPIRATION_WITH_DATA
- MQRO_EXPIRATION_WITH_FULL_DATA *
- MQRO_COA
- MQRO_COA_WITH_DATA
- MQRO_COA_WITH_FULL_DATA *
- MQRO_COD
- MQRO_COD_WITH_DATA
- MQRO_COD_WITH_FULL_DATA *
- MQRO_PAN
- MQRO_NAN
- MQRO_NEW_MSG_ID
- MQRO_NEW_CORREL_ID
- MQRO_COPY_MSG_ID_TO_CORREL_ID
- MQRO_PASS_CORREL_ID
- MQRO_DEAD_LETTER_Q

- MQRO_DISCARD_MSG

kde * označuje hodnoty, které nejsou v systému IBM MQ for z/OSpodporovány.

pořadové číslo
Informace o posloupnosti identifikující zprávu ve skupině. Počáteční hodnota je jedna.

celková délka zprávy
Počet bajtů, které byly k dispozici během posledního pokusu o čtení zprávy. Toto číslo bude větší než ImqCache **délka zprávy** , pokud byla poslední zpráva oříznuta, nebo pokud nebyla přečtena poslední zpráva, protože by došlo k oříznutí. Tento atribut je určen jen pro čtení. Počáteční hodnota je nula.

Tento atribut může být užitečný v jakékoli situaci zahrnující oříznuté zprávy.

Jméno uživatele
Identita uživatele přidružená ke zprávě. Počáteční hodnota je řetězec s hodnotou Null.

Konstruktory

ImqMessage( );
Výchozí konstruktor.

ImqMessage(const ImqMessage & ***msg*** **);**
Konstruktor kopírování. Podrobnosti viz metoda **operator =** .

Metody objektů (veřejné)

void operator = (const ImqMessage & ***msg*** **);**
Zkopíruje data MQMD a zprávy z *msg*. Pokud byla uživatelem pro tento objekt dodána vyrovnávací paměť, je množství zkopírovaných dat omezeno na dostupnou velikost vyrovnávací paměti. Jinak systém zajistí, že pro zkopírovaná data bude k dispozici vyrovnávací paměť odpovídající velikosti.

ImqString applicationIdData () const ;
Vrací kopii **dat ID aplikace**.

void setApplicationIdData (const char * ***data*** **= 0);**
Nastaví **data ID aplikace**.

ImqString applicationOriginData () const ;
Vrací kopii **původních dat aplikace**.

void setApplicationOriginData (const char * ***data*** **= 0);**
Nastaví **původní data aplikace**.

MQLONG backoutCount () const ;
Vrací **počet vrácení**.

MQLONG characterSet () const ;
Vrátí **znakovou sadu**.

void setCharacterSet (const MQLONG ***ccsid*** **= MQCCSI_Q_MGR);**
Nastaví **znakovou sadu**.

MQLONG kódování () const ;
Vrací **kódování**.

void setEncoding (const MQLONG ***encoding*** **= MQENC_NATIVE);**
Nastaví **kódování**.

MQLONG vypršení platnosti () const ;
Vrátí hodnotu **expirace**.

void setExpiry (const MQLONG ***vypršení platnosti*** **);**
Nastaví **vypršení platnosti**.

ImqString formát () const ;
Vrací kopii **formátu**, včetně koncových mezer.

ImqBoolean formatIs (const char * ***format-to-test*** **) const ;**
Vrátí hodnotu TRUE, pokud je **formát** stejný jako *format-to-test*.

void setFormat (const char * *name* = 0);
Nastaví **formát**, vyplněný na osm znaků s koncovými mezerami.

MQLONG messageFlags () const ;
Vrací **příznaky zprávy**.

void setMessageFlags (const MQLONG *flags*);
Nastaví **příznaky zprávy**.

MQLONG messageType () const ;
Vrací **typ zprávy**.

void setMessageType (const MQLONG *type*);
Nastaví **typ zprávy**.

MQLONG offset () const ;
Vrátí hodnotu **offset**.

void setOffset (const MQLONG *offset*);
Nastaví **offset**.

MQLONG originalLength () const ;
Vrátí **původní délku**.

void setOriginalLength (const MQLONG *length*);
Nastaví **původní délku**.

MQLONG perzistence () const ;
Vrací **perzistenci**.

void setPersistence (const MQLONG *persistence*);
Nastaví **perzistenci**.

MQLONG priorita () const ;
Vrátí **prioritu**.

void setPriority (const MQLONG *priority*);
Nastaví **prioritu**.

ImqString putApplicationNázev () const ;
Vrací kopii **názvu vkládací aplikace**.

void setPutApplicationName (const char * *name* = 0);
Nastaví **název aplikace vložení**.

MQLONG putApplicationTyp () const ;
Vrátí **put application type**(typ aplikace vložení).

void setPutApplicationType (const MQLONG *type* = MQAT_NO_CONTEXT);
Nastaví **typ aplikace vložení**.

ImqString putDate () const ;
Vrátí kopii **data vložení**.

void setPutDate (const char * *date* = 0);
Nastaví **datum vložení**.

ImqString putTime () const ;
Vrátí kopii **času vložení**.

void setPutTime (const char * *time* = 0);
Nastaví **čas vložení**.

ImqBoolean readItem (ImqItem & *položka*);
Čte do objektu *item* z vyrovnávací paměti zpráv pomocí metody ImqItem **pasteIn** . V případě úspěchu vrátí hodnotu TRUE.

ImqString replyToQueueManagerNázev () const ;
Vrací kopii **názvu správce front pro odpověď**.

void setReplyToQueueManagerName (const char * *name* = 0);
Nastaví **název správce front pro odpověď**.

ImqString replyToQueueName () const ;
Vrací kopii **názvu fronty pro odpověď**.

void setReplyToQueueName (const char * *name* = 0);
Nastaví **název fronty pro odpověď**.

MQLONG sestava () const ;
Vrátí **sestavu**.

void setReport (const MQLONG *report*);
Nastaví **sestavu**.

MQLONG sequenceNumber () const ;
Vrátí **pořadové číslo**.

void setSequenceNumber (const MQLONG *number*);
Nastaví **pořadové číslo**.

size_t totalMessageDélka () const ;
Vrací **celkovou délku zprávy**.

ImqString userId () const ;
Vrátí kopii **ID uživatele**.

void setUserId (const char * *id* = 0);
Nastaví **ID uživatele**.

ImqBoolean writeItem (ImqItem & *položka*);
Zapisuje z objektu *item* do vyrovnávací paměti zpráv pomocí metody ImqItem **copyOut** . Zápis může mít formu vložení, nahrazení nebo připojení: závisí na třídě objektu *item* . Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

Metody objektů (chráněné)

statická void setVersionSupported (const MQLONG);
Nastaví **verzi MQMD**. Výchozí hodnota je **MQMD_VERSION_2**.

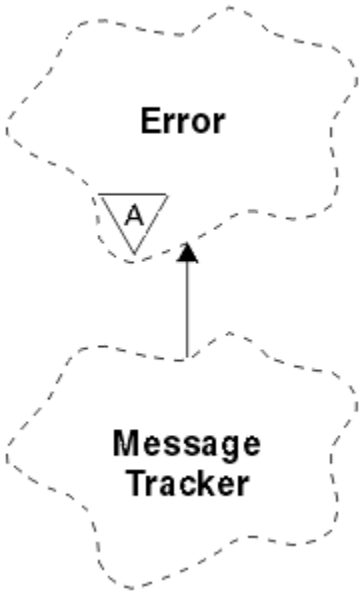
Data objektu (chráněná)

z/OS **MQMD1 *omqmd***
Datová struktura MQMD v systému z/OS.

Multi **MQMD2 *omqmd***
Datová struktura MQMD v systému Multiplatforms.

Třída ImqMessageTracker C++

Tato třída zapouzdřuje atributy objektu ImqMessage nebo ImqQueue , které lze přidružit k objektu.



Obrázek 27. Třída sledovače ImqMessage

Tato třída souvisí s voláními MQI uvedenými v seznamu “Křížový odkaz sledování ImqMessage” na stránce 1785.

Atributy objektů

Token evidence
Binární hodnota (MQBYTE32) délky MQ_ACCOUNTING_TOKEN_LENGTH. Počáteční hodnota je MQACT_NONE.

ID korelace
Binární hodnota (MQBYTE24) délky MQ_CORREL_ID_LENGTH, kterou přiřadíte ke korelování zpráv. Počáteční hodnota je MQCI_NONE. Je možná další hodnota MQCI_NEW_SESSION.

Zpětná vazba
Informace o zpětné vazbě, které mají být odeslány se zprávou. Počáteční hodnota je MQFB_NONE. Jsou možné následující další hodnoty:

- MQFB_SYSTEM_FIRST
- MQFB_SYSTEM_LAST
- MQFB_APPL_FIRST
- MQFB_APPL_LAST
- MQFB_COA
- MQFB_COD
- MQFB_EXPIRATION
- MQFB_PAN
- MQFB_NAN
- MQFB_QUIT
- MQFB_DATA_LENGTH_ZERO
- MQFB_DATA_LENGTH_NEGATIVE

- MQFB_DATA_LENGTH_TOO_BIG
- MQFB_BUFFER_OVERFLOW
- MQFB_LENGTH_OFF_BY_ONE
- MQFB_IIH_CHYBA
- MQFB_NOT_AUTHORIZED_FOR_IMS
- MQFB_IMS_ERROR
- MQFB_IMS_FIRST
- MQFB_IMS_LAST
- MQFB_CICS_APPL_ABENDED
- MQFB_CICS_APPL_NOT_STARTED
- MQFB_CICS_BRIDGE_FAILURE
- MQFB_CICS_CCSID_ERROR
- MQFB_CICS_CIH_ERROR
- MQFB_CICS_COMMAREA_ERROR
- MQFB_CICS_CORREL_ID_ERROR
- MQFB_CICS_DLQ_ERROR
- MQFB_CICS_ENCODING_ERROR
- MQFB_CICS_INTERNAL_ERROR
- MQFB_CICS_NOT_AUTHORIZED
- MQFB_CICS_UOW_BACKED_OUT
- MQFB_CICS_UOW_ERROR

Můžete také použít vámi zvolený řetězec specifický pro aplikaci. Další informace naleznete v poli "Zpětná vazba (MQLONG) pro MQMD" na stránce 444 deskriptoru zprávy (MQMD).

ID skupiny
Binární hodnota (MQBYTE24) délky MQ_GROUP_ID_LENGTH je v rámci fronty jedinečná. Počáteční hodnota je MQGI_NONE.

ID zprávy
Binární hodnota (MQBYTE24) délky MQ_MSG_ID_LENGTH jedinečná v rámci fronty. Počáteční hodnota je MQMI_NONE.

Konstruktory

ImqMessageTracker ();
Výchozí konstruktor.

ImqMessageTracker (const ImqMessageTracker & *tracker*);
Konstruktor kopírování. Podrobnosti viz metoda **operator =** .

Metody objektů (veřejné)

void operator = (const ImqMessageTracker & *tracker*);
Zkopíruje data instance ze *sledovače*a nahradí existující data instance.

ImqBinary accountingToken () const ;
Vrátí kopii **tokenu evidence**.

ImqBoolean setAccounting (const ImqBinary & *token*);
Nastaví **token evidence**. **Délka dat** *tokenu* musí být buď nula, nebo MQ_ACCOUNTING_TOKEN_LENGTH. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

void setAccountingToken (const MQBYTE32 *token* = 0);
Nastaví **token evidence**. Hodnota *token* může být nula, což je stejné jako hodnota MQACT_NONE. Je-li *token* nenulový, musí adresovat MQ_ACCOUNTING_TOKEN_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, jako např. MQACT_NONE, může být nutné provést přetypování, aby byla zajištěna shoda podpisu; například (MQBYTE *) MQACT_NONE.

ImqBinary correlationId () const ;
Vrací kopii **ID korelace**.

ImqBoolean setCorrelationId (const ImqBinary & *token*);
Nastaví **ID korelace**. **Délka dat** *tokenu* musí být buď nula, nebo MQ_CORREL_ID_LENGTH. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

void setCorrelationId (const MQBYTE24 *id* = 0);
Nastaví **ID korelace**. *id* může být nula, což je stejné jako uvedení MQCI_NONE. Pokud je *id* nenulové, musí adresovat MQ_CORREL_ID_LENGTH bajtů binárních dat. Používáte-li předdefinované hodnoty, jako např. MQCI_NONE, možná budete muset provést přetypování, abyste zajistili shodu podpisu; například (MQBYTE *) MQCI_NONE.

MQLONG zpětná vazba () const ;
Vrátí **zpětnou vazbu**.

void setFeedback (const MQLONG *feedback*);
Nastaví **zpětnou vazbu**.

ImqBinary groupId () const ;
Vrací kopii **ID skupiny**.

ImqBoolean setGroupId (const ImqBinary & *token*);
Nastaví **ID skupiny**. **Délka dat** *tokenu* musí být buď nula, nebo MQ_GROUP_ID_LENGTH. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

void setGroupId (const MQBYTE24 *id* = 0);
Nastaví **ID skupiny**. *id* může být nula, což je stejné jako určení MQGI_NONE. Pokud je *id* nenulové, musí adresovat MQ_GROUP_ID_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, například MQGI_NONE, může být nutné provést přetypování, aby byla zajištěna shoda s podpisem, například (MQBYTE *) MQGI_NONE.

ImqBinary messageId () const ;
Vrací kopii **ID zprávy**.

ImqBoolean setMessageId (const ImqBinary & *token*);
Nastaví **ID zprávy**. **Délka dat** *tokenu* musí být buď nula, nebo MQ_MSG_ID_LENGTH. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

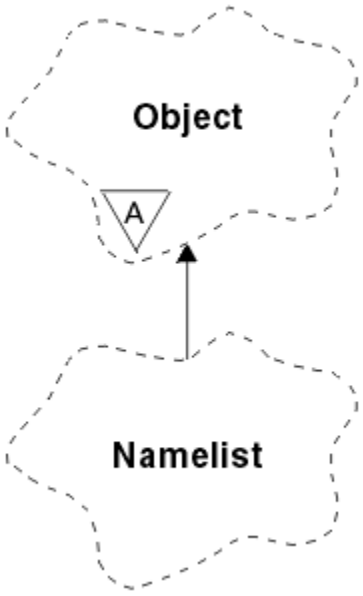
void setMessageId (const MQBYTE24 *id* = 0);
Nastaví **ID zprávy**. *id* může být nula, což je stejné jako určení MQMI_NONE. Pokud je *id* nenulové, musí adresovat MQ_MSG_ID_LENGTH bajtů binárních dat. Používáte-li předdefinované hodnoty, jako např. MQMI_NONE, možná budete muset provést přetypování, abyste zajistili shodu podpisu, například (MQBYTE *) MQMI_NONE.

Kódy příčin

- MQRC_BINARY_DATA_LENGTH_ERROR

Třída ImqNamelist C++

Tato třída zapouzdřuje seznam názvů.



Obrázek 28. Třída ImqNamelist

Tato třída souvisí s voláními MQI uvedenými v seznamu “Křížový odkaz ImqNamelist” na stránce 1786.

Atributy objektů

Počet názvů
Počet názvů objektů v **názvech seznamů názvů**. Tento atribut je určen jen pro čtení.

názvy seznamů názvů
Názvy objektů, jejichž počet je označen **počtem názvů**. Tento atribut je určen jen pro čtení.

Konstruktory

ImqNamelist( );
Výchozí konstruktor.

ImqNamelist(const ImqNamelist & *list*);
Konstruktor kopírování. ImqObject **stav otevření** je false.

ImqNamelist(const char * *name*);
Nastaví název ImqObject na **name**.

Metody objektů (veřejné)

void operator = (const ImqNamelist & *seznam*);
Zkopíruje data instance ze seznamu *list*a nahradí existující data instance. ImqObject **stav otevření** je false.

ImqBoolean nameCount(MQLONG & *count*);
Poskytuje kopii **počtu názvů**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG nameCount ();
Vrací **počet názvů** bez jakéhokoli označení možných chyb.

ImqBoolean namelistName (const MQLONG *index*, ImqString & *název*);
Poskytuje kopii jednoho z **názvů seznamů názvů** podle indexu založeného na nule. V případě úspěchu vrátí hodnotu TRUE.

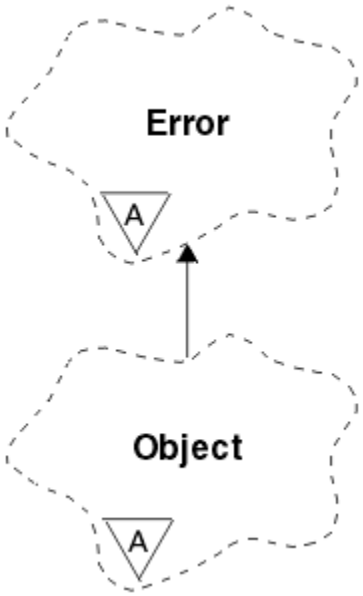
ImqString namelistName (const MQLONG *index*);
Vrací jeden z **názvů seznamů názvů** indexem založeným na nule bez jakéhokoli označení možných chyb.

Kódy příčin

- MQRC_INDEX_ERROR
- MQRC_INDEX_NOT_PŘÍTOMNÝ_XX_ENCODE_CASE_ONE mqrc_index_dárek

Třída ImqObject C++

Tato třída je abstraktní. Je-li objekt této třídy zničen, je automaticky zavřen a jeho připojení ke správci ImqQueueManager bylo přerušeno.



Obrázek 29. Třída ImqObject

Tato třída souvisí s voláními MQI uvedenými v seznamu "Křížový odkaz ImqObject" na stránce 1786.

Atributy třídy

chování
Řídí chování implicitního otevření.

IMQ_IMPL_OPEN (8L)
Implicitní otevření je povoleno. Toto nastavení je výchozí.

Atributy objektů

Datum změny
Datum změny. Tento atribut je určen jen pro čtení.

Čas změny
Doba změny. Tento atribut je určen jen pro čtení.

Jméno alternativního uživatele
Alternativní ID uživatele, nejvýše MQ_USER_ID_LENGTH znaků. Počáteční hodnota je řetězec s hodnotou Null.

alternativní ID zabezpečení
Alternativní ID zabezpečení. Binární hodnota (MQBYTE40) délky MQ_SECURITY_ID_LENGTH. Počáteční hodnota je MQSID_NONE.

volby zavření
Volby, které se použijí při zavření objektu. Počáteční hodnota je MQCO_NONE. Tento atribut je ignorován během implicitních operací opětovného otevření, kde se vždy používá hodnota MQCO_NONE.

odkaz na připojení
Odkaz na objekt ImqQueueManager, který poskytuje požadované připojení k (lokálnímu) správci front. V případě objektu ImqQueueManager se jedná o samotný objekt. Počáteční hodnota je nula.

Poznámka: Nepleťte si to s názvem správce front, který identifikuje správce front (pravděpodobně vzdáleného) pro pojmenovanou frontu.

description
Popisný název (až 64 znaků) správce front, fronty, seznamu názvů nebo procesu. Tento atribut je určen jen pro čtení.

Název
Název (až 48 znaků) správce front, fronty, seznamu názvů nebo procesu. Počáteční hodnota je řetězec s hodnotou Null. Název modelové fronty se po **open** změní na název výsledné dynamické fronty.

Poznámka: Správce ImqQueueManager může mít název s hodnotou Null představující výchozího správce front. Název se po úspěšném otevření změní na skutečného správce front. Seznam ImqDistributionje dynamický a musí mít název s hodnotou null.

další spravovaný objekt
Toto je další objekt této třídy, bez konkrétního pořadí, který má stejný odkaz na připojení jako tento objekt. Počáteční hodnota je nula.

Volby otevření
Volby, které se použijí při otevření objektu. Počáteční hodnota je MQOO_INQUIRE. Existují dva způsoby, jak nastavit příslušné hodnoty:

1. Nenastavujte volby otevření a nepoužívejte metodu otevření. Produkt IBM MQ automaticky upraví volby otevření a podle potřeby automaticky otevře, znovu otevře a zavře objekty. To může vést k nepotřebným operacím opětovného otevření, protože produkt IBM MQ používá metodu openFor , a to přidává volby otevření pouze přírůstkově.
2. Před použitím metod, jejichž výsledkem je volání MQI, nastavte volby otevření (viz “Křížový odkaz C++ a MQI” na stránce 1779). Tím zajistíte, že se nevyskytují nadbytečné operace opětovného otevření. Nastavte volby otevření explicitně, pokud se pravděpodobně vyskytnou potenciální problémy s opětovným otevřením (viz Znovu otevřít).

   Pokud použijete metodu otevření, musíte se nejprve ujistit, že volby otevření jsou vhodné. Použití otevřené metody však není povinné; IBM MQ stále vykazuje stejné chování jako v případě 1, ale za této situace je chování efektivní.

Nula není platná hodnota; nastavte odpovídající hodnotu před pokusem o otevření objektu. To lze provést buď pomocí **setOpenOptions** (*lOpenVolby*) následované **open** (), nebo **openFor** (*lRequiredOpenOption*).

Poznámka:

1. Příkaz MQOO_OUTPUT je během metody **open** nahrazen parametrem MQOO_INQUIRE pro distribuční seznam, protože parametr MQOO_OUTPUT je v tuto chvíli jediným platným parametrem **open option** . Doporučuje se však vždy explicitně nastavit MQOO_OUTPUT v aplikačních programech, které používají metodu **open** .
2. Zadejte MQOO_RESOLVE_NAMES, chcete-li použít atributy **resolved queue manager name** a **resolved queue name** třídy.

stav otevření
Zda je objekt otevřený (TRUE) nebo zavřený (FALSE). Počáteční hodnota je FALSE. Tento atribut je určen jen pro čtení.

předchozí spravovaný objekt
Předchozí objekt této třídy, bez konkrétního pořadí, se stejným odkazem na připojení jako tento objekt. Počáteční hodnota je nula.

identifikátor-správce-front
Identifikátor správce front. Tento atribut je určen jen pro čtení.

Konstruktory

ImqObject( );
Výchozí konstruktor.

ImqObject(const ImqObject & ***object*** **);**
Konstruktor kopírování. Stav otevření bude FALSE.

Metody třídy (veřejné)

statické chování MQLONG ();
Vrátí chování.

void setBehavior(const MQLONG ***behavior*** **= 0);**
Nastaví chování.

Metody objektů (veřejné)

void operator = (const ImqObject & ***object*** **);**
V případě potřeby provede zavření a zkopíruje data instance z objektu *object*. Stav otevření bude FALSE.

ImqBoolean alterationDate(ImqString & ***datum*** **);**
Poskytuje kopii data změny. V případě úspěchu vrátí hodnotu TRUE.

ImqString alterationDate( );
Vrátí datum změny bez jakékoli indikace možných chyb.

ImqBoolean alterationTime(ImqString & ***čas*** **);**
Poskytuje kopii doby změny. V případě úspěchu vrátí hodnotu TRUE.

ImqString alterationTime( );
Vrátí čas změny bez jakékoli indikace možných chyb.

ImqString alternateUserId () const;
Vrátí kopii alternativního ID uživatele.

ImqBoolean setAlternateUserId (const char * ***id*** **);**
Nastaví alternativní ID uživatele. Alternativní ID uživatele lze nastavit pouze v případě, že stav otevření je FALSE. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBinary alternateSecurityId () const;
Vrátí kopii alternativního ID zabezpečení.

ImqBoolean setAlternateSecurityId(const ImqBinary & *token*);
Nastaví alternativní ID zabezpečení. Alternativní ID zabezpečení lze nastavit pouze v případě, že stav otevření je FALSE. Délka dat *token* musí být buď nula, nebo MQ_SECURITY_ID_LENGTH. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean setAlternateSecurityId(const MQBYTE* *token* = 0);
Nastaví alternativní ID zabezpečení. *token* může mít hodnotu nula, což je stejné jako zadání parametru MQSID_NONE. Je-li *token* nenulový, musí adresovat bajty MQ_SECURITY_ID_LENGTH binárních dat. Při použití předdefinovaných hodnot, jako např. MQSID_NONE, může být nutné provést přetypování, aby byla zajištěna shoda podpisu; například (MQBYTE *) MQSID_NONE.

Alternativní ID zabezpečení lze nastavit pouze v případě, že stav otevření je TRUE. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean setAlternateSecurityId(const unsigned char * *id* = 0);
Nastaví alternativní ID zabezpečení.

ImqBoolean close ();
Nastaví stav otevření na FALSE. V případě úspěchu vrátí hodnotu TRUE.

MQLONG closeOptions () const;
Vrátí volby zavření.

void setCloseVolby (const MQLONG *options*);
Nastaví volby zavření.

ImqQueueManager * connectionReference () const;
Vrátí odkaz na připojení.

void setConnectionReference (ImqQueueManager & *manager*);
Nastaví odkaz na připojení.

void setConnectionReference (ImqQueueManager * *manager* = 0);
Nastaví odkaz na připojení.

virtuální popis ImqBoolean (ImqString & *popis*) = 0;
Poskytuje kopii popisu. V případě úspěchu vrátí hodnotu TRUE.

ImqString description ();
Vrátí kopii popisu bez označení možných chyb.

virtuální ImqBoolean název (ImqString & *název*);
Poskytuje kopii názvu. V případě úspěchu vrátí hodnotu TRUE.

ImqString name ();
Vrací kopii názvu bez jakéhokoli označení možných chyb.

ImqBoolean setName (const char * *name* = 0);
Nastaví název. Název lze nastavit pouze v případě, že stav otevření je FALSE, a v případě správce ImqQueueManager v případě, že stav připojení je FALSE. V případě úspěchu vrátí hodnotu TRUE.

ImqObject * nextManagedObject () const;
Vrátí další spravovaný objekt.

ImqBoolean open ();
Změní stav otevření na TRUE otevřením objektu podle potřeby, s použitím mimo jiné atributů voleb otevření a názvu. Tato metoda používá referenční informace o připojení a v případě potřeby metodu připojení správce ImqQueueManager, aby se zajistilo, že stav připojení správce ImqQueueje TRUE. Vrátí stav otevření.

ImqBoolean openFor (const MQLONG *required-options* = 0);
Pokusí se zajistit, aby byl objekt otevřen s otevřenými volbami nebo s otevřenými volbami, které zaručují chování odvozené z hodnoty parametru *required-option*s .

Je-li *required-options* nula, vstup je povinný a všechny volby vstupu postačují. Pokud tedy otevřené volby již obsahují jednu z následujících možností:

- MQOO_INPUT_AS_Q_DEF
- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE

Volby otevření jsou již uspokojivé a nejsou změněny. Pokud volby otevření ještě neobsahují žádnou z těchto voleb, je v otevřených volbách nastavena volba MQOO_INPUT_AS_Q_DEF.

Je-li *required-options* nenulová, požadované volby se přidají do otevřených voleb; Je-li *required-options* některou z těchto voleb, ostatní se vynulují.

Pokud se některá z voleb otevření změní a objekt je již otevřený, objekt se dočasně zavře a znovu otevře, aby se upravily možnosti otevření.

V případě úspěchu vrátí hodnotu TRUE. Úspěch označuje, že objekt je otevřený s odpovídajícími volbami.

MQLONG openOptions () const;
Vrátí otevřené volby.

ImqBoolean setOpenVolby (const MQLONG *options*);
Nastaví volby otevření. Volby otevření lze nastavit pouze v případě, že stav otevření je FALSE. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean openStatus () const;
Vrátí stav otevření.

ImqObject * previousManagedObject () const;
Vrátí předchozí spravovaný objekt.

ImqBoolean queueManagerIdentifikátor (ImqString & *id*);
Poskytuje kopii identifikátoru správce front. V případě úspěchu vrátí hodnotu TRUE.

ImqString queueManagerIdentifier ();
Vrací identifikátor správce front bez jakékoli indikace možných chyb.

Metody objektů (chráněné)

virtuální ImqBoolean closeTemporarily ();
Před opětovným otevřením objekt bezpečně zavře. V případě úspěchu vrátí hodnotu TRUE. Tato metoda předpokládá, že stav otevření je TRUE.

MQHCONN connectionHandle () const;
Vrátí hodnotu MQHCONN přidruženou k odkazu na připojení. Tato hodnota je nula, pokud neexistuje žádný odkaz na připojení nebo pokud není správce připojen.

ImqBoolean inquire (const MQLONG *int-attr*, MQLONG & *hodnota*);
Vrací celočíselnou hodnotu, jejíž index je hodnotou MQIA_ *. V případě chyby je hodnota nastavena na MQIAV_UNDEFINED.

ImqBoolean inquire (const MQLONG *char-attr*, char * & *buffer*, const size_t *length*);
Vrací znakový řetězec, jehož indexem je hodnota MQCA_ *.

Poznámka: Obě tyto metody vrací pouze jednu hodnotu atributu. Pokud je *snímek* požadován s více než jednou hodnotou, kde jsou hodnoty v okamžiku vzájemně konzistentní, produkt IBM MQ C++ tento prostředek neposkytuje a musíte použít volání MQINQ s příslušnými parametry.

virtuální void openInformationDisperse ();
Okamžitě po volání MQOPEN rozptýlí informace ze sekce proměnných datové struktury MQOD.

virtual ImqBoolean openInformationPřipravit ();
Připraví informace pro sekci proměnných datové struktury MQOD bezprostředně před voláním MQOPEN a v případě úspěchu vrátí hodnotu TRUE.

ImqBoolean set (const MQLONG *int-attr*, const MQLONG *hodnota*);
Nastaví celočíselný atribut IBM MQ .

ImqBoolean set (const MQLONG *char-attr*, const char * *buffer*, const size_t *required-length*);
Nastaví znakový atribut IBM MQ .

void setNextManagedObject (const ImqObject * *objekt* = 0);
Nastaví další spravovaný objekt.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nepřeruší seznam spravovaných objektů.

void setPreviousManagedObject (const ImqObject * *objekt* = 0);
Nastaví předchozí spravovaný objekt.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nepřeruší seznam spravovaných objektů.

Data objektu (chráněná)

MQHOBJ *ohobj*
Popisovač objektu IBM MQ (platný pouze v případě, že stav otevření je TRUE).

MQOD *omqod*
Vestavěná datová struktura MQOD. Velikost úložiště přiděleného pro tuto datovou strukturu je ta, která je nezbytná pro MQOD verze 2. Zkontrolujte číslo verze *(omqod.Version)* a přistupte k dalším polím následujícím způsobem:

MQOD_VERSION_1
Ke všem ostatním polím v poli *omqod* lze přistupovat.

MQOD_VERSION_2
Ke všem ostatním polím v poli *omqod* lze přistupovat.

MQOD_VERSION_3
omqod.pmqod je ukazatel na dynamicky přidělenou, větší MQOD. K žádným dalším polím v poli *omqod* nelze přistupovat. Ke všem polím adresovaným příkazem *omqod.pmqod* lze přistupovat.

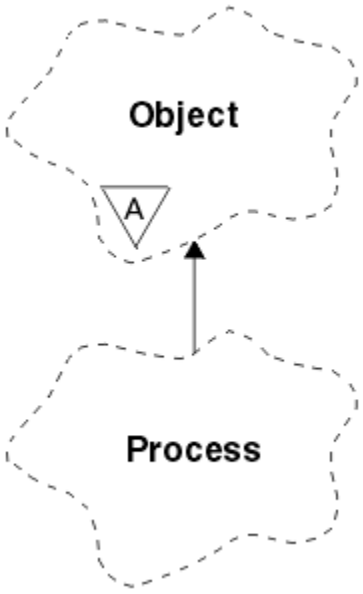
Poznámka: *omqod.pmqod.Version* může být menší než *omqod.Version*, což znamená, že IBM MQ MQI client má více funkcí než server IBM MQ .

Kódy příčin

- MQRC_ATTRIBUTE_LOCKED
- MQRC_INCONSISTENT_OBJECT_STATE
- MQRC_NO_CONNECTION_REFERENCE
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_REOPEN_SAVED_CONTEXT_ERR
- (kódy příčiny z MQCLOSE)
- (kódy příčiny z MQCONN)
- (kódy příčiny z MQINQ)
- (kódy příčiny z MQOPEN)
- (kódy příčiny z MQSET)

Třída ImqProcess C++

Tato třída zapouzdřuje aplikační proces (objekt IBM MQ typu MQOT_PROCESS), který může spustit monitor spouštěčů.



Obrázek 30. Třída ImqProcess

- "Atributy objektů" na stránce 1845
- "Konstruktory" na stránce 1845
- "Metody objektů (veřejné)" na stránce 1845

Atributy objektů

ID aplikace
Identita aplikačního procesu. Tento atribut je určen jen pro čtení.

Typ aplikace
Typ procesu aplikace. Tento atribut je určen jen pro čtení.

Data prostředí
Informace o prostředí pro proces. Tento atribut je určen jen pro čtení.

Data uživatele
Uživatelská data pro proces. Tento atribut je určen jen pro čtení.

Konstruktory

ImqProcess( );
Výchozí konstruktor.

ImqProcess(const ImqProcess & *proces*);
Konstruktor kopírování. ImqObject **stav otevření** je FALSE.

ImqProcess(const char * *name*);
Nastaví ImqObject **název**.

Metody objektů (veřejné)

void operator = (const ImqProcess & *process*);
V případě potřeby provede zavření a poté zkopíruje data instance z *procesu*. ImqObject **stav otevření** bude FALSE.

ImqBoolean applicationId (ImqString & *id*);
Poskytuje kopii **ID aplikace**. V případě úspěchu vrátí hodnotu TRUE.

ImqString applicationId ( );
Vrací **ID aplikace** bez jakékoli indikace možných chyb.

ImqBoolean applicationType (MQLONG & *typ*);
Poskytuje kopii **typu aplikace**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG applicationType ();
Vrací **typ aplikace** bez označení možných chyb.

ImqBoolean environmentData (ImqString & *data*);
Poskytuje kopii **dat prostředí**. V případě úspěchu vrátí hodnotu TRUE.

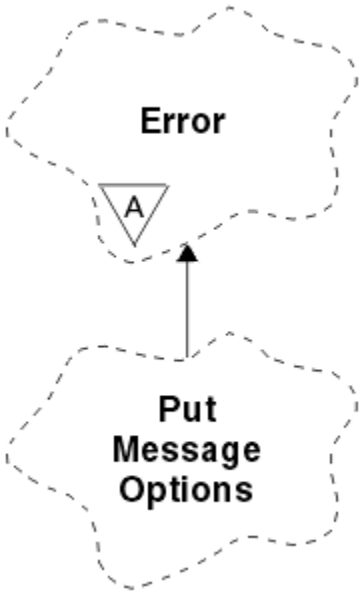
ImqString environmentData ( );
Vrací **data prostředí** bez jakéhokoli označení možných chyb.

ImqBoolean userData (ImqString & *data*);
Poskytuje kopii **uživatelských dat**. V případě úspěchu vrátí hodnotu TRUE.

ImqString userData ( );
Vrací **uživatelská data** bez jakéhokoli označení možných chyb.

ImqPutMessageOptions třída C++

Tato třída zapouzdřuje datovou strukturu MQPMO.



Obrázek 31. Třída ImqPutMessageOptions

- "Atributy objektů" na stránce 1846
- "Konstruktory" na stránce 1847
- "Metody objektů (veřejné)" na stránce 1847
- "Data objektu (chráněná)" na stránce 1848
- "Kódy příčin" na stránce 1848

Atributy objektů

odkaz na kontext
ImqQueue , která poskytuje kontext pro zprávy. Na začátku neexistuje žádný odkaz.

volby
Volby vložení zprávy. Počáteční hodnota je MQPMO_NONE. Jsou možné následující další hodnoty:

- MQPMO_SYNCPOINT
- MQPMO_NO_SYNCPOINT
- MQPMO_NEW_MSG_ID

- MQPMO_NEW_CORREL_ID
- MQPMO_LOGICAL_ORDER
- MQPMO_NO_CONTEXT
- MQPMO_DEFAULT_CONTEXT
- MQPMO_PASS_IDENTITY_CONTEXT
- MQPMO_PASS_ALL_CONTEXT
- MQPMO_SET_IDENTITY_CONTEXT
- MQPMO_SET_ALL_CONTEXT
- Oprávnění MQPMO_ALTERNATE_USER_AUTHORITY
- MQPMO_FAIL_IF_QUIESCING

pole záznamu
Příznaky, které řídí zahrnutí záznamů vložených zpráv při vložení zprávy. Počáteční hodnota je MQPMRF_NONE. Jsou možné následující další hodnoty:

- MQPMRF_MSG_ID
- MQPMRF_CORREL_ID
- MQPMRF_GROUP_ID
- MQPMRF_ZPĚTNÁ vazba
- MQPMRF_ACCOUNTING_TOKEN

ImqMessageAtributy sledovače jsou převzaty z objektu pro libovolné uvedené pole. Atributy sledování ImqMessagejsou převzaty z objektu ImqMessage pro libovolné pole, které není uvedeno.

název vyřešeného správce front
Název cílového správce front určeného během operace vložení. Počáteční hodnota je null. Tento atribut je určen jen pro čtení.

vyřešený název fronty
Název cílové fronty určený během operace vložení. Počáteční hodnota je null. Tento atribut je určen jen pro čtení.

účast synchronizačního bodu
TRUE když jsou zprávy umístěny pod řízení synchronizačního bodu.

Konstruktory

ImqPutMessageOptions( );
Výchozí konstruktor.

ImqPutMessageOptions(const ImqPutMessageOptions & *pmo*);
Konstruktor kopírování.

Metody objektů (veřejné)

void operator = (const ImqPutMessageOptions & *pmo*);
Zkopíruje data instance z *pmo*a nahradí existující data instance.

ImqQueue * contextReference () const;
Vrací odkaz na kontext.

void setContextReference (const ImqQueue & *queue*);
Nastaví odkaz na kontext.

void setContextReference (const ImqQueue * *queue* = 0);
Nastaví odkaz na kontext.

Volby MQLONG () const;
Vrátí volby.

void setOptions (const MQLONG *options*);
Nastaví volby včetně hodnoty účasti synchronizačního bodu.

MQLONG recordFields () const;
Vrátí pole záznamu.

void setRecordFields (const MQLONG *fields*);
Nastaví pole záznamu.

ImqString resolvedQueueManagerName () const;
Vrací kopii rozlišeného názvu správce front.

ImqString resolvedQueueName () const;
Vrátí kopii rozlišeného názvu fronty.

ImqBoolean syncPointParticipace () const;
Vrací hodnotu účasti synchronizačního bodu, která je TRUE, pokud volby zahrnují MQPMO_SYNCPOINT.

void setSyncPointParticipation (const ImqBoolean *sync*);
Nastaví hodnotu účasti synchronizačního bodu. Má-li parametr *sync* hodnotu TRUE, jsou volby změněny tak, aby zahrnovaly MQPMO_SYNCPOINT a vylučovaly MQPMO_NO_SYNCPOINT. Má-li parametr *sync* hodnotu FALSE, jsou volby změněny tak, aby zahrnovaly MQPMO_NO_SYNCPOINT a vylučovaly MQPMO_SYNCPOINT.

Data objektu (chráněná)

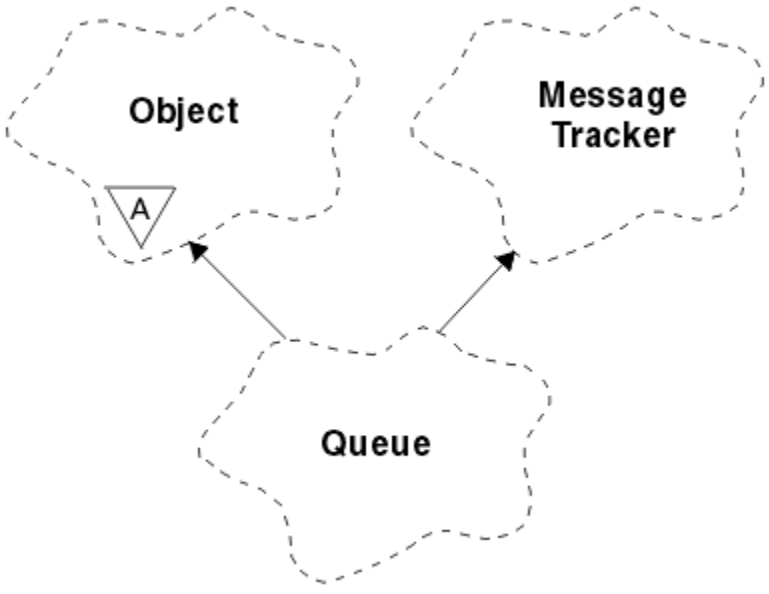
MQPMO *omqpmo*
Datová struktura MQPMO.

Kódy příčin

- MQRC_STORAGE_NOT_AVAILABLE

Třída ImqQueue C++

Tato třída zapouzdřuje frontu zpráv (objekt IBM MQ typu MQOT_Q).



Obrázek 32. třída ImqQueue

Tato třída souvisí s voláními MQI uvedenými v seznamu Tabulka 862 na stránce 1787.

- “Atributy objektů” na stránce 1849

- “Konstruktory” na stránce 1852
- “Metody objektů (veřejné)” na stránce 1852
- “Metody objektů (chráněné)” na stránce 1858
- “Kódy příčin” na stránce 1858

Atributy objektů

Zpětné jméno přefrontování
Nadměrný počet vrácený název fronty. Tento atribut je určen jen pro čtení.

Práh vrácení
Prahová hodnota vyřazených zpráv. Tento atribut je určen jen pro čtení.

název základní fronty
Název fronty, na kterou se alias interpretuje. Tento atribut je určen jen pro čtení.

název klastru
Název klastru. Tento atribut je určen jen pro čtení.

Název seznamu názvů klastru
Název seznamu názvů klastru. Tento atribut je určen jen pro čtení.

Rozsah vytíľení klastru
Úroveň vytíľení klastru. Tento atribut je určen jen pro čtení.

Priorita vytíľení klastru
Priorita vytíľení klastru. Tento atribut je určen jen pro čtení.

Pracovní zátěž klastru - použitá fronta
Hodnota fronty využití pracovní zátěže klastru. Tento atribut je určen jen pro čtení.

Datum vytvoření
Data vytvoření fronty. Tento atribut je určen jen pro čtení.

Čas vytvoření
Čas vytvoření fronty. Tento atribut je určen jen pro čtení.

Aktuální délka
Počet zpráv ve frontě. Tento atribut je určen jen pro čtení.

výchozí vazba
Výchozí vazba. Tento atribut je určen jen pro čtení.

Výchozí volba otevření pro vstup
Výchozí volba open-for-input. Tento atribut je určen jen pro čtení.

Výchozí trvání
Výchozí perzistence zpráv. Tento atribut je určen jen pro čtení.

Výchozí priorita
Výchozí priorita zprávy. Tento atribut je určen jen pro čtení.

Typ definice
Typ definice fronty. Tento atribut je určen jen pro čtení.

událost s vysokou hloubkou
Řídicí atribut pro události vysoké hloubky fronty. Tento atribut je určen jen pro čtení.

horní mez hloubky
Horní limit pro hloubku fronty. Tento atribut je určen jen pro čtení.

hloubka nízká událost
Řídicí atribut pro události nízké hloubky fronty. Tento atribut je určen jen pro čtení.

dolní mez hloubky
Dolní limit pro hloubku fronty. Tento atribut je určen jen pro čtení.

událost maxima hloubky
Řídicí atribut pro události maxima hloubky fronty. Tento atribut je určen jen pro čtení.

odkaz na distribuční seznam
Volitelný odkaz na seznam ImqDistribution, který lze použít k distribuci zpráv do více než jedné fronty, včetně této. Počáteční hodnota je null.

Poznámka: Při otevření objektu ImqQueue se automaticky zavře každý otevřený objekt ImqDistributionList, na který odkazuje.

distribuční seznamy
Schopnost přenosové fronty podporovat distribuční seznamy. Tento atribut je určen jen pro čtení.

název dynamické fronty
Název dynamické fronty. Počáteční hodnota je AMQ.* pro všechny platformy AIX, Linux, and Windows .

Uloľení počtu vrácení
Zda zatvrdit počet vrácení. Tento atribut je určen jen pro čtení.

Typ indexu
Typ indexu. Tento atribut je určen jen pro čtení.

inhibovat získání
Zda jsou povoleny operace získání. Počáteční hodnota závisí na definici fronty. Tento atribut je platný pouze pro alias nebo lokální frontu.

blokování vložení
Zda jsou povoleny operace vložení. Počáteční hodnota závisí na definici fronty.

Název inicializační fronty
Název inicializační fronty. Tento atribut je určen jen pro čtení.

Maximální hloubka
Maximální počet zpráv povolených ve frontě. Tento atribut je určen jen pro čtení.

Maximální délka zprávy
Maximální délka pro libovolnou zprávu v této frontě, která může být menší než maximální délka pro libovolnou frontu spravovanou přidruženým správcem front. Tento atribut je určen jen pro čtení.

Pořadí doručení zpráv
Zda je priorita zprávy relevantní. Tento atribut je určen jen pro čtení.

další distribuovaná fronta
Další objekt této třídy, v žádném konkrétním pořadí, který má stejný **odkaz na distribuční seznam** jako tento objekt. Počáteční hodnota je nula.

Pokud je objekt v řetězci odstraněn, předchozí objekt a další objekt jsou aktualizovány tak, aby jejich distribuované odkazy na frontu již neukazovaly na odstraněný objekt.

dočasná třída zprávy
Úroveň spolehlivosti pro dočasné zprávy vkládané do této fronty. Tento atribut je určen jen pro čtení.

Otevření pro vstup - počet
Počet objektů ImqQueue , které jsou otevřeny pro vstup. Tento atribut je určen jen pro čtení.

Otevření pro výstup - počet
Počet objektů ImqQueue , které jsou otevřeny pro výstup. Tento atribut je určen jen pro čtení.

předchozí distribuovaná fronta
Předchozí objekt této třídy, bez konkrétního pořadí, se stejným **odkazem na distribuční seznam** jako tento objekt. Počáteční hodnota je nula.

Pokud je objekt v řetězci odstraněn, předchozí objekt a další objekt jsou aktualizovány tak, aby jejich distribuované odkazy na frontu již neukazovaly na odstraněný objekt.

Název procesu
Název definice procesu. Tento atribut je určen jen pro čtení.

Účtování fronty
Úroveň informací o účtování pro fronty. Tento atribut je určen jen pro čtení.

název-správce-front
Název správce front (případně vzdáleného), ve kterém je fronta umístěna. Nezaměňujte zde uvedeného správce front s produktem ImqObject **odkaz na připojení**, který odkazuje na (lokálního) správce front poskytujícího připojení. Počáteční hodnota je null.

Monitorování fronty
Úroveň shromažďování dat monitorování pro frontu. Tento atribut je určen jen pro čtení.

Statistiky fronty
Úroveň statistických dat pro frontu. Tento atribut je určen jen pro čtení.

Typ fronty
Typ fronty. Tento atribut je určen jen pro čtení.

Název vzdáleného správce front
Název vzdáleného správce front. Tento atribut je určen jen pro čtení.

Název vzdálené fronty
Název vzdálené fronty, jak je znám ve vzdáleném správci front. Tento atribut je určen jen pro čtení.

název vyřešeného správce front
Vyřešený název správce front. Tento atribut je určen jen pro čtení.

vyřešený název fronty
Vyřešený název fronty. Tento atribut je určen jen pro čtení.

Interval uchování
Interval uchování fronty. Tento atribut je určen jen pro čtení.

rozsah
Rozsah definice fronty. Tento atribut je určen jen pro čtení.

interval služeb
Servisní interval. Tento atribut je určen jen pro čtení.

událost intervalu služeb
Řídicí atribut pro události servisního intervalu. Tento atribut je určen jen pro čtení.

Možnost sdílení
Zda fronta může být sdílená. Tento atribut je určen jen pro čtení.

paměťová třída
Paměťová třída. Tento atribut je určen jen pro čtení.

Jméno přenosové fronty
Název přenosové fronty. Tento atribut je určen jen pro čtení.

Řízení spouštěče
Ovládací prvek spouštěče. Počáteční hodnota závisí na definici fronty. Tento atribut je platný pouze pro lokální frontu.

Data spouątěče
Data spouątěče. Počáteční hodnota závisí na definici fronty. Tento atribut je platný pouze pro lokální frontu.

Hloubka spouątěče
Hloubka spouątěče. Počáteční hodnota závisí na definici fronty. Tento atribut je platný pouze pro lokální frontu.

Priorita zpráv spouątěče
Priorita zprávy prahové hodnoty pro spouątěče. Počáteční hodnota závisí na definici fronty. Tento atribut je platný pouze pro lokální frontu.

typ spouštěče
Typ spouątěče. Počáteční hodnota závisí na definici fronty. Tento atribut je platný pouze pro lokální frontu.

Využití
Použití. Tento atribut je určen jen pro čtení.

Konstruktory

ImqQueue( );
Výchozí konstruktor.

ImqQueue(const ImqQueue & ***queue*** **);**
Konstruktor kopírování. ImqObject **stav otevření** bude FALSE.

ImqQueue(const char * ***name*** **);**
Nastaví ImqObject **název**.

Metody objektů (veřejné)

void operator = (const ImqQueue & ***queue*** **);**
V případě potřeby provede zavření a poté zkopíruje data instance z *fronty*. ImqObject **stav otevření** bude FALSE.

ImqBoolean backoutRequeueName (ImqString & ***name*** **);**
Poskytuje kopii **backout requeue name**. V případě úspěchu vrátí hodnotu TRUE.

ImqString backoutRequeueNázev ();
Vrací **název vrácení** bez označení možných chyb.

ImqBoolean backoutThreshold (MQLONG & ***threshold*** **);**
Poskytuje kopii **prahové hodnoty vrácení**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG backoutThreshold ();
Vrací hodnotu **práh vrácení** bez jakékoli indikace možných chyb.

ImqBoolean baseQueueNázev (ImqString & ***name*** **);**
Poskytuje kopii **názvu základní fronty**. V případě úspěchu vrátí hodnotu TRUE.

ImqString baseQueueName ();
Vrací **název základní fronty** bez označení možných chyb.

ImqBoolean clusterName(ImqString & ***název*** **);**
Poskytuje kopii **názvu klastru**. V případě úspěchu vrátí hodnotu TRUE.

ImqString clusterName( );
Vrací **název klastru** bez jakéhokoli označení možných chyb.

ImqBoolean clusterNamelistNázev (ImqString & ***name*** **);**
Poskytuje kopii **názvu seznamu názvů klastru**. V případě úspěchu vrátí hodnotu TRUE.

ImqString clusterNamelistName ();
Vrací **název seznamu názvů klastru** bez označení chyb.

ImqBoolean clusterWorkLoadPriority (MQLONG & priority);
Poskytuje kopii hodnoty priority pracovní zátěže klastru. V případě úspěchu vrátí hodnotu TRUE.

MQLONG clusterWorkLoadPriority ();
Vrací hodnotu priority pracovní zátěže klastru bez jakékoli indikace možných chyb.

ImqBoolean clusterWorkLoadRank (MQLONG & rank);
Poskytuje kopii hodnoty kategorie pracovní zátěže klastru. V případě úspěchu vrátí hodnotu TRUE.

MQLONG clusterWorkLoadRank ();
Vrací hodnotu očíslování pořadí pracovní zátěže klastru bez jakékoli indikace možných chyb.

ImqBoolean clusterWorkLoadUseQ (MQLONG & useq);
Poskytuje kopii hodnoty fronty využití pracovní zátěže klastru. V případě úspěchu vrátí hodnotu TRUE.

MQLONG clusterWorkLoadUseQ ();
Vrací hodnotu fronty použití pracovní zátěže klastru bez jakékoli indikace možných chyb.

ImqBoolean creationDate (ImqString & ***datum*** **);**
Poskytuje kopii **data vytvoření**. V případě úspěchu vrátí hodnotu TRUE.

ImqString creationDate ( );
Vrací **datum vytvoření** bez jakékoli indikace možných chyb.

ImqBoolean creationTime (ImqString & ***čas*** **);**
Poskytuje kopii **času vytvoření**. V případě úspěchu vrátí hodnotu TRUE.

ImqString creationTime ( );
Vrací **čas vytvoření** bez označení možných chyb.

ImqBoolean currentDepth (MQLONG & ***hloubka*** **);**
Poskytuje kopii **aktuální hloubky**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG currentDepth ();
Vrací **aktuální hloubku** bez jakékoli indikace možných chyb.

ImqBoolean defaultInputOpenOption (& MQLONG ***volba*** **);**
Poskytuje kopii **výchozí volby otevření vstupu**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG defaultInputOpenOption ();
Vrací **výchozí volbu otevření vstupu** bez jakékoli indikace možných chyb.

ImqBoolean defaultPersistence (MQLONG & ***perzistence*** **);**
Poskytuje kopii **výchozí perzistence**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG defaultPersistence ();
Vrací **výchozí perzistenci** bez jakékoli indikace možných chyb.

ImqBoolean defaultPriority (MQLONG & ***priorita*** **);**
Poskytuje kopii **výchozí priority**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG defaultPriority ();
Vrací **výchozí prioritu** bez jakékoli indikace možných chyb.

ImqBoolean defaultBind (MQLONG & ***bind*** **);**
Poskytuje kopii **výchozí vazby**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG defaultBind ();
Vrací **výchozí vazbu** bez jakékoli indikace možných chyb.

ImqBoolean definitionType (MQLONG & ***typ*** **);**
Poskytuje kopii **typu definice**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG definitionType ();
Vrací **typ definice** bez jakéhokoli označení možných chyb.

ImqBoolean depthHighEvent (MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení **události s vysokou hloubkou**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG depthHighUdálost ();
Vrací stav povolení **události s vysokou hloubkou** bez označení možných chyb.

ImqBoolean depthHighLimit (MQLONG & ***limit*** **);**
Poskytuje kopii **vysokého limitu hloubky**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG depthHighLimit ();
Vrátí hodnotu **horní mez hloubky** bez jakékoli indikace možných chyb.

ImqBoolean depthLowUdálost (MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení **události dolní hloubky**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG depthLowUdálost ();
Vrací stav zpřístupnění **události hloubkového minima** bez označení možných chyb.

ImqBoolean depthLowLimit (MQLONG & ***limit*** **);**
Poskytuje kopii **dolního limitu hloubky**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG depthLowLimit ();
Vrací hodnotu **dolní mez hloubky** bez jakékoli indikace možných chyb.

ImqBoolean depthMaximumUdálost (MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení **události maxima hloubky**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG depthMaximumUdálost ();
Vrací stav povolení **události maxima hloubky** bez označení možných chyb.

ImqDistributionSeznam * distributionListReference () const ;
Vrátí **referenci rozdělovníku**.

void setDistributionListReference (ImqDistributionList & *list*);
Nastaví **odkaz na distribuční seznam**.

void setDistributionListReference (ImqDistributionList * *list* = 0);
Nastaví **odkaz na distribuční seznam**.

ImqBoolean distributionLists (& MQLONG *podpora*);
Poskytuje kopii hodnoty **rozdělovníků** . V případě úspěchu vrátí hodnotu TRUE.

MQLONG distributionLists ();
Vrátí hodnotu **rozdělovníků** bez jakéhokoli označení možných chyb.

ImqBoolean setDistributionSeznamy (const MQLONG *podpora*);
Nastaví hodnotu **rozdělovníky** . V případě úspěchu vrátí hodnotu TRUE.

ImqString dynamicQueueNázev () const ;
Vrací kopii **názvu dynamické fronty**.

ImqBoolean setDynamicQueueName (const char * *name*);
Nastaví **název dynamické fronty**. **Název dynamické fronty** lze nastavit pouze v případě, že ImqObject **stav otevření** je FALSE. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean get (ImqMessage & *msg*, ImqGetMessageOptions & *volby*);
Načte zprávu z fronty pomocí uvedených *voleb*. V případě potřeby vyvolá metodu ImqObject **openFor** , aby se zajistilo, že ImqObject **otevřené volby** budou obsahovat jednu z hodnot MQOO_INPUT_ * nebo hodnotu MQOO_BROWSE v závislosti na *volbách*. Má-li objekt *zpráva* hodnotu ImqCache **automatická vyrovnávací paměť**, vyrovnávací paměť se zvětšuje tak, aby pojmula všechny načtené zprávy. Metoda **clearMessage** je vyvolána proti objektu *msg* před načtením.

Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

Poznámka: Výsledek vyvolání metody je FALSE, pokud je ImqObject **kód příčiny** MQRC_TRUNCATED_MSG_FAILED, i když je tento **kód příčiny** klasifikován jako varování. Je-li přijata ořízntutá zpráva, ImqCache **délka zprávy** odráží zkrácenou délku. V obou případech ImqMessage **celková délka zprávy** označuje počet bajtů, které byly k dispozici.

ImqBoolean get (ImqMessage & *msg*);
Stejně jako u předchozí metody, s výjimkou toho, že jsou použity výchozí volby získání zprávy.

ImqBoolean get (ImqMessage & *msg*, ImqGetMessageOptions & *volby*, const size_t *velikost vyrovnávací paměti*);
Stejně jako u předchozích dvou metod, kromě toho, že je označena přepisující *velikost vyrovnávací paměti* . Pokud objekt *zpráva* používá ImqCache **automatická vyrovnávací paměť**, je metoda **resizeBuffer** vyvolána na objektu *zpráva* před načtením zprávy a vyrovnávací paměť se dále nezvyšuje, aby pojmula větší zprávu.

ImqBoolean get (ImqMessage & *msg*, const size_t *velikost vyrovnávací paměti*);
Stejně jako u předchozí metody, s výjimkou toho, že jsou použity výchozí volby získání zprávy.

ImqBoolean hardenGetBackout (MQLONG & *harden*);
Poskytuje kopii hodnoty **harden get backout** . V případě úspěchu vrátí hodnotu TRUE.

MQLONG hardenGetBackout ();
Vrátí hodnotu **harden get backout** bez označení možných chyb.

ImqBoolean indexType(MQLONG & *typ, typ*);
Poskytuje kopii **typu indexu**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG indexType();
Vrací **typ indexu** bez označení možných chyb.

ImqBoolean inhibitGet (MQLONG & *inhibovat*);
Poskytuje kopii hodnoty **inhibovat získání** . V případě úspěchu vrátí hodnotu TRUE.

MQLONG inhibitGet ();
Vrátí hodnotu **inhibit get** bez jakékoli indikace možných chyb.

ImqBoolean setInhibitGet (const MQLONG *inhibit*);
Nastaví hodnotu **inhibovat získání** . V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean inhibitPut (MQLONG &*inhibit*);
Poskytuje kopii hodnoty **zablokovat vložení** . V případě úspěchu vrátí hodnotu TRUE.

MQLONG inhibitPut ();
Vrátí hodnotu **inhibují vložení** bez jakékoli indikace možných chyb.

ImqBoolean setInhibitPut (const MQLONG *inhibit*);
Nastaví hodnotu **zablokovat vložení** . V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean initiationQueueNázev (ImqString &*název*);
Poskytuje kopii **názvu inicializační fronty**. V případě úspěchu vrátí hodnotu TRUE.

ImqString initiationQueueNázev ();
Vrací **název inicializační fronty** bez označení možných chyb.

ImqBoolean maximumDepth (MQLONG &*hloubka*);
Poskytuje kopii **maximální hloubky**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumDepth ();
Vrací **maximální hloubku** bez jakékoli indikace možných chyb.

ImqBoolean maximumMessageDélka (MQLONG &*délka*);
Poskytuje kopii **maximální délky zprávy**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumMessageLength ();
Vrací **maximální délku zprávy** bez jakékoli indikace možných chyb.

ImqBoolean messageDeliverySequence (MQLONG &*sequence*);
Poskytuje kopii **posloupnosti doručení zpráv**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG messageDeliverySequence ();
Vrátí hodnotu **pořadí doručení zprávy** bez jakéhokoli označení možných chyb.

ImqQueue * nextDistributedQueue () const ;
Vrátí **další distribuovanou frontu**.

ImqBoolean nonPersistentMessageClass (MQLONG & monq);
Poskytuje kopii hodnoty dočasné třídy zpráv. V případě úspěchu vrátí hodnotu TRUE.

MQLONG nonPersistentMessageClass ();
Vrací dočasnou hodnotu třídy zpráv bez jakékoli indikace možných chyb.

ImqBoolean openInputCount (MQLONG &*count*);
Poskytuje kopii **počtu otevřených vstupů**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG openInputPočet ();
Vrací **počet otevřených vstupů** bez jakékoli indikace možných chyb.

ImqBoolean openOutputCount (MQLONG &*count*);
Poskytuje kopii **počtu otevřených výstupů**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG openOutputPočet ();
Vrací **počet otevřených výstupů** bez jakékoli indikace možných chyb.

ImqQueue * previousDistributedQueue () const ;
Vrátí **předchozí distribuovanou frontu**.

ImqBoolean processName (ImqString &*název*);
Poskytuje kopii **názvu procesu**. V případě úspěchu vrátí hodnotu TRUE.

ImqString processName ( );
Vrací **název procesu** bez jakékoli indikace možných chyb.

ImqBoolean put (ImqMessage &*msg*);
Umístí zprávu do fronty s použitím výchozích voleb vložení zprávy. V případě potřeby použije metodu ImqObject **openFor** , aby se zajistilo, že ImqObject **volby otevření** budou obsahovat MQOOO_OUTPUT.

Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean put (ImqMessage &*msg*, ImqPutMessageOptions &*pmo*);
Umístí zprávu do fronty pomocí uvedeného *pmo*. Podle potřeby používá metodu ImqObject **openFor** , aby se zajistilo, že ImqObject **volby otevření** budou obsahovat MQOOO_OUTPUT a (pokud *pmo*

volby zahrnují některou z hodnot MQPMO_PASS_IDENTITY_CONTEXT, MQPMO_PASS_ALL_CONTEXT, MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT) odpovídající hodnoty MQOO_ * _CONTEXT.

Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

Poznámka: Pokud *pmo* obsahuje **odkaz na kontext**, otevře se odkazovaný objekt, je-li to nezbytné, pro poskytnutí kontextu.

ImqBoolean queueAccounting (MQLONG & acctq);
Poskytuje kopii hodnoty evidence front. V případě úspěchu vrátí hodnotu TRUE.

MQLONG queueAccounting ();
Vrátí hodnotu evidence front bez jakéhokoli označení možných chyb.

ImqString queueManagerNázev () const ;
Vrací **název správce front**.

ImqBoolean setQueueManagerName (const char * *name*);
Nastaví **název správce front**. **Název správce front** lze nastavit pouze v případě, že ImqObject **stav otevření** je FALSE. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean queueMonitoring (MQLONG & monq);
Poskytuje kopii hodnoty monitorování fronty. V případě úspěchu vrátí hodnotu TRUE.

MQLONG queueMonitoring ();
Vrací hodnotu monitorování fronty bez jakékoli indikace možných chyb.

ImqBoolean queueStatistics (MQLONG & statq);
Poskytuje kopii hodnoty statistiky fronty. V případě úspěchu vrátí hodnotu TRUE.

MQLONG queueStatistics ();
Vrátí hodnotu statistiky fronty bez jakékoli indikace možných chyb.

ImqBoolean queueType (MQLONG & *typ*);
Poskytuje kopii hodnoty **typ fronty** . V případě úspěchu vrátí hodnotu TRUE.

MQLONG queueType ();
Vrací **typ fronty** bez indikace možných chyb.

ImqBoolean remoteQueueManagerName (ImqString & *name*);
Poskytuje kopii **názvu vzdáleného správce front**. V případě úspěchu vrátí hodnotu TRUE.

ImqString remoteQueueManagerName ( );
Vrací **název vzdáleného správce front** bez jakékoli indikace možných chyb.

ImqBoolean remoteQueueNázev (ImqString & *name*);
Poskytuje kopii **názvu vzdálené fronty**. V případě úspěchu vrátí hodnotu TRUE.

ImqString remoteQueueName ();
Vrací **název vzdálené fronty** bez označení možných chyb.

ImqBoolean resolvedQueueManagerName(ImqString & *name*);
Poskytuje kopii **rozlišeného názvu správce front**. V případě úspěchu vrátí hodnotu TRUE.

Poznámka: Tato metoda selže, pokud MQOO_RESOLVE_NAMES není mezi ImqObject **otevřenými volbami**.

ImqString resolvedQueueManagerName( ) ;
Vrací **vyřešený název správce front**bez označení možných chyb.

ImqBoolean resolvedQueueNázev (ImqString & *name*);
Poskytuje kopii **rozlišeného názvu fronty**. V případě úspěchu vrátí hodnotu TRUE.

Poznámka: Tato metoda selže, pokud MQOO_RESOLVE_NAMES není mezi ImqObject **otevřenými volbami**.

ImqString resolvedQueueName ();
Vrátí **vyřešený název fronty**bez označení možných chyb.

ImqBoolean retentionInterval (MQLONG & *interval*);
Poskytuje kopii **intervalu uchování**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG retentionInterval ();
Vrací **interval uchování** bez jakékoli indikace možných chyb.

ImqBoolean rozsah (MQLONG &*rozsah*);
Poskytuje kopii **rozsahu**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG scope ();
Vrací **rozsah** bez jakéhokoli označení možných chyb.

ImqBoolean serviceInterval (MQLONG &*interval*);
Poskytuje kopii **intervalu služby**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG serviceInterval ();
Vrací **servisní interval** bez jakéhokoli označení možných chyb.

ImqBoolean serviceIntervalUdálost (MQLONG &*událost*);
Poskytuje kopii stavu povolení **události intervalu služby**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG serviceIntervalUdálost ();
Vrací stav povolení **události intervalu služby** bez označení možných chyb.

ImqBoolean shareability (MQLONG &*shareability*);
Poskytuje kopii hodnoty **shareability** . V případě úspěchu vrátí hodnotu TRUE.

MQLONG shareability ();
Vrátí hodnotu **shareability** bez označení možných chyb.

ImqBoolean storageClass(ImqString &*třída*);
Poskytuje kopii **paměťové třídy**. V případě úspěchu vrátí hodnotu TRUE.

ImqString storageClass( );
Vrací **paměťovou třídu** bez označení možných chyb.

ImqBoolean transmissionQueueNázev (ImqString &*název*);
Poskytuje kopii **názvu přenosové fronty**. V případě úspěchu vrátí hodnotu TRUE.

ImqString transmissionQueueNázev ();
Vrátí **název přenosové fronty** bez označení možných chyb.

ImqBoolean triggerControl (MQLONG &*ovládací prvek*);
Poskytuje kopii hodnoty **ovládacího prvku spouštěče** . V případě úspěchu vrátí hodnotu TRUE.

MQLONG triggerControl ();
Vrací hodnotu **řízení spouštěče** bez jakékoli indikace možných chyb.

ImqBoolean setTriggerControl (const MQLONG *control*);
Nastaví hodnotu **ovládacího prvku spouštěče** . V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean triggerData (ImqString &*data*);
Poskytuje kopii **dat spouštěče**. V případě úspěchu vrátí hodnotu TRUE.

ImqString triggerData ( );
Vrací kopii **dat spouštěče** bez označení možných chyb.

ImqBoolean setTriggerData (const char *&*data*);
Nastaví **data spouštěče**. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean triggerDepth (MQLONG &*hloubka*);
Poskytuje kopii **hloubky spouštěče**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG triggerDepth ();
Vrací **hloubku spouštěče** bez označení možných chyb.

ImqBoolean setTriggerHloubka (const MQLONG *hloubka*);
Nastaví **hloubku spouštěče**. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean triggerMessagePriorita (MQLONG &*priorita*);
Poskytuje kopii **priority zprávy spouštěče**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG triggerMessagePriorita ();
Vrací **prioritu zprávy spouštěče** bez označení možných chyb.

ImqBoolean setTriggerMessagePriority (const MQLONG *priority*);
Nastaví **prioritu zprávy spouštěče**. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean triggerType (MQLONG &*typ*);
Poskytuje kopii **typu spouštěče**. V případě úspěchu vrátí hodnotu TRUE.

MQLONG triggerType ();
Vrací **typ spouštěče** bez jakéhokoli označení možných chyb.

ImqBoolean setTriggerTyp (const MQLONG *typ*);
Nastaví **typ spouštěče**. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean použití (MQLONG &*použití*);
Poskytuje kopii hodnoty **využití** . V případě úspěchu vrátí hodnotu TRUE.

MQLONG použití ();
Vrátí hodnotu **usage** bez jakékoli indikace možných chyb.

Metody objektů (chráněné)

void setNextDistributedQueue (ImqQueue * *queue* = 0);
Nastaví **další distribuovanou frontu**.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nepřeruší seznam distribuovaných front.

void setPreviousDistributedQueue (ImqQueue * *queue* = 0);
Nastaví **předchozí distribuovanou frontu**.

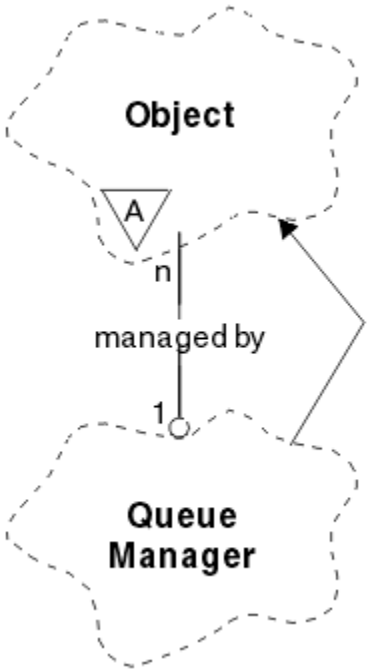
Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nepřeruší seznam distribuovaných front.

Kódy příčin

- MQRC_ATTRIBUTE_LOCKED
- MQRC_CONTEXT_OBJECT_NOT_VALID
- MQRC_CONTEXT_OPEN_ERROR
- MQRC_CURSOR_NOT_VALID
- MQRC_NO_BUFFER
- MQRC_REOPEN_EXCL_INPUT_ERROR
- MQRC_REOPEN_INQUIRE_ERROR
- MQRC_REOPEN_TEMPORARY_Q_ERROR
- (kódy příčiny z MQGET)
- (kódy příčiny z MQPUT)

Třída ImqQueueManager C++

Tato třída zapouzdřuje správce front (objekt IBM MQ typu MQOT_Q_MGR).



Obrázek 33. Třída správce ImqQueue

Tato třída souvisí s voláními MQI uvedenými v seznamu “Křížový odkaz správce ImqQueue” na stránce 1789. Ne všechny uvedené metody jsou použitelné pro všechny platformy; další podrobnosti viz ALTER QMGR .

Atributy třídy

chování
Řídí chování implicitního připojení a odpojení.

IMQ_EXPL_DISC_BACKOUT (0L)
Explicitní volání metody odpojení znamená odvolání. Tento atribut se vzájemně vylučuje s IMQ_EXPL_DISC_COMMIT.

IMQ_EXPL_DISC_COMMIT (1L)
Explicitní volání metody disconnect znamená commit (výchozí). Tento atribut se vzájemně vylučuje s atributem IMQ_EXPL_DISC_BACKOUT.

IMQ_IMPL_CONN (2L)
Implicitní připojení je povoleno (předvolba).

IMQ_IMPL_DISC_BACKOUT (0L)
Implicitní volání metody odpojení, které se může vyskytnout během zničení objektu, znamená vrácení zpět. Tento atribut se vzájemně vylučuje s atributem IMQ_IMPL_DISC_COMMIT.

IMQ_IMPL_DISC_COMMIT (4L)
Implicitní volání metody disconnect, které se může vyskytnout během zničení objektu, znamená commit (výchozí). Tento atribut se vzájemně vylučuje s IMQ_IMPL_DISC_BACKOUT.

Ve verzi IBM MQ V7.0 a vyšší, aplikace C++, které používají implicitní připojení, musí uvést IMQ_IMPL_CONN spolu s dalšími volbami poskytnutými v metodě `setBehavior()` na objektu třídy `ImqQueueManager`. Pokud vaše aplikace nepoužívá metodu `setBehavior()` k explicitnímu nastavení voleb chování, například:

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

tato změna vás neovlivní, protože MQ_IMPL_CONN je standardně povoleno.

Pokud vaše aplikace explicitně nastaví volby chování, například:

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

musíte zahrnout IMQ_IMPL_CONN do metody `setBehavior()` následujícím způsobem, aby vaše aplikace mohla dokončit implicitní připojení:

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_CONN | IMQ_IMPL_DISC_COMMIT)
```

Atributy objektů

přepsání evidenčních připojení
Umožňuje aplikacím přepsat nastavení hodnot values.This je jen pro čtení.

Interval evidence
Jak dlouho před zápisem mezilehlých účetních záznamů (v sekundách). Tento atribut je určen jen pro čtení.

Záznam činnosti
Ovládá generování sestav aktivity. Tento atribut je určen jen pro čtení.

Převzetí nového agenta MCA - kontrola
Prvky kontrolované za účelem určení, zda má být agent MCA převzat při zjištění nového příchozího kanálu, který má stejný název jako agent MCA, který je již aktivní. Tento atribut je určen jen pro čtení.

Převzetí nového agenta MCA - typ
Zda má být osiřelá instance agenta MCA konkrétního typu kanálu automaticky restartována, když je zjištěn nový příchozí požadavek kanálu odpovídající parametrům kontroly převzetí nové mca. Tento atribut je určen jen pro čtení.

Typ ověřování
Označuje typ ověřování, které se provádí.

událost oprávnění
Řídí události oprávnění. Tento atribut je určen jen pro čtení.

volby začátku
Volby, které platí pro metodu zahájení. Počáteční hodnota je MQBO_NONE.

událost mostu
Zda jsou generovány události mostu IMS . Tento atribut je určen jen pro čtení.

Automatická definice kanálů
Hodnota automatické definice kanálu. Tento atribut je určen jen pro čtení.

Událost automatické definice kanálu
Hodnota události automatické definice kanálu. Tento atribut je určen jen pro čtení.

Uživatelská procedura automatické definice kanálů
Název uživatelské procedury automatické definice kanálu. Tento atribut je určen jen pro čtení.

událost kanálu
Zda jsou generovány události kanálu. Tento atribut je určen jen pro čtení.

Adaptéry inicializátoru kanálu
Počet dílčích úloh adaptéru, které se mají použít pro zpracování volání IBM MQ . Tento atribut je určen jen pro čtení.

Řízení iniciátoru kanálu
Určuje, zda má být inicializátor kanálu automaticky spuštěn při spuštění správce front. Tento atribut je určen jen pro čtení.

Dispečeři inicializátoru kanálu
Počet dispečerů, které mají být použity pro inicializátor kanálu. Tento atribut je určen jen pro čtení.

automatické spuštění trasování inicializátoru kanálu
Určuje, zda se má trasování inicializátoru kanálu spouštět automaticky či nikoli. Tento atribut je určen jen pro čtení.

Velikost tabulky trasování inicializátoru kanálu
Velikost prostoru dat trasování inicializátoru kanálu (v MB). Tento atribut je určen jen pro čtení.

Monitorování kanálů
Ovládá shromažďování online monitorovacích dat pro kanály. Tento atribut je určen jen pro čtení.

odkaz na kanál
Odkaz na definici kanálu pro použití během připojení klienta. Při připojení lze tento atribut nastavit na hodnotu null, ale nelze jej změnit na žádnou jinou hodnotu. Počáteční hodnota je null.

Statistika kanálů
Ovládá shromažďování statistických dat kanály. Tento atribut je určen jen pro čtení.

znaková sada
Identifikátor kódované znakové sady (CCSID). Tento atribut je určen jen pro čtení.

Monitorování odesílatele klastru
Řídí shromažďování online dat monitorování pro automaticky definované odesílací kanály klastru. Tento atribut je určen jen pro čtení.

Statistiky odesílatele klastru
Řídí shromažďování statistických dat pro automaticky definované odesílací kanály klastru. Tento atribut je určen jen pro čtení.

Data pracovní zátěže klastru
Data uživatelské procedury pracovní zátěže klastru. Tento atribut je určen jen pro čtení.

Uživatelská procedura pracovní zátěže klastru
Název uživatelské procedury pracovní zátěže klastru. Tento atribut je určen jen pro čtení.

Délka pracovní zátěže klastru
Délka pracovní zátěže klastru. Tento atribut je určen jen pro čtení.

mru pracovní zátěže klastru
Hodnota naposledy použitých kanálů pracovní zátěže klastru. Tento atribut je určen jen pro čtení.

Pracovní zátěž klastru - použitá fronta
Hodnota fronty využití pracovní zátěže klastru. Tento atribut je určen jen pro čtení.

událost příkazu
Zda jsou generovány události příkazu. Tento atribut je určen jen pro čtení.

Název fronty vstupu příkazů
Název vstupní fronty systémového příkazu. Tento atribut je určen jen pro čtení.

Úroveň příkazů
Úroveň příkazů podporovaná správcem front. Tento atribut je určen jen pro čtení.

Řízení příkazového serveru
Určuje, zda má být příkazový server automaticky spuštěn při spuštění správce front. Tento atribut je určen jen pro čtení.

Volby připojení
Volby, které platí pro metodu připojení. Počáteční hodnota je MQCNO_NONE. V závislosti na platformě mohou být možné následující dodatečné hodnoty:

- MQCNO_STANDARDNÍ_VAZBA
- 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 připojení
Jedinečný identifikátor, který umožňuje produktu MQ spolehlivě identifikovat aplikaci.

Stav připojení
TRUE při připojení ke správci front. Tento atribut je určen jen pro čtení.

Značka připojení
Značka, která má být přidružena k připojení. Tento atribut lze nastavit pouze v případě, že není připojen. Počáteční hodnota je null.

Kryptografický hardware
Podrobnosti konfigurace šifrovacího hardwaru. Pro připojení klienta MQI produktu MQ .

název fronty nedoručených zpráv
Název fronty nedoručených zpráv. Tento atribut je určen jen pro čtení.

název výchozí přenosové fronty
Výchozí název přenosové fronty. Tento atribut je určen jen pro čtení.

distribuční seznamy
Schopnost správce front podporovat distribuční seznamy.

skupina dns
Název skupiny, ke které by se měl připojit modul listener protokolu TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, při použití podpory služeb dynamického názvu domény správce pracovní zátěže. Tento atribut je určen jen pro čtení.

dns wlm
Určuje, zda se má modul listener protokolu TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, registrovat ve správci pracovní zátěže pro služby dynamického názvu domény. Tento atribut je určen jen pro čtení.

první záznam ověření
První z jednoho nebo více objektů třídy ImqAuthenticationRecord, v žádném konkrétním pořadí, ve kterém se odkaz na připojení záznamu ImqAuthenticationadresuje na tento objekt. Pro připojení klienta MQI produktu MQ .

první spravovaný objekt
První z jednoho nebo více objektů třídy ImqObject, bez konkrétního pořadí, ve kterém připojení ImqObject odkazuje na tento objekt. Počáteční hodnota je nula.

blokovat událost
Řídí blokování událostí. Tento atribut je určen jen pro čtení.

Verze adresy IP
Který protokol IP (IPv4 nebo IPv6) se má použít pro připojení kanálu. Tento atribut je určen jen pro čtení.

úložiště klíčů
Umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty. Pro připojení IBM MQ MQI client .

počet resetů klíče
Počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS, než bude znovu vyjednán tajný klíč. Tento atribut platí pouze pro připojení klienta pomocí MQCONNX. Viz také počet resetů klíče ssl.

Časovač modulu listener
Časový interval (v sekundách) mezi pokusy produktu IBM MQ o restartování modulu listener, pokud došlo k selhání APPC nebo TCP/IP. Tento atribut je určen jen pro čtení.

lokální událost
Řídí lokální události. Tento atribut je určen jen pro čtení.

Událost modulu protokolování
Řídí, zda jsou generovány události protokolu pro zotavení. Tento atribut je určen jen pro čtení.

Název skupiny LU
Generické jméno LU, které by měl používat modul listener 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front. Tento atribut je určen jen pro čtení.

Název jednotky LU
Název LU, která má být použita pro odchozí přenosy LU 6.2 . Tento atribut je určen jen pro čtení.

lu62 přípona ramene
Přípona SYS1.PARMLIB člen APPCPMxx, který nominuje LUADD pro tento inicializátor kanálu. Tento atribut je určen jen pro čtení.

lu62 kanálů
Maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni, kteří používají přenosový protokol LU 6.2 . Tento atribut je určen jen pro čtení.

maximální počet aktivních kanálů
Maximální počet kanálů, které mohou být současně aktivní. Tento atribut je určen jen pro čtení.

Maximální počet kanálů
Maximální počet kanálů, které mohou být aktuální (včetně kanálů připojení serveru s připojenými klienty). Tento atribut je určen jen pro čtení.

maximální počet manipulátorů
Maximální počet popisovačů. Tento atribut je určen jen pro čtení.

Maximální délka zprávy
Maximální možná délka pro libovolnou zprávu v libovolné frontě spravované tímto správcem front. Tento atribut je určen jen pro čtení.

maximální priorita
Maximální priorita zprávy. Tento atribut je určen jen pro čtení.

Maximum nepotvrzených zpráv
Maximální počet nepotvrzených zpráv v rámci jednotky nebo práce. Tento atribut je určen jen pro čtení.

Evidence MQI
Ovládá shromažďování informací o účtu pro data MQI. Tento atribut je určen jen pro čtení.

Statistika MQI
Ovládá shromažďování informací o monitorování statistiky pro správce front. Tento atribut je určen jen pro čtení.

maximální počet odchozích portů
Vyšší konec rozsahu čísel portů, které mají být použity při vázání odchozích kanálů. Tento atribut je určen jen pro čtení.

minimum odchozího portu
Dolní konec rozsahu čísel portů, které mají být použity při vázání odchozích kanálů. Tento atribut je určen jen pro čtení.

heslo
heslo přidružené k ID uživatele

událost výkonu
Řídí události výkonu. Tento atribut je určen jen pro čtení.

platforma
Platforma, na které je umístěn správce front. Tento atribut je určen jen pro čtení.

Účtování fronty
Ovládá shromažďování informací o účtu pro fronty. Tento atribut je určen jen pro čtení.

Monitorování fronty
Ovládá shromažďování online monitorovacích dat pro fronty. Tento atribut je určen jen pro čtení.

Statistiky fronty
Ovládá shromažďování statistických dat pro fronty. Tento atribut je určen jen pro čtení.

Časový limit pro příjem
Přibližně jak dlouho bude kanál zpráv TCP/IP čekat na příjem dat, včetně prezenčních signálů, od svého partnera, než se vrátí do neaktivního stavu. Tento atribut je určen jen pro čtení.

minimum časového limitu příjmu
Minimální doba, po kterou bude kanál TCP/IP čekat na příjem dat, včetně prezenčních signálů, od svého partnera, než se vrátí do neaktivního stavu. Tento atribut je určen jen pro čtení.

Typ časového limitu pro příjem
Kvalifikátor použitý pro časový limit přijetí. Tento atribut je určen jen pro čtení.

vzdálená událost
Řídí vzdálené události. Tento atribut je určen jen pro čtení.

REPOSITORY NAME
Název úložiště. Tento atribut je určen jen pro čtení.

Seznam názvů úložiště
Název seznamu názvů úložiště. Tento atribut je určen jen pro čtení.

název sdíleného správce front
Určuje, zda mají být položky MQOPENs sdílené fronty, ve které je název ObjectQMgrjiným správcem front ve skupině sdílení front, interpretována jako otevřená sdílená fronta v lokálním správci front. Tento atribut je určen jen pro čtení.

Událost ssl
Zda jsou generovány události SSL. Tento atribut je určen jen pro čtení.

Požadován standard SSL FIPS
Zda by se měly použít pouze algoritmy certifikované FIPS, pokud se šifrování provádí v softwaru IBM MQ . Tento atribut je určen jen pro čtení.

Počet resetování klíče SSL
Počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace SSL, než je znovu vyjednán tajný klíč. Tento atribut je určen jen pro čtení.

událost start-stop
Řídí události start-stop. Tento atribut je určen jen pro čtení.

Interval statistiky
Jak často se data monitorování statistiky zapisují do fronty monitorování. Tento atribut je určen jen pro čtení.

Dostupnost synchronizačního bodu
Dostupnost účasti synchronizačního bodu. Tento atribut je určen jen pro čtení.

Poznámka: Globální jednotky práce koordinované správcem front nejsou na platformě IBM i podporovány. 

Pomocí nativních systémových volání _Rcommit a _Rback můžete naprogramovat jednotku práce, která je externě koordinována produktem IBM i. Spusťte tento typ pracovní jednotky spuštěním aplikace IBM MQ pod vázaným zpracováním na úrovni úlohy pomocí příkazu STRCMTCTL. Další podrobnosti viz Rozhraní pro IBM i externího správce synchronizačního bodu . Vrácení a potvrzení jsou podporována na platformě IBM i pro lokální pracovní jednotky koordinované správcem front.

kanály tcp
Maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni, kteří používají přenosový protokol TCP/IP. Tento atribut je určen jen pro čtení.

TCP - Udržování aktivity
Zda se má prostředek KEEPALIVE protokolu TCP použít ke kontrole, zda je druhý konec připojení stále k dispozici. Tento atribut je určen jen pro čtení.

Název TCP
Název jediného nebo výchozího systému TCP/IP, který se má použít, v závislosti na hodnotě typu zásobníku tcp. Tento atribut je určen jen pro čtení.

Typ sady protokolů TCP
Určuje, zda má inicializátor kanálu povoleno používat pouze adresní prostor TCP/IP uvedený v názvu tcp nebo zda může být svázán s libovolnou vybranou adresou TCP/IP. Tento atribut je určen jen pro čtení.

Záznam přenosových tras
Řídí záznam informací o trasování trasy. Tento atribut je určen jen pro čtení.

Interval spouštěče
Interval spouštěče. Tento atribut je určen jen pro čtení.

Jméno uživatele
Na platformách AIX and Linux se jedná o skutečné ID uživatele aplikace. Na platformách Windows se jedná o ID uživatele aplikace.

Konstruktory

ImqQueueManager ();
Výchozí konstruktor.

ImqQueueManager (const ImqQueueManager & *manager*);
Konstruktor kopírování. Stav připojení bude FALSE.

ImqQueueManager (znak const * *název*);
Nastaví název ImqObject na *name*.

Destruktory

Dojde-li ke zničení objektu ImqQueueManager, dojde k jeho automatickému odpojení.

Metody třídy (veřejné)

statické chování MQLONG ();
Vrátí chování.

void setBehavior(const MQLONG *behavior* = 0);
Nastaví chování.

Metody objektů (veřejné)

void operator = (const ImqQueueManager & *mgr*);
V případě potřeby se odpojí a zkopíruje data instance ze správce *mgr*. Stav připojení je FALSE.

ImqBoolean accountingConnOverride (MQLONG & statint);
Poskytuje kopii hodnoty přepsání evidenčních připojení. V případě úspěchu vrátí hodnotu TRUE.

MQLONG accountingConnOverride ();
Vrátí hodnotu přepsání evidenčních připojení bez jakékoli indikace možných chyb.

ImqBoolean accountingInterval (MQLONG & statint);
Poskytuje kopii hodnoty intervalu účtování. V případě úspěchu vrátí hodnotu TRUE.

MQLONG accountingInterval ();
Vrátí hodnotu intervalu účtování bez jakékoli indikace možných chyb.

ImqBoolean activityRecording (MQLONG & rec);
Poskytuje kopii hodnoty záznamu aktivity. V případě úspěchu vrátí hodnotu TRUE.

MQLONG activityRecording ();
Vrací hodnotu záznamu aktivity bez jakékoli indikace možných chyb.

ImqBoolean adoptNewMCACheck (MQLONG & check);
Poskytuje kopii kontrolní hodnoty převzetí nového agenta MCA. V případě úspěchu vrátí hodnotu TRUE.

MQLONG adoptNewMCACheck ();
Vrací novou hodnotu kontroly převzetí MCA bez jakékoli indikace možných chyb.

ImqBoolean adoptNewMCAType (MQLONG & typ);
Poskytuje kopii adopce nového typu MCA. V případě úspěchu vrátí hodnotu TRUE.

MQLONG adoptNewMCAType ();
Vrací nový typ adopce MCA bez označení možných chyb.

QLONG authenticationType () const;
Vrátí typ ověřování.

void setAuthenticationType (const MQLONG type = MQCSP_AUTH_NONE);
Nastaví typ ověřování.

ImqBoolean authorityEvent(MQLONG & *událost*);
Poskytuje kopii stavu povolení události oprávnění. V případě úspěchu vrátí hodnotu TRUE.

MQLONG authorityEvent();
Vrací stav povolení události oprávnění bez označení možných chyb.

ImqBoolean backout ();
Vrací nepotvrzené změny. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean begin ();
Zahájí jednotku práce. Volby zahájení ovlivňují chování této metody. Vrací hodnotu TRUE, je-li úspěšná, ale vrátí hodnotu TRUE i v případě, že základní volání MQBEGIN vrátí hodnotu MQRC_NO_EXTERNAL_ÚČASTNÍCI nebo MQRC_PARTICIPANT_NOT_AVAILABLE (obě jsou přidruženy k MQCC_WARNING).

MQLONG beginOptions() const;
Vrátí volby začátku.

void setBeginOptions (const MQLONG *options* = MQBO_NONE);
Nastaví volby začátku.

ImqBoolean bridgeEvent (událost MQLONG &);
Poskytuje kopii hodnoty události mostu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG bridgeEvent ();
Vrátí hodnotu události mostu bez jakékoli indikace možných chyb.

ImqBoolean channelAutoDefinice (MQLONG & *hodnota*);
Poskytuje kopii hodnoty automatické definice kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelAutoDefinition ();
Vrací hodnotu automatické definice kanálu bez jakékoli indikace možných chyb.

ImqBoolean channelAutoDefinitionEvent(MQLONG & *hodnota*);
Poskytuje kopii hodnoty události automatické definice kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelAutoDefinitionEvent();
Vrací hodnotu události automatické definice kanálu bez indikace možných chyb.

ImqBoolean channelAutoDefinitionExit(ImqString & *název*);
Poskytuje kopii názvu uživatelské procedury automatické definice kanálu. V případě úspěchu vrátí hodnotu TRUE.

ImqString channelAutoDefinitionExit( );
Vrací název uživatelské procedury automatické definice kanálu bez jakékoli indikace možných chyb.

ImqBoolean channelEvent (událost MQLONG &);
Poskytuje kopii hodnoty události kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelEvent();
Vrací hodnotu události kanálu bez jakékoli indikace možných chyb.

MQLONG channelInitiatorAdapters ();
Vrací hodnotu adaptérů inicializátoru kanálu bez jakékoli indikace možných chyb.

ImqBoolean channelInitiatorAdaptéry (adaptéry MQLONG &);
Poskytuje kopii hodnoty adaptérů inicializátoru kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelInitiatorControl ();
Vrací hodnotu spuštění inicializátoru kanálu bez jakékoli indikace možných chyb.

ImqBoolean channelInitiatorControl (MQLONG & init);
Poskytuje kopii spouštěcí hodnoty řízení inicializátoru kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelInitiatorDispatchers ();
Vrací hodnotu dispečerů inicializátoru kanálu bez jakékoli indikace možných chyb.

ImqBoolean channelInitiatorDispatchers (MQLONG & dispečery);
Poskytuje kopii hodnoty dispečerů inicializátoru kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelInitiatorTraceAutoStart ();
Vrací hodnotu automatického spuštění trasování inicializátoru kanálu bez jakékoli indikace možných chyb.

ImqBoolean channelInitiatorTraceAutoSpuštění (MQLONG & auto);
Poskytuje kopii hodnoty automatického spuštění trasování inicializátoru kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelInitiatorTraceTableSize ();
Vrací hodnotu velikosti tabulky trasování inicializátoru kanálu bez indikace možných chyb.

ImqBoolean channelInitiatorTraceTableVelikost (MQLONG & velikost);
Poskytuje kopii hodnoty velikosti tabulky trasování inicializátoru kanálu. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean channelMonitoring (MQLONG & monchl);
Poskytuje kopii hodnoty monitorování kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelMonitoring ();
Vrací hodnotu monitorování kanálu bez jakékoli indikace možných chyb.

ImqBoolean channelReference(ImqChannel * & ***pchannel*** **);**
Poskytuje kopii odkazu na kanál. Pokud je odkaz na kanál neplatný, nastaví *pchannel* na hodnotu null. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqChannel * channelReference( );
Vrací odkaz na kanál bez jakékoli indikace možných chyb.

ImqBoolean setChannelReference (ImqChannel & ***kanál*** **);**
Nastaví odkaz na kanál. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setChannelReference (ImqChannel * ***kanál*** **= 0);**
Nastaví nebo resetuje odkaz na kanál. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean channelStatistics (MQLONG & statchl);
Poskytuje kopii hodnoty statistiky kanálu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG channelStatistics ();
Vrací hodnotu statistiky kanálu bez jakékoli indikace možných chyb.

ImqBoolean characterSet(MQLONG & ***ccsid*** **);**
Poskytuje kopii znakové sady. V případě úspěchu vrátí hodnotu TRUE.

MQLONG characterSet();
Vrátí kopii znakové sady bez jakékoli indikace možných chyb.

MQLONG clientSslKeyResetCount () const;
Vrací hodnotu počtu resetů klíče SSL použitou pro připojení klienta.

void setClientSslKeyResetCount(počet const MQLONG);
Nastaví počet resetů klíče SSL použitých pro připojení klienta.

ImqBoolean clusterSenderMonitorování (MQLONG & monacls);
Poskytuje kopii výchozí hodnoty monitorování odesilatele klastru. V případě úspěchu vrátí hodnotu TRUE.

MQLONG clusterSenderMonitoring ();
Vrací výchozí hodnotu monitorování odesilatele klastru bez jakékoli indikace možných chyb.

ImqBoolean clusterSender(MQLONG & statacls);
Poskytuje kopii hodnoty statistiky odesílatele klastru. V případě úspěchu vrátí hodnotu TRUE.

MQLONG clusterSenderStatistics ();
Vrací hodnotu statistiky odesilatele klastru bez jakékoli indikace možných chyb.

ImqBoolean clusterWorkload(ImqString & ***data*** **);**
Poskytuje kopii dat uživatelské procedury pracovní zátěže klastru. V případě úspěchu vrátí hodnotu TRUE.

ImqString clusterWorkloadData ();
Vrací data uživatelské procedury pracovní zátěže klastru bez jakékoli indikace možných chyb.

ImqBoolean clusterWorkloadKonec (ImqString & ***název*** **);**
Poskytuje kopii názvu uživatelské procedury pracovní zátěže klastru. V případě úspěchu vrátí hodnotu TRUE.

ImqString clusterWorkloadExit ();
Vrací název uživatelské procedury pracovní zátěže klastru bez jakékoli indikace možných chyb.

ImqBoolean clusterWorkload(MQLONG & ***délka*** **);**
Poskytuje kopii délky pracovní zátěže klastru. V případě úspěchu vrátí hodnotu TRUE.

MQLONG clusterWorkloadLength ();
Vrací délku pracovní zátěže klastru bez jakékoli indikace možných chyb.

ImqBoolean clusterWorkLoadMRU (MQLONG & mru);
Poskytuje kopii hodnoty naposledy použitých kanálů pracovní zátěže klastru. V případě úspěchu vrátí hodnotu TRUE.

MQLONG clusterWorkLoadMRU ();
Vrátí hodnotu naposledy použitých kanálů pracovní zátěže klastru bez označení možných chyb.

ImqBoolean clusterWorkLoadUseQ (MQLONG & useq);
Poskytuje kopii hodnoty fronty využití pracovní zátěže klastru. V případě úspěchu vrátí hodnotu TRUE.

MQLONG clusterWorkLoadUseQ ();
Vrací hodnotu fronty použití pracovní zátěže klastru bez jakékoli indikace možných chyb.

ImqBoolean commandEvent (událost MQLONG &);
Poskytuje kopii hodnoty události příkazu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG commandEvent ();
Vrací hodnotu události příkazu bez označení možných chyb.

ImqBoolean commandInputQueueName(ImqString & ***name*** **);**
Poskytuje kopii názvu vstupní fronty příkazu. V případě úspěchu vrátí hodnotu TRUE.

ImqString commandInputQueueName( );
Vrací název vstupní fronty příkazu bez označení možných chyb.

ImqBoolean commandLevel(MQLONG & ***úroveň*** **);**
Poskytuje kopii úrovně příkazu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG commandLevel();
Vrátí úroveň příkazu bez označení možných chyb.

MQLONG commandServerControl ();
Vrací hodnotu spuštění příkazového serveru bez jakékoli indikace možných chyb.

ImqBoolean commandServerControl (MQLONG & server);
Poskytuje kopii spouštěcí hodnoty řízení příkazového serveru. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean commit ();
Potvrdí nepotvrzené změny. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean connect ();
Připojí se ke správci front s daným názvem ImqObject , přičemž výchozím názvem je lokální správce front. Chcete-li se připojit ke specifickému správci front, použijte před připojením metodu ImqObject setName . Pokud existuje odkaz na kanál, používá se k předávání informací o definici kanálu do MQCONNX v MQCD. ChannelType v MQCD je nastaven na MQCHT_CLNTCONN. Referenční informace kanálu, které mají význam pouze pro připojení klienta, jsou pro připojení serveru ignorovány. Volby připojení ovlivňují chování této metody. Tato metoda nastaví stav připojení na TRUE, pokud je úspěšné. Vrátí nový stav připojení.

Pokud existuje první záznam ověření, použije se řetěz záznamů ověření k ověření digitálních certifikátů pro zabezpečené klientské kanály.

Ke stejnému správci front lze připojit více objektů ImqQueueManager. Všechny používají stejný manipulátor připojení MQHCONN a sdílejí funkčnost UOW pro připojení přidružené k podprocesu. První správce ImqQueueManager, který se připojil, získá manipulátor MQHCONN. Poslední správce ImqQueue, který provedl odpojení, provede příkaz MQDISC.

Pro program s podporou podprocesů se doporučuje použít pro každý podproces samostatný objekt správce ImqQueue.

ImqBinary connectionId () const;
Vrátí ID připojení.

ImqBinary connectionTag () const;
Vrací značku připojení.

ImqBoolean setConnectionZnačka (const MQBYTE128 *tag* = 0);
Nastaví značku připojení. Je-li *tag* nula, vymaže značku připojení. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setConnectionZnačka (const ImqBinary & *tag*);
Nastaví značku připojení. Délka dat *značky* musí být buď nula (pro vymazání značky připojení) nebo MQ_CONN_TAG_LENGTH. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

MQLONG connectOptions() const;
Vrátí volby připojení.

void setConnectOptions (const MQLONG *options* = MQCNO_NONE);
Nastaví volby připojení.

ImqBoolean connectionStatus() const;
Vrátí stav připojení.

ImqString cryptographicHardware ( );
Vrátí kryptografický hardware.

ImqBoolean setCryptographicHardware (const char * *hardware* = 0);
Nastaví kryptografický hardware. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean deadLetterQueueName(ImqString & *name*);
Poskytuje kopii názvu fronty nedoručených zpráv. V případě úspěchu vrátí hodnotu TRUE.

ImqString deadLetterQueueName( );
Vrací kopii názvu fronty nedoručených zpráv bez jakékoli indikace možných chyb.

ImqBoolean defaultTransmissionQueueName(ImqString & *name*);
Poskytuje kopii výchozího názvu přenosové fronty. V případě úspěchu vrátí hodnotu TRUE.

ImqString defaultTransmissionQueueName( );
Vrátí výchozí název přenosové fronty bez indikace možných chyb.

ImqBoolean disconnect ();
Odpojí se od správce front a nastaví stav připojení na FALSE. Zavře všechny objekty ImqProcess a ImqQueue přidružené k tomuto objektu a před odpojením jejich odkaz na připojení. Je-li ke stejnému správci front připojen více než jeden objekt ImqQueueManager, provede fyzické odpojení

pouze poslední odpojený objekt; ostatní objekty provedou logické odpojení. Nepotvrzené změny jsou potvrzeny pouze při fyzickém odpojení.

Tato metoda vrátí hodnotu TRUE, pokud je úspěšná. Pokud je volána, když neexistuje žádné připojení, návratový kód je také true.

ImqBoolean distributionLists(MQLONG & ***podpora*** **);**
Poskytuje kopii hodnoty distribučních seznamů. V případě úspěchu vrátí hodnotu TRUE.

MQLONG distributionLists();
Vrátí hodnotu rozdělovníku bez jakékoli indikace možných chyb.

ImqBoolean dnsGroup (ImqString & group);
Poskytuje kopii názvu skupiny DNS. V případě úspěchu vrátí hodnotu TRUE.

ImqString dnsGroup ( );
Vrátí název skupiny DNS bez jakékoli indikace možných chyb.

ImqBoolean dnsWlm (MQLONG & wlm);
Poskytuje kopii hodnoty WLM DNS. V případě úspěchu vrátí hodnotu TRUE.

MQLONG dnsWlm ();
Vrátí hodnotu WLM DNS bez jakékoli indikace možných chyb.

ImqAuthenticationRecord * firstAuthenticationRecord () const;
Vrátí první záznam ověření.

void setFirstAuthenticationRecord (const ImqAuthenticationRecord * ***air*** **= 0);**
Nastaví první záznam ověření.

ImqObject * firstManagedObject () const;
Vrátí první spravovaný objekt.

ImqBoolean inhibitEvent(MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení blokovací události. V případě úspěchu vrátí hodnotu TRUE.

MQLONG inhibitEvent();
Vrací stav povolení blokovací události bez jakékoli indikace možných chyb.

ImqBoolean ipAddressVerze (MQLONG & verze);
Poskytuje kopii hodnoty verze adresy IP. V případě úspěchu vrátí hodnotu TRUE.

MQLONG ipAddressVerze ();
Vrátí hodnotu verze adresy IP bez uvedení možných chyb.

ImqBoolean keepAlive (MQLONG & keepalive);
Poskytuje kopii hodnoty udržení aktivity. V případě úspěchu vrátí hodnotu TRUE.

MQLONG keepAlive ();
Vrátí hodnotu udržení aktivity bez jakékoli indikace možných chyb.

ImqString keyRepository ( );
Vrátí úložiště klíčů.

ImqBoolean setKeyÚložiště (const char * ***repository*** **= 0);**
Nastaví úložiště klíčů. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean listenerTimer (& časovač MQLONG);
Poskytuje kopii hodnoty časovače modulu listener. V případě úspěchu vrátí hodnotu TRUE.

MQLONG listenerTimer ();
Vrací hodnotu časovače modulu listener bez jakékoli indikace možných chyb.

ImqBoolean localEvent(MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení lokální události. V případě úspěchu vrátí hodnotu TRUE.

MQLONG localEvent();
Vrací stav povolení lokální události bez jakékoli indikace možných chyb.

ImqBoolean loggerEvent (MQLONG & count);
Poskytuje kopii hodnoty události modulu protokolování. V případě úspěchu vrátí hodnotu TRUE.

MQLONG loggerEvent ();
Vrací hodnotu události modulu protokolování bez jakékoli indikace možných chyb.

ImqBoolean luGroupNázev (ImqString & name);.
Poskytuje kopii názvu skupiny LU. V případě úspěchu vrátí hodnotu TRUE.

ImqString luGroupName ();
Vrátí název skupiny LU bez označení možných chyb.

ImqBoolean lu62ARMSuffix (ImqString & přípona);
Poskytuje kopii přípony LU62 ARM. V případě úspěchu vrátí hodnotu TRUE.

ImqString lu62ARMSuffix ( );
Vrací příponu LU62 ARM bez jakékoli indikace možných chyb.

ImqBoolean luName (ImqString & name);
Poskytuje kopii jména LU. V případě úspěchu vrátí hodnotu TRUE.

ImqString luName ( );
Vrací jméno LU bez označení možných chyb.

ImqBoolean maximumActiveKanály (kanály MQLONG &);
Poskytuje kopii hodnoty maximálního počtu aktivních kanálů. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumActiveKanály ();
Vrací hodnotu maximálního počtu aktivních kanálů bez jakékoli indikace možných chyb.

ImqBoolean maximumCurrentKanály (kanály MQLONG &);
Poskytuje kopii hodnoty maximálního počtu aktuálních kanálů. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumCurrentKanály ();
Vrací maximální hodnotu aktuálního kanálu bez jakékoli indikace možných chyb.

ImqBoolean maximumHandles(MQLONG & ***číslo*** **);**
Poskytuje kopii maximálního počtu popisovačů. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumHandles();
Vrátí maximální počet popisovačů bez jakékoli indikace možných chyb.

ImqBoolean maximumLu62Channels (kanály MQLONG &);
Poskytuje kopii maximální hodnoty LU62 kanálů. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumLu62Channels ();.
Vrací maximální hodnotu LU62 kanálů bez jakékoli indikace možných chyb.

ImqBoolean maximumMessageDélka (MQLONG & ***délka*** **);**
Poskytuje kopii maximální délky zprávy. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumMessageLength ();
Vrací maximální délku zprávy bez jakékoli indikace možných chyb.

ImqBoolean maximumPriority(priorita MQLONG & ***priority*** **);**
Poskytuje kopii maximální priority. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumPriority();
Vrátí kopii maximální priority bez jakékoli indikace možných chyb.

ImqBoolean maximumTcpKanály (MQLONG & kanály);
Poskytuje kopii maximální hodnoty kanálů TCP. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumTcpKanály ();
Vrací maximální hodnotu kanálů TCP bez jakékoli indikace možných chyb.

ImqBoolean maximumUncommittedzprávy (MQLONG & ***číslo*** **);**
Poskytuje kopii maximálního počtu nepotvrzených zpráv. V případě úspěchu vrátí hodnotu TRUE.

MQLONG maximumUncommittedMessages ();
Vrací maximální počet nepotvrzených zpráv bez jakékoli indikace možných chyb.

ImqBoolean mqiAccounting (MQLONG & statint);
Poskytuje kopii účetní hodnoty MQI. V případě úspěchu vrátí hodnotu TRUE.

MQLONG mqiAccounting ();
Vrátí hodnotu evidence rozhraní MQI bez označení možných chyb.

ImqBoolean mqiStatistics (MQLONG & statmqi);
Poskytuje kopii statistické hodnoty MQI. V případě úspěchu vrátí hodnotu TRUE.

MQLONG mqiStatistics ();
Vrací hodnotu statistiky rozhraní MQI bez indikace možných chyb.

ImqBoolean outboundPortMax (MQLONG & max);
Poskytuje kopii maximální hodnoty odchozího portu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG outboundPortMax ();
Vrací maximální hodnotu odchozího portu bez jakékoli indikace možných chyb.

ImqBoolean outboundPortMin (MQLONG & min);
Poskytuje kopii minimální hodnoty odchozího portu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG outboundPortMin ();
Vrací minimální hodnotu odchozího portu bez jakékoli indikace možných chyb.

ImqBinary -heslo () const;
Vrací heslo použité pro připojení klienta.

ImqBoolean setPassword (const ImqString & password);
Nastaví heslo použité pro připojení klienta.

ImqBoolean setPassword (const char * = 0 heslo);
Nastaví heslo použité pro připojení klienta.

ImqBoolean setPassword (const ImqBinary & password);
Nastaví heslo použité pro připojení klienta.

ImqBoolean performanceEvent(událost MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení události výkonu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG performanceEvent();
Vrací stav povolení události výkonu bez označení možných chyb.

ImqBoolean platforma (MQLONG & ***platforma*** **);**
Poskytuje kopii platformy. V případě úspěchu vrátí hodnotu TRUE.

platforma MQLONG ();
Vrátí platformu bez jakékoli indikace možných chyb.

ImqBoolean queueAccounting (MQLONG & acctq);
Poskytuje kopii hodnoty evidence front. V případě úspěchu vrátí hodnotu TRUE.

MQLONG queueAccounting ();
Vrátí hodnotu evidence front bez jakéhokoli označení možných chyb.

ImqBoolean queueMonitoring (MQLONG & monq);
Poskytuje kopii hodnoty monitorování fronty. V případě úspěchu vrátí hodnotu TRUE.

MQLONG queueMonitoring ();
Vrací hodnotu monitorování fronty bez jakékoli indikace možných chyb.

ImqBoolean queueStatistics (MQLONG & statq);
Poskytuje kopii hodnoty statistiky fronty. V případě úspěchu vrátí hodnotu TRUE.

MQLONG queueStatistics ();
Vrátí hodnotu statistiky fronty bez jakékoli indikace možných chyb.

ImqBoolean receiveTimeout (MQLONG & timeout);
Poskytuje kopii hodnoty časového limitu příjmu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG receiveTimeout ();
Vrací hodnotu časového limitu příjmu bez jakékoli indikace možných chyb.

ImqBoolean receiveTimeoutMin (MQLONG & min);
Poskytuje kopii minimální hodnoty časového limitu příjmu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG receiveTimeoutMin ();
Vrací minimální hodnotu časového limitu příjmu bez jakékoli indikace možných chyb.

ImqBoolean receiveTimeoutTyp (MQLONG & typ);
Poskytuje kopii typu časového limitu příjmu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG receiveTimeoutTyp ();
Vrací typ časového limitu příjmu bez jakékoli indikace možných chyb.

ImqBoolean remoteEvent(MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení vzdálené události. V případě úspěchu vrátí hodnotu TRUE.

MQLONG remoteEvent();
Vrací stav povolení vzdálené události bez označení možných chyb.

ImqBoolean repositoryName(ImqString & ***název*** **);**
Poskytuje kopii názvu úložiště. V případě úspěchu vrátí hodnotu TRUE.

ImqString repositoryName( );
Vrací název úložiště bez jakéhokoli označení možných chyb.

ImqBoolean repositoryNamelistNázev (ImqString & ***name*** **);**
Poskytuje kopii názvu seznamu názvů úložiště. V případě úspěchu vrátí hodnotu TRUE.

ImqString repositoryNamelistName ();
Vrací kopii názvu seznamu názvů úložiště bez jakéhokoli označení možných chyb.

ImqBoolean sharedQueueQueueManager(MQLONG & name);
Poskytuje kopii hodnoty názvu sdíleného správce front. V případě úspěchu vrátí hodnotu TRUE.

MQLONG sharedQueueQueueManagerName ();
Vrací hodnotu názvu sdíleného správce front bez jakékoli indikace možných chyb.

ImqBoolean sslEvent (MQLONG & event);
Poskytuje kopii hodnoty události SSL. V případě úspěchu vrátí hodnotu TRUE.

MQLONG sslEvent ();
Vrací hodnotu události SSL bez jakékoli indikace možných chyb.

ImqBoolean sslFips (MQLONG & sslfips);
Poskytuje kopii hodnoty SSL FIPS. V případě úspěchu vrátí hodnotu TRUE.

MQLONG sslFips ();
Vrací hodnotu SSL FIPS bez označení možných chyb.

ImqBoolean sslKeyResetCount (MQLONG & count);
Poskytuje kopii hodnoty počtu resetů klíče SSL. V případě úspěchu vrátí hodnotu TRUE.

MQLONG sslKeyResetCount ();
Vrací hodnotu počtu resetů klíče SSL bez označení možných chyb.

ImqBoolean startStop(událost MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení události start-stop. V případě úspěchu vrátí hodnotu TRUE.

MQLONG startStopUdálost ();
Vrací stav povolení události start-stop bez jakéhokoli označení možných chyb.

ImqBoolean statisticsInterval (MQLONG & statint);
Poskytuje kopii hodnoty intervalu statistiky. V případě úspěchu vrátí hodnotu TRUE.

MQLONG statisticsInterval ();
Vrací hodnotu intervalu statistiky bez jakékoli indikace možných chyb.

ImqBoolean syncPointAvailability (MQLONG & ***sync*** **);**
Poskytuje kopii hodnoty dostupnosti synchronizačního bodu. V případě úspěchu vrátí hodnotu TRUE.

MQLONG syncPointAvailability ();
Vrací kopii hodnoty dostupnosti synchronizačního bodu bez jakékoli indikace možných chyb.

ImqBoolean tcpName (ImqString & name);
Poskytuje kopii názvu systému TCP. V případě úspěchu vrátí hodnotu TRUE.

ImqString tcpName ( );
Vrací název systému TCP bez označení možných chyb.

ImqBoolean tcpStack(typ MQLONG &);
Poskytuje kopii typu zásobníku TCP. V případě úspěchu vrátí hodnotu TRUE.

MQLONG tcpStackType ();
Vrací typ zásobníku TCP bez označení možných chyb.

ImqBoolean traceRouteZáznam (MQLONG & routerec);
Poskytuje kopii hodnoty záznamu trasovací trasy. V případě úspěchu vrátí hodnotu TRUE.

MQLONG traceRouteRecording ();
Vrací hodnotu záznamu trasovací trasy bez jakékoli indikace možných chyb.

ImqBoolean triggerInterval(MQLONG & ***interval*** **);**
Poskytuje kopii intervalu spouštěče. V případě úspěchu vrátí hodnotu TRUE.

MQLONG triggerInterval();
Vrátí interval spouštěče bez jakékoli indikace možných chyb.

ImqBinary userId () const;
Vrací ID uživatele použité pro připojení klienta.

ImqBoolean setUserId (const ImqString & id);
Nastaví ID uživatele použité pro připojení klienta.

ImqBoolean setUserId (const char * = 0 id);
Nastaví ID uživatele použité pro připojení klienta.

ImqBoolean setUserId (const ImqBinary & id);
Nastaví ID uživatele použité pro připojení klienta.

Metody objektů (chráněné)

void setFirstManagedObject (const ImqObject * ***object*** **= 0);**
Nastaví první spravovaný objekt.

Data objektu (chráněná)

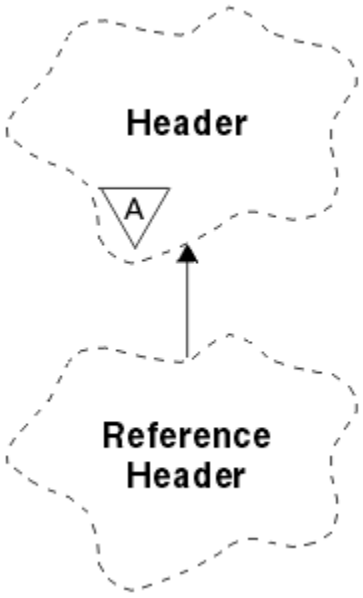
MQHCONN ***ohconn***
Manipulátor připojení IBM MQ (smysluplný pouze v případě, že je stav připojení TRUE).

Kódy příčin

- MQRC_ATTRIBUTE_LOCKED
- MQRC_ENVIRONMENT_ERROR
- MQRC_FUNCTION_NOT_SUPPORTED
- MQRC_REFERENCE_ERROR
- (kódy příčiny pro MQBACK)
- (kódy příčiny pro MQBEGIN)
- (kódy příčiny pro MQCMIT)
- (kódy příčiny pro MQCONNX)
- (kódy příčiny pro MQDISC)
- (kódy příčiny pro MQCONN)

Třída ImqReferenceHeader C++

Tato třída zapouzdřuje funkce datové struktury MQRMH.



Obrázek 34. Třída záhlaví ImqReference

Tato třída souvisí s voláními MQI uvedenými v seznamu “Křížový odkaz záhlaví ImqReference” na stránce 1794.

Atributy objektů

cílové prostředí
Prostředí pro cíl. Počáteční hodnota je řetězec s hodnotou Null.

Název místa určení
Název místa určení dat. Počáteční hodnota je řetězec s hodnotou Null.

ID instance
Identifikátor instance. Binární hodnota (MQBYTE24) délky MQ_OBJECT_INSTANCE_ID_LENGTH. Počáteční hodnota je MQOII_NONE.

logická délka
Logická nebo zamýšlená délka dat zprávy, která následují za tímto záhlavím. Počáteční hodnota je nula.

logické posunutí
Logická odchylka pro následující data zprávy, která mají být interpretována v kontextu dat jako celek, v konečném místě určení. Počáteční hodnota je nula.

logický posun 2
Rozšíření vysokého pořadí k logickému posunu. Počáteční hodnota je nula.

Typ odkazu
Typ odkazu. Počáteční hodnota je řetězec s hodnotou Null.

Zdrojové prostředí
Prostředí pro zdroj. Počáteční hodnota je řetězec s hodnotou Null.

Zdrojový název
Název zdroje dat. Počáteční hodnota je řetězec s hodnotou Null.

Konstruktory

ImqReferenceHeader ();
Výchozí konstruktor.

ImqReferenceHeader (const ImqReferenceHeader & *header*);
Konstruktor kopírování.

Přetížené metody ImqItem

virtuální ImqBoolean copyOut (ImqMessage & *msg*);
Vloží datovou strukturu MQRMH do vyrovnávací paměti zpráv na začátku, přesune existující data zprávy dále a nastaví formát *msg* na MQFMT_REF_MSG_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader na webu "Třída ImqHeader C++" na stránce 1822 .

virtuální ImqBoolean pasteIn (ImqMessage & *msg*);
Načte datovou strukturu MQRMH z vyrovnávací paměti zpráv.

Chcete-li být úspěšní, formát ImqMessage musí být MQFMT_REF_MSG_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader na webu "Třída ImqHeader C++" na stránce 1822 .

Metody objektů (veřejné)

void operator = (const ImqReferenceZáhlaví & *záhlaví*);
Zkopíruje data instance ze záhlaví *header*a nahradí existující data instance.

ImqString destinationEnvironment () const;
Vrací kopii cílového prostředí.

void setDestinationEnvironment (const char * *environment* = 0);
Nastaví cílové prostředí.

ImqString destinationName () const;
Vrátí kopii názvu místa určení.

void setDestinationName (const char * *name* = 0);
Nastaví název místa určení.

ImqBinary instanceId () const;
Vrací kopii ID instance.

ImqBoolean setInstanceId (const ImqBinary & *id*);
Nastaví ID instance. Délka dat *tokenu* musí být buď 0, nebo MQ_OBJECT_INSTANCE_ID_LENGTH. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

void setInstanceId (const MQBYTE24 *id* = 0);
Nastaví ID instance. *id* může být nula, což je stejné jako určení MQOII_NONE. Je-li *id* nenulové, musí adresovat MQ_OBJECT_INSTANCE_ID_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, například MQOII_NONE, může být nutné provést přetypování, aby byla zajištěna shoda podpisu, například (MQBYTE *) MQOII_NONE.

MQLONG logicalLength () const;
Vrátí logickou délku.

void setLogicalLength (const MQLONG *length*);
Nastaví logickou délku.

MQLONG logicalOffset () const;
Vrátí logický posun.

void setLogicalOffset (const MQLONG *offset*);
Nastaví logický posun.

MQLONG logicalOffset2 () const;
Vrátí logický posun 2.

void setLogicalOffset2 (const MQLONG *offset*);
Nastaví logický posun 2.

ImqString referenceType () const;
Vrátí kopii typu odkazu.

void setReferenceType (const char * *name* = 0);
Nastaví typ odkazu.

ImqString sourceEnvironment () const;
Vrátí kopii zdrojového prostředí.

void setSourceEnvironment (const char * *environment* = 0);
Nastaví zdrojové prostředí.

ImqString sourceName () const;
Vrátí kopii názvu zdroje.

void setSourceName (const char * *name* = 0);
Nastaví název zdroje.

Data objektu (chráněná)

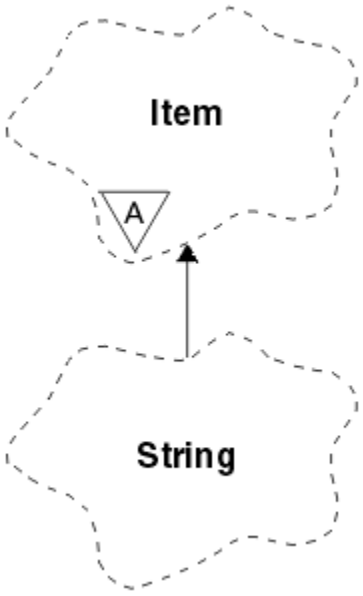
MQRMH *omqrmh*
Datová struktura MQRMH.

Kódy příčin

- MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_STRUC_LENGTH_ERROR
- MQRC_STRUC_ID_ERROR
- MQRC_INDOSTATEČNÁ_DATA
- MQRC_NEKONZISTENTNÍ_FORMÁT
- MQRC_ENCODING_ERROR

ImqString Třída C + +

Tato třída poskytuje úložiště znakových řetězců a manipulaci pro řetězce ukončené znakem null.



Obrázek 35. Třída ImqString

Použijte ImqString místo **char *** ve většině situací, kdy parametr volá **char ***.

- "Atributy objektů" na stránce 1878
- "Konstruktory" na stránce 1878
- "Metody třídy (veřejné)" na stránce 1879
- "Přetížené metody ImqItem" na stránce 1879
- "Metody objektů (veřejné)" na stránce 1879
- "Metody objektů (chráněné)" na stránce 1882
- "Kódy příčin" na stránce 1882

Atributy objektů

znaků
Znaky v **úložišti** , které předcházejí koncové hodnotě null.

délka
Počet bajtů ve **znacích**. Pokud není **úložiště**, **délka** je nula. Počáteční hodnota je nula.

úložný prostor
Dočasné pole bajtů libovolné velikosti. Koncová hodnota null musí být vždy přítomna v **úložišti** po **znacích**, aby mohl být zjištěn konec **znaků** . Metody zajišťují, že tato situace je zachována, ale při přímém nastavení bajtů v poli se ujistěte, že po úpravě existuje koncová hodnota null. Na počátku neexistuje žádný atribut **storage** .

Konstruktory

ImqString( );
Výchozí konstruktor.

ImqString(const ImqString & *řetězec*);
Konstruktor kopírování.

ImqString(const char *c*);
znaky tvoří *c*.

ImqString(znak const * *text*);
znaky se zkopírují z *textu*.

ImqString(const void * *buffer*, const size_t *length*);
Zkopíruje *délku* bajtů počínaje *vyrovnávací pamětí* a přiřadí je **znakům**. Náhrada se provede pro všechny kopírované znaky s hodnotou null. Substituční znak je tečka (.). Žádný zvláštní ohled není dán na žádné jiné netisknutelné nebo nezobrazitelné kopírované znaky.

Metody třídy (veřejné)

static ImqBoolean copy (char * *destination-buffer*, const size_t *length*, const char * *source-buffer*, const char *pad* = 0);
Zkopíruje až *délku* bajtů z *zdrojové vyrovnávací paměti* do *cílové vyrovnávací paměti*. Pokud je počet znaků ve *zdrojové vyrovnávací paměti* nedostatečný, vyplní zbývající prostor v poli *destination-buffer* znaky *pad* . Hodnota *source-buffer* může být nula. Hodnota *destination-buffer* může být nula, pokud *length* je také nula. Všechny kódy chyb jsou ztraceny. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

static ImqBoolean copy (char * *destination-buffer*, const size_t *length*, const char * *source-buffer*, ImqError & *error-object*, const char *pad* = 0);
Zkopíruje až *délku* bajtů z *zdrojové vyrovnávací paměti* do *cílové vyrovnávací paměti*. Pokud je počet znaků ve *zdrojové vyrovnávací paměti* nedostatečný, vyplní zbývající prostor v poli *destination-buffer* znaky *pad* . Hodnota *source-buffer* může být nula. Hodnota *destination-buffer* může být nula, pokud *length* je také nula. Všechny kódy chyb jsou nastaveny v poli *error-object*. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

Přetížené metody ImqItem

virtuální ImqBoolean copyOut (ImqMessage & *msg*);
Zkopíruje **znaky** do vyrovnávací paměti zpráv a nahradí veškerý existující obsah. Nastaví *zpráva* **formát** na MQFMT_STRING.

Další podrobnosti viz popis metody nadřízené třídy.

virtuální ImqBoolean pasteIn (ImqMessage & *msg*);
Nastaví **znaky** přenosem zbývajících dat z vyrovnávací paměti zpráv a nahradí existující **znaky**.

Chcete-li být úspěšní, **kódování** objektu *msg* musí být MQENC_NATIVE. Načíst zprávy s parametrem MQGMO_CONVERT na MQENC_NATIVE.

Aby byl produkt ImqMessage **formát** úspěšný, musí být MQFMT_STRING.

Další podrobnosti viz popis metody nadřízené třídy.

Metody objektů (veřejné)

Znak & operátor [] (const size_t *offset*) const;
Odkazuje na znak na offsetu *offset* v **úložišti**. Ujistěte se, že příslušný bajt existuje a je adresovatelný.

operátor ImqString () (const size_t *offset*, const size_t *délka* = 1) const;
Vrací podřetězec zkopírováním bajtů ze **znaků** začínajících na *offset*. Je-li *length* nula, vrátí zbytek **znaků**. Pokud kombinace *offset* a *length* nevytvoří odkaz v rámci **znaků**, vrátí prázdný ImqString.

void operator = (const ImqString & *řetězec*);
Zkopíruje data instance z *řetězce*a nahradí existující data instance.

ImqString operátor + (const char *c*) const;
Vrací výsledek připojení *c* k **znakům**.

ImqString operátor + (const char * *text*) const;
Vrací výsledek připojení *textu* k **znakům**. To může být také obrácené. Příklad:

```
strOne + "string two" ;
"string one" + strTwo ;
```

Poznámka: Ačkoli většina kompilátorů přijímá řetězec **strOne + "string two";** Microsoft Visual C++ vyžaduje řetězec **strOne + (char *) "string two";**

ImqString operátor + (const ImqString & *string1*) const;
Vrací výsledek připojení *string1* ke **znakům**.

ImqString operátor + (const double *číslo*) const;
Vrací výsledek připojení *number* k **znakům** po převodu na text.

ImqString operátor + (const long *číslo*) const;
Vrací výsledek připojení *number* k **znakům** po převodu na text.

void operator + = (const char *c*);
Připojí řetězec *c* k **znakům**.

void operátor + = (const char * *text*);
Připojí *text* k **znakům**.

void operator + = (const ImqString & *řetězec*);
Připojí *řetězec* k **znakům**.

void operátor + = (const double *number*);
Připojí *číslo* k **znakům** po převodu na text.

void operátor + = (const long *number*);
Připojí *číslo* k **znakům** po převodu na text.

znak operátoru * () const;
Vrátí adresu prvního bajtu v **úložišti**. Tato hodnota může být nula a je nestálá. Tuto metodu používejte pouze pro účely jen pro čtení.

ImqBoolean operátor < (const ImqString & *řetězec*) const;
Porovná **znaky** s znaky *řetězce* pomocí metody **compare** . Výsledek je TRUE, pokud je menší než, a FALSE, pokud je větší nebo rovno.

ImqBoolean operátor > (const ImqString & *řetězec*) const;
Porovná **znaky** s znaky *řetězce* pomocí metody **compare** . Výsledek je TRUE, pokud je větší než, a FALSE, pokud je menší nebo rovno.

ImqBoolean operátor < = (const ImqString & *řetězec*) const;
Porovná **znaky** s znaky *řetězce* pomocí metody **compare** . Výsledek je TRUE, pokud je menší nebo rovno, a FALSE, pokud je větší než.

ImqBoolean operátor > = (const ImqString & *řetězec*) const;
Porovná **znaky** s znaky *řetězce* pomocí metody **compare** . Výsledek je TRUE, pokud je větší nebo rovno, a FALSE, pokud je menší než.

ImqBoolean operátor == (const ImqString & *řetězec*) const;
Porovná **znaky** s znaky *řetězce* pomocí metody **compare** . Vrací hodnotu TRUE nebo FALSE.

ImqBoolean operátor! = (const ImqString & *řetězec*) const;
Porovná **znaky** s znaky *řetězce* pomocí metody **compare** . Vrací hodnotu TRUE nebo FALSE.

short compare (const ImqString & *řetězec*) const;
Porovná **znaky** s znaky *řetězce*. Výsledek je nula, pokud se **znaky** rovnají, záporný, pokud je menší než, a kladný, pokud je větší než. V porovnání se rozlišují malá a velká písmena. Hodnota ImqString s hodnotou Null je považována za menší než hodnota ImqStrings hodnotou jinou než Null.

ImqBoolean copyOut(char * *buffer*, const size_t *length*, const char *pad* = 0);
Zkopíruje až *délku* bajtů z **znaků** do *vyrovnávací paměti*. Pokud je počet **znaků** nedostatečný, vyplní zbývající prostor ve *vyrovnávací paměti* znaky *pad* . Hodnota *buffer* může být nula, pokud *length* je také nula. V případě úspěchu vrátí hodnotu TRUE.

size_t copyOut(dlouhé & *číslo*) const;
Nastaví *číslo* z **znaků** po převodu z textu a vrátí počet znaků zahrnutých do převodu. Pokud je tato hodnota nula, nebyl proveden žádný převod a *number* není nastaveno. Konvertibilní posloupnost znaků musí začínat následujícími hodnotami:

```
<blank(s)>
<+|->
digit(s)
```

size_t copyOut(ImqString & *token*, const char *c* = '') const;

Pokud **znaky** obsahují jeden nebo více znaků, které se liší od znaků *c*, identifikuje token jako první souvislou posloupnost těchto znaků. V tomto případě je *token* nastaven na tuto posloupnost a vrácená hodnota je součtem počtu úvodních znaků *c* a počtu bajtů v posloupnosti. Jinak vrací nulu a nenastavuje *token*.

size_t cutOut(dlouhé & *číslo*);

Nastaví *číslo* jako pro metodu **copy** , ale také odebere z **znaků** počet bajtů označených návratovou hodnotou. Například řetězec zobrazený v následujícím příkladu lze snížit na tři čísla pomocí **cutOut** (*number*) třikrát:

```
strNumbers = "-1 0   +55 "

while ( strNumbers.cutOut( number ) );
number becomes -1, then 0, then 55
leaving strNumbers == " "
```

size_t cutOut(ImqString & *token*, const char *c* = " '

Nastaví *token* jako pro metodu **copyOut** a odebere z **znaky** znaků *strToken* a také všechny znaky *c* , které předcházejí znakům *token* . Pokud *c* není mezera, odebere znaky *c* , které přímo nahradí znaky *token* . Vrátí počet odebraných znaků. Například řetězec zobrazený v následujícím příkladu lze vyjmout do tří tokenů pomocí **cutOut** (*token*). třikrát:

```
strText = " Program Version 1.1 "

while ( strText.cutOut( token ) );

// token becomes "Program", then "Version",
// then "1.1" leaving strText == " "
```

Následující příklad uvádí, jak analyzovat název cesty 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 & *řetězec*);

Vyhledá přesnou shodu pro *řetězec* kdekoli v rámci **znaků**. Pokud není nalezena žádná shoda, vrátí hodnotu FALSE. Jinak vrátí hodnotu TRUE. Má-li *řetězec* hodnotu null, vrátí hodnotu TRUE.

ImqBoolean find (const ImqString & *řetězec*, velikost_t & *offset*);

Vyhledá přesnou shodu pro *řetězec* někde uvnitř **znaků** od offsetu *offset* a dále. Má-li *řetězec* hodnotu null, vrátí hodnotu TRUE bez aktualizace *offsetu*. Pokud není nalezena žádná shoda, vrátí hodnotu FALSE (hodnota *offset* mohla být zvýšena). Je-li nalezena shoda, vrátí hodnotu TRUE a aktualizuje *offset* na offset *řetězce* v rámci **znaků**.

size_t délka () const;

Vrací **délku**.

ImqBoolean pasteIn(const double *number*, const char * *formát* = "%f");

Připojí *číslo* k **znakům** po převodu na text. V případě úspěchu vrátí hodnotu TRUE.

Specifikace *format* se používá k formátování převodu s pohyblivou řádovou čárkou. Je-li uveden, musí být vhodný pro použití s **printf** a čísly s pohyblivou řádovou čárkou, například **%.3f**.

ImqBoolean pasteIn(const long *number*);

Připojí *číslo* k **znakům** po převodu na text. V případě úspěchu vrátí hodnotu TRUE.

ImqBoolean pasteIn(const void * *buffer*, const size_t *length*);
Připojí *délku* bajtů z *vyrovnávací paměti* k **znakům**a přidá konečnou koncovou hodnotu null. Nahradí všechny zkopírované znaky null. Substituční znak je tečka (.). Žádný zvláštní ohled se nedává na žádné jiné netisknutelné nebo nezobrazitelné kopírované znaky. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean set (const char * *buffer*, const size_t *length*);
Nastaví **znaky** ze znakového pole s pevnou délkou, které může obsahovat hodnotu null. V případě potřeby připojí k znakům z pole s pevnou délkou hodnotu null. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqBoolean setStorage(const size_t *length*);
Přiděluje (nebo znovu přiděluje) **úložiště**. Zachová všechny původní **znaky**, včetně všech koncových hodnot null, pokud pro ně stále existuje místo, ale neinicializuje žádné další úložiště.

Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

size_t úložiště () const;
Vrací počet bajtů v **úložišti**.

size_t stripLeading(const char *c* = '');
Odstraní úvodní znaky *c* z **znaků** a vrátí odebrané číslo.

size_t stripTrailing(const char *c* = '');
Odstraní koncové znaky *c* z **znaků** a vrátí odebrané číslo.

ImqString upperCase() const;
Vrací kopii **znaků**s velkými písmeny.

Metody objektů (chráněné)

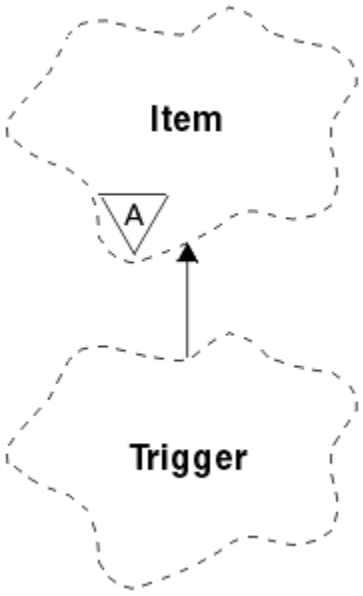
ImqBoolean přiřazení (const ImqString & *řetězec*);
Ekvivalentní ekvivalentní metodě **operator =** , avšak nevirtuální. V případě úspěchu vrátí hodnotu TRUE.

Kódy příčin

- MQRC_DATA_TRUNCATED
- Ukazatel MQRC_NULL_POINTER
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_BUFFER_ERROR
- MQRC_NEKONZISTENTNÍ_FORMÁT

Třída ImqTrigger C++

Tato třída zapouzdřuje datovou strukturu MQTM (trigger message).



Obrázek 36. Třída ImqTrigger

Objekty této třídy jsou obvykle používány programem monitoru spouštěčů. Úlohou programu monitoru spouštěčů je čekat na tyto konkrétní zprávy a pracovat s nimi, aby se zajistilo, že ostatní aplikace IBM MQ budou spuštěny, když na ně zprávy čekají.

Příklad použití viz ukázkový program IMQSTRG.

- "Atributy objektů" na stránce 1883
- "Konstruktory" na stránce 1884
- "Přetížené metody ImqItem" na stránce 1884
- "Metody objektů (veřejné)" na stránce 1884
- "Data objektu (chráněná)" na stránce 1885
- "Kódy příčin" na stránce 1885

Atributy objektů

ID aplikace
Identita aplikace, která odeslala zprávu. Počáteční hodnota je řetězec s hodnotou Null.

Typ aplikace
Typ aplikace, která odeslala zprávu. Počáteční hodnota je nula. Jsou možné následující další hodnoty:

- MQAT_AIX-operační systém
- 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

Data prostředí
Data prostředí pro proces. Počáteční hodnota je řetězec s hodnotou Null.

Název procesu
Název procesu. Počáteční hodnota je řetězec s hodnotou Null.

Název fronty
Název fronty, která má být spuštěna. Počáteční hodnota je řetězec s hodnotou Null.

Data spouątěče
Data spouštěče pro proces. Počáteční hodnota je řetězec s hodnotou Null.

Data uživatele
Uživatelská data pro proces. Počáteční hodnota je řetězec s hodnotou Null.

Konstruktory

ImqTrigger( );
Výchozí konstruktor.

ImqTrigger(const ImqTrigger & *trigger*);
Konstruktor kopírování.

Přetížené metody ImqItem

virtuální ImqBoolean copyOut (ImqMessage & *msg*);
Zapíše datovou strukturu MQTM do vyrovnávací paměti zpráv a nahradí veškerý existující obsah. Nastaví formát *msg* na hodnotu MQFMT_TRIGGER.

Další podrobnosti naleznete v popisu metody třídy ImqItem na adrese “Třída ImqItem C++” na stránce 1826 .

virtuální ImqBoolean pasteIn (ImqMessage & *msg*);
Čte datovou strukturu MQTM z vyrovnávací paměti zpráv.

Chcete-li být úspěšní, formát ImqMessage musí být MQFMT_TRIGGER.

Další podrobnosti naleznete v popisu metody třídy ImqItem na adrese “Třída ImqItem C++” na stránce 1826 .

Metody objektů (veřejné)

void operator = (const ImqTrigger & *trigger*);
Zkopíruje data instance z *spouštěče*a nahradí existující data instance.

ImqString applicationId () const;
Vrátí kopii ID aplikace.

void setApplicationId (const char * *id*);
Nastaví ID aplikace.

MQLONG applicationType () const;
Vrátí typ aplikace.

void setApplicationType (const MQLONG *type*);
Nastaví typ aplikace.

ImqBoolean copyOut (MQTMC2 * *ptmc2*);
Zapouzdřuje datovou strukturu MQTM, což je struktura přijatá v inicializačních frontách. Vyplní ekvivalentní datovou strukturu MQTMC2 poskytnutou volajícím a nastaví pole QMgrName (které není přítomno v datové struktuře MQTM) na všechny mezery. Datová struktura MQTMC2 se tradičně používá jako parametr pro aplikace spuštěné monitorem spouštěčů. Tato metoda vrátí hodnotu TRUE, pokud je úspěšná.

ImqString environmentData () const;
Vrátí kopii dat prostředí.

void setEnvironmentData (const char * *data*);
Nastaví data prostředí.

ImqString processName () const;
Vrátí kopii názvu procesu.

void setProcessName (const char * *name*);
Nastaví název procesu vyplněný mezerami na 48 znaků.

ImqString queueName () const;
Vrátí kopii názvu fronty.

void setQueueName (const char * *name*);
Nastaví název fronty s mezerami na 48 znaků.

ImqString triggerData () const;
Vrátí kopii dat spouštěče.

void setTriggerData (const char * *data*);
Nastaví data spouštěče.

ImqString userData () const;
Vrátí kopii uživatelských dat.

void setUserData (const char * *data*);
Nastaví uživatelská data.

Data objektu (chráněná)

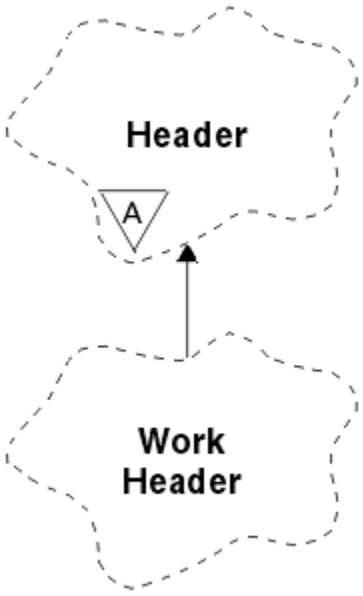
MQTM *omqtm*
Datová struktura MQTM.

Kódy příčin

- Ukazatel MQRC_NULL_POINTER
- MQRC_NEKONZISTENTNÍ_FORMÁT
- MQRC_ENCODING_ERROR
- MQRC_STRUC_ID_ERROR

Třída ImqWorkHeader C++

Tato třída zapouzdřuje specifické funkce datové struktury MQWIH.



Obrázek 37. Třída záhlaví ImqWork

Objekty této třídy jsou používány aplikacemi, které vkládají zprávy do fronty spravované správcem zátěže z/OS .

- "Atributy objektů" na stránce 1886
- "Konstruktory" na stránce 1886
- "Přetížené metody ImqItem" na stránce 1886
- "Metody objektů (veřejné)" na stránce 1887
- "Data objektu (chráněná)" na stránce 1887
- "Kódy příčin" na stránce 1887

Atributy objektů

token zprávy
Token zprávy pro z/OS Workload Manager, o délce MQ_MSG_TOKEN_LENGTH. Počáteční hodnota je MQMTOK_NONE.

Název služby
Název procesu o délce 32 znaků. Název je zpočátku prázdný.

servisní krok
Osmiznakový název kroku v rámci procesu. Název je zpočátku prázdný.

Konstruktory

ImqWork-záhlaví ();
Výchozí konstruktor.

ImqWorkHeader (const ImqWorkHeader & ***header*** **);**
Konstruktor kopírování.

Přetížené metody ImqItem

virtuální ImqBoolean copyOut(ImqMessage & ***msg*** **);**
Vloží datovou strukturu MQWIH na začátek vyrovnávací paměti zpráv, přesune existující data zprávy dále a nastaví *zpráva* **formát** na hodnotu MQFMT_WORK_INFO_HEADER.

Další podrobnosti viz popis metody nadřízené třídy.

virtuální ImqBoolean pasteIn(ImqMessage & *msg*);
Čte datovou strukturu MQWIH z vyrovnávací paměti zpráv.

Aby bylo kódování objektu *msg* úspěšné, musí být MQENC_NATIVE. Načíst zprávy s parametrem MQGMO_CONVERT na MQENC_NATIVE.

Formát ImqMessage musí být MQFMT_WORK_INFO_HEADER.

Další podrobnosti viz popis metody nadřízené třídy.

Metody objektů (veřejné)

void operator = (const ImqWorkZáhlaví & *záhlaví*);
Zkopíruje data instance ze záhlaví *header*a nahradí existující data instance.

ImqBinary messageToken () const;
Vrací **token zprávy**.

ImqBoolean setMessage(const ImqBinary & *token*);
Nastaví **token zprávy**. Délka dat *tokenu* musí být buď nula, nebo MQ_MSG_TOKEN_LENGTH.
V případě úspěchu vrátí hodnotu TRUE.

void setMessageToken (const MQBYTE16 *token* = 0);
Nastaví **token zprávy**. *token* může být nula, což je stejné jako zadání MQMTOK_NONE. Pokud je *token* nenulový, musí adresovat bajty MQ_MSG_TOKEN_LENGTH binárních dat.

Při použití předdefinovaných hodnot, jako např. MQMTOK_NONE, může být nutné provést přetypování, aby byla zajištěna shoda podpisu; například (MQBYTE *) MQMTOK_NONE.

ImqString serviceName () const;
Vrací **název služby**, včetně koncových mezer.

void setServiceName (const char * *name*);
Nastaví **název služby**.

ImqString serviceStep () const;
Vrátí **servisní krok**včetně koncových mezer.

void setServiceStep (const char * *step*);
Nastaví **servisní krok**.

Data objektu (chráněná)

MQWIH omqwih
Datová struktura MQWIH.

Kódy příčin

- MQRC_BINARY_DATA_LENGTH_ERROR

Vlastnosti objektů IBM MQ classes for JMS

Všechny objekty v adresáři IBM MQ classes for JMS mají vlastnosti. Různé vlastnosti se vztahují na různé typy objektů. Různé vlastnosti mají různé přípustné hodnoty a hodnoty symbolických vlastností se liší mezi nástrojem pro administraci a kódem programu.

Produkt IBM MQ classes for JMS poskytuje prostředky pro nastavení a dotazování vlastností objektů pomocí nástroje pro administraci produktu IBM MQ JMS , IBM MQ Explorer nebo v aplikaci. Mnoho vlastností je relevantní pouze pro specifickou podmnožinu typů objektů.

Informace o použití nástroje pro administraci produktu IBM MQ JMS naleznete v tématu Konfigurace objektů JMS pomocí nástroje pro administraci.

Tabulka 868 na stránce 1888 poskytuje stručný popis každé vlastnosti a ukazuje pro každou vlastnost, na které typy objektů se vztahuje. Typy objektů jsou identifikovány pomocí klíčových slov; vysvětlení těchto objektů naleznete v tématu Konfigurace objektů JMS pomocí administračního nástroje .

Čísla odkazují na poznámky na konci tabulky. Další informace najdete v tématu "Závislosti mezi vlastnostmi objektů IBM MQ classes for JMS" na stránce 1891.

Vlastnost se skládá z dvojice název-hodnota ve formátu:

```
PROPERTY_NAME(property_value)
```

Témata v této sekci uvádějí pro každou vlastnost název vlastnosti a stručný popis a zobrazují platné hodnoty vlastností použité v nástroji pro administraci. a metodu set, která se používá k nastavení hodnoty vlastnosti v aplikaci. Témata také zobrazují platné hodnoty vlastností pro každou vlastnost a mapování mezi symbolickými hodnotami vlastností použitými v nástroji a jejich programovatelnými ekvivalenty.

Názvy vlastností nerozlišují velká a malá písmena a jsou omezeny na sadu rozpoznaných názvů uvedených v těchto tématech.

Tabulka 868. Názvy vlastností a použitelné typy objektů

Vlastnost	**Zkrácená forma**	**Typ objektu**							
		CF	**QCF-počet**	**TCF-zařízení**	**Q**	**T**	**XACF**	**XAQCF**	**XATCF**
"APPLICATIONNAME" na stránce 1893	APPNAME	Y	Y	Y			Y	Y	Y
"Výjimka ASYNCEXCEPTION" na stránce 1894	AEX	Y	Y	Y			Y	Y	Y
"BALOPTIONS" na stránce 1895	Volby	Y	Y	Y			Y	Y	Y
"BALTYPE" na stránce 1895	TYPE	Y	Y	Y			Y	Y	Y
"BALTIMEOUT" na stránce 1896	TIMEOUT	Y	Y	Y			Y	Y	Y
"BROKERCCDURSUBQ" na stránce 1896 [1]	CCDSUB					Y			
"BROKERCCSUBQ" na stránce 1897 [1]	CCSUB	Y		Y			Y		Y
"BROKERCONQ" na stránce 1897 [1]	BCON	Y		Y			Y		Y
"BROKERDURSUBQ" na stránce 1898 [1]	BDSUB					Y			
"BROKERPUBQ" na stránce 1898 [1]	BPUB	Y		Y		Y	Y		Y
"BROKERPUBQMGR" na stránce 1898 [1]	BPQM					Y			
"BROKERQMGR" na stránce 1899 [1]	BQM	Y		Y			Y		Y
"BROKERSUBQ" na stránce 1899 [1]	BSUB	Y		Y			Y		Y
"BROKERVER" na stránce 1900 [1]	BVER	Y [2]		Y [2]		Y	Y		Y
"CCDTURL" na stránce 1900 [3]	CCDT	Y	Y	Y			Y	Y	Y
"CCSID" na stránce 1901	CCS	Y	Y	Y	Y	Y	Y	Y	Y
"CHANNEL" na stránce 1901 [3]	CHAN	Y	Y	Y			Y	Y	Y
"CLEANUP" na stránce 1902 [1]	CL	Y		Y			Y		Y
"CLEANUPINT" na stránce 1902 [1]	CLINT	Y		Y			Y		Y

Tabulka 868. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Zkrácená forma	Typ objektu							
		CF	QCF-počet	TCF-zařízení	Q	T	XACF	XAQCF	XATCF
"ConnectionNameList" na stránce 1903	CNLIST	Y	Y	Y					
"CLIENTRECONNECTOPTIONS" na stránce 1903	CROPT	Y	Y	Y					
"CLIENTRECONNECTTIMEOUT" na stránce 1904	CRT	Y	Y	Y					
"CLIENTID" na stránce 1905	CID	Y [2]	Y	Y [2]			Y	Y	Y
"CLONESUPP" na stránce 1905	CLS	Y		Y			Y		Y
"COMPHDR" na stránce 1905	HC	Y		Y			Y		Y
"COMPMSG" na stránce 1906	MC	Y	Y	Y			Y	Y	Y
"CONNOPT" na stránce 1906	CNOPT	Y	Y	Y			Y	Y	Y
"CONNTAG" na stránce 1907	CNTAG	Y	Y	Y			Y	Y	Y
"DESCRIPTION" na stránce 1908	DESC	Y [2]	Y	Y [2]	Y	Y	Y	Y	Y
"DIRECTAUTH" na stránce 1908	DAUTH	Y [2]		Y [2]					
"ENCODING" na stránce 1909	ENC				Y	Y			
"EXPIRY" na stránce 1910	EXP				Y	Y			
"FAILIFQUIESCE" na stránce 1910	FIQ	Y	Y	Y	Y	Y	Y	Y	Y
"HOSTNAME" na stránce 1911	HOST	Y [2]	Y	Y [2]			Y	Y	Y
"LOCALADDRESS" na stránce 1911	LA	Y [2]	Y	Y [2]			Y	Y	Y
"MAPNAMESTYLE" na stránce 1912	MNST	Y	Y	Y			Y	Y	Y
"MAXBUFFSIZE" na stránce 1913	MBSZ	Y [2]		Y [2]					
"MDREAD" na stránce 1913	MDR				Y	Y			
"MDWRITE" na stránce 1914	MDW				Y	Y			
"MDMSGCTX" na stránce 1914	MDCTX				Y	Y			
"MSGBATCHSZ" na stránce 1915 [1]	MBS	Y	Y	Y			Y	Y	Y
"MSGBODY" na stránce 1915	MBODY				Y	Y			
"MSGRETENTION" na stránce 1916	MRET	Y	Y				Y	Y	
"MSGSELECTION" na stránce 1916 [1]	MSEL	Y		Y			Y		Y
"MULTICAST" na stránce 1917	MCAST	Y [2]		Y [2]		Y			
"OPTIMISTICPUBLICATION" na stránce 1917 [1]	OPTPUB	Y		Y					
"OUTCOMENOTIFICATION" na stránce 1918 [1]	NOTIFY	Y		Y					
"PERSISTENCE" na stránce 1918	PER				Y	Y			

Tabulka 868. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Zkrácená forma	Typ objektu							
		CF	QCF-počet	TCF-zařízení	Q	T	XACF	XAQCF	XATCF
"POLLINGINT" na stránce 1919 [1]	PINT	Y	Y	Y			Y	Y	Y
"PORT" na stránce 1920	PORT	Y [2]	Y	Y [2]			Y	Y	Y
"PRIORITY" na stránce 1920	PRI				Y	Y			
"PROCESSDURATION" na stránce 1921 [1]	PROCDUR	Y		Y					
"PROVIDERVERSION" na stránce 1921	PVER	Y	Y	Y			Y	Y	Y
"PROXYHOSTNAME" na stránce 1923	PHOST	Y [2]		Y [2]					
"PROXYPORT" na stránce 1924	PPORT	Y [2]		Y [2]					
"PUBACKINT" na stránce 1924 [1]	PAI	Y		Y			Y		Y
"PUTASYNCALLOWED" na stránce 1925	PAALD				Y	Y			
"QMANAGER" na stránce 1925	QMGR	Y	Y	Y	Y		Y	Y	Y
"QUEUE" na stránce 1926	QU				Y				
"READAHEADALLOWED" na stránce 1926	RAALD-všechny				Y	Y			
"READAHEADCLOSEPOLICY" na stránce 1927	RACP				Y	Y			
"RECEIVECCSID" na stránce 1927	RCCS				Y	Y			
"RECEIVECONVERSION" na stránce 1928	RCNV				Y	Y			
"RECEIVEISOLATION" na stránce 1928 [1]	RCVISOL	Y		Y					
"RECEXIT" na stránce 1929	RCX	Y	Y	Y			Y	Y	Y
"RECEXITINIT" na stránce 1929	RCXI	Y	Y	Y			Y	Y	Y
"REPLYTOSTYLE" na stránce 1930	RTOST				Y	Y			
"RESCANINT" na stránce 1930 [1]	RINT	Y	Y				Y	Y	
"SECEXIT" na stránce 1931	SCX	Y	Y	Y			Y	Y	Y
"SECEXITINIT" na stránce 1931	SCXI	Y	Y	Y			Y	Y	Y
"SENDCHECKCOUNT" na stránce 1932	SCC	Y	Y	Y			Y	Y	Y
"SENDEXIT" na stránce 1932	SDX	Y	Y	Y			Y	Y	Y
"SENDEXITINIT" na stránce 1933	SDXI	Y	Y	Y			Y	Y	Y
"SHARECONVALLOWED" na stránce 1933	SCALD	Y	Y	Y			Y	Y	Y

Tabulka 868. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Zkrácená forma	Typ objektu							
		CF	QCF-počet	TCF-zařízení	Q	T	XACF	XAQCF	XATCF
"SPARSESUBS" na stránce 1934 [1]	SSUBS	Y		Y					
"SSLCIPHERSUITE" na stránce 1934	SCPHS	Y	Y	Y			Y	Y	Y
"SSLCRL" na stránce 1935	SCRL	Y	Y	Y			Y	Y	Y
"SSLFIPSREQUIRED" na stránce 1935	SFIPS	Y	Y	Y			Y	Y	Y
"SSLPEERNAME" na stránce 1936	SPEER	Y	Y	Y			Y	Y	Y
"SSLRESETCOUNT" na stránce 1936	SRC	Y	Y	Y			Y	Y	Y
"STATREFRESHINT" na stránce 1937 [1]	SRI	Y		Y			Y		Y
"SUBSTORE" na stránce 1937 [1]	SS	Y		Y			Y		Y
"SYNCPOINTALLGETS" na stránce 1938	SPAG	Y	Y	Y			Y	Y	Y
"TARGCLIENT" na stránce 1938	TC				Y	Y			
"TARGCLIENTMATCHING" na stránce 1939	TCM	Y	Y				Y	Y	
"TEMPMODEL" na stránce 1939	TM	Y	Y				Y	Y	
"TEMPQPREFIX" na stránce 1940	TQP	Y	Y				Y	Y	
"TEMPTOPICPREFIX" na stránce 1940	TTP	Y		Y			Y		Y
"TOPIC" na stránce 1941	TOP					Y			
"TRANSPORT" na stránce 1941	TRAN	Y [2]	Y	Y [2]			Y	Y	Y
"WILDCARDFORMAT" na stránce 1942	WCFMT	Y		Y			Y		Y

Poznámka:

1. Tuto vlastnost lze použít s verzí 70 produktu IBM MQ classes for JMS , ale nemá žádný vliv na aplikaci připojenou ke správci front IBM WebSphere MQ 7.0 , pokud není vlastnost PROVIDERVERSION továrny připojení nastavena na číslo verze menší než 7.
2. Při použití připojení v reálném čase ke zprostředkovateli jsou podporovány pouze vlastnosti BROKERVER, CLIENTID, DESCRIPTION, DIRECTAUTH, HOSTNAME, LOCALADDRESS, MAXBUFFSIZE, MULTICAST, PORT, PROXYHOSTNAME, PROXYPORT a TRANSPORT pro objekt továrny ConnectionFactory nebo TopicConnectionFactory.
3. Vlastnosti CCDTURL a CHANNEL objektu nesmí být nastaveny současně.

Závislosti mezi vlastnostmi objektů IBM MQ classes for JMS

Platnost některých vlastností závisí na konkrétních hodnotách jiných vlastností.

Tato závislost se může vyskytnout v následujících skupinách vlastností:

- Vlastnosti klienta
- Vlastnosti pro připojení v reálném čase ke zprostředkovateli
- Ukončit inicializační řetězce

Vlastnosti klienta

Pro připojení ke správci front jsou následující vlastnosti relevantní pouze v případě, že má volba TRANSPORT hodnotu CLIENT:

- HOSTNAME
- PORT
- CHANNEL
- LOCALADDRESS
- CCDTURL
- CCSID
- COMPHDR
- COMPMSG
- RECEXIT
- RECEXITINIT
- SECEXIT
- SECEXITINIT
- SENDEXIT
- SENDEXITINIT
- SHARECONVALLOWED
- SSLCIPHERSUITE
- SSLCRL
- SSLFIPSREQUIRED
- SSLPEERNAME
- SSLRESETCOUNT
- APPLICATIONNAME

Nemůžete nastavit hodnoty pro tyto vlastnosti pomocí nástroje pro administraci, pokud má TRANSPORT hodnotu BIND.

Pokud má TRANSPORT hodnotu CLIENT, výchozí hodnota vlastnosti BROKERVER je V1 a výchozí hodnota vlastnosti PORT je 1414. Pokud nastavíte hodnotu BROKERVER nebo PORT explicitně, pozdější změna hodnoty TRANSPORT nepřepíše vaše volby.

Vlastnosti pro připojení v reálném čase ke zprostředkovateli

Pokud má TRANSPORT hodnotu DIRECT nebo DIRECTHTTP, jsou relevantní pouze následující vlastnosti:

- BROKERVER
- CLIENTID
- DESCRIPTION
- DIRECTAUTH
- HOSTNAME
- LOCALADDRESS
- MAXBUFFSIZE
- MULTICAST (podporováno pouze pro DIRECT)
- PORT

- PROXYHOSTNAME (podporováno pouze pro DIRECT)
- PROXYPORT (podporováno pouze pro režim DIRECT)

Pokud má TRANSPORT hodnotu DIRECT nebo DIRECTHTTP, výchozí hodnota vlastnosti BROKERVER je V2a výchozí hodnota vlastnosti PORT je 1506. Pokud nastavíte hodnotu BROKERVER nebo PORT explicitně, pozdější změna hodnoty TRANSPORT nepřepíše vaše volby.

Ukončit inicializační řetězce

Nenastavujte žádný z inicializačních řetězců uživatelské procedury bez zadání odpovídajícího názvu uživatelské procedury. Vlastnosti inicializace uživatelské procedury jsou:

- RECEXITINIT
- SECEXITINIT
- SENDEXITINIT

Například uvedení `RECEXITINIT(myString)` bez určení `RECEXIT(some.exit.classname)` způsobí chybu.

Související odkazy

"TRANSPORT" na stránce 1941
Povaha připojení ke správci front nebo zprostředkovateli.

APPLICATIONNAME

Aplikace může nastavit název, který identifikuje její připojení ke správci front. Tento název aplikace je zobrazen příkazem **DISPLAY CONN MQSC/PCF** (kde se pole nazývá **APPLTAG**). nebo v zobrazení IBM MQ Průzkumník **Připojení aplikací** (kde se pole nazývá **App name**).

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : APPLICATIONNAME

Krátký název nástroje pro administraci JMS : APPNAME

Programový přístup

Metody setter/getter

- MQConnectionFactory.setAppName ()
- MQConnectionFactory.getAppNázev ()

Hodnoty

Jakýkoli platný řetězec, který není delší než 28 znaků. Delší názvy jsou v případě potřeby upraveny tak, aby se vešly, a to odstraněním úvodních názvů balíků. Pokud je například vyvolávající třída `com.example.MainApp`, použije se úplný název, ale pokud je vyvolávající třída `com.example.dictionaryAndThesaurus.multilingual.mainApp`, použije se název `multilingual.mainApp` , protože se jedná o nejdelší kombinaci názvu třídy a nejpravějšího názvu balíku, který odpovídá dostupné délce.

Je-li samotný název třídy delší než 28 znaků, je oříznut tak, aby se vešel. Například `com.example.mainApplicationForSecondTestCase` se změní na `mainApplicationForSecondTest`.

z/OS V systému z/OSje APPNAME v:

- Režim vazeb je ignorován, pokud je nastaven, a pokud je nastaven, může být nastaven pouze na mezery.
- Režim klienta lze nastavit a použít.

Výjimka ASYNCEXCEPTION

Tato vlastnost určuje, zda má modul IBM MQ classes for JMS informovat modul ExceptionListener pouze v případě přerušení připojení nebo v případě asynchronního výskytu jakékoli výjimky pro volání rozhraní API JMS . To platí pro všechna připojení vytvořená z této ConnectionFactory , která mají registrovaný modul ExceptionListener .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : ASYNCEXCEPTION

Krátký název nástroje pro administraci JMS : AEX

Programový přístup

Metody setter/Getter

- MQConnectionFactory.setAsyncVýjimky ()
- MQConnectionFactory.getAsyncVýjimky ()

Hodnoty

ASYNC_EXCEPTIONS_ALL

Všechny výjimky zjištěné asynchronně mimo rozsah synchronního volání rozhraní API a všechny výjimky s přerušením připojení jsou odeslány do modulu ExceptionListener.

Tabulka 869. Všechny asynchronní výjimky: prostředí a související názvy konstant

Prostředí	Hodnota
JMS nástroj administrace	ALL
Programový	WMQCONSTANTS.ASYNC_EXCEPTIONS_ALL = -1
IBM MQ Explorer	Vše

ASYNC_EXCEPTIONS_CONNECTIONBROKEN

Do modulu ExceptionListenerjsou odesílány pouze výjimky označující přerušené připojení. Jakékoli další výjimky, které se vyskytnou během asynchronního zpracování, nejsou hlášeny modulu ExceptionListener, a proto aplikace není o těchto výjimkách informována. Toto je výchozí hodnota z IBM MQ 8.0.0 Fix Pack 2. Viz JMS: Změny modulu listener pro výjimky v produktu IBM MQ 8.0.

Tabulka 870. Výjimky označující přerušené připojení: prostředí a související názvy konstant

Prostředí	Hodnota
JMS nástroj administrace	PŘERUŠENÉ připojení
Programový	WMQCONSTANTS.ASYNC_EXCEPTIONS_CONNECTIONBROKEN = 1
IBM MQ Explorer	Připojení přerušeno

Je definována následující dodatečná konstanta:

- Z IBM MQ 8.0.0 Fix Pack 2: WMQCONSTANTS.ASYNC_EXCEPTIONS_DEFAULT = ASYNC_EXCEPTIONS_CONNECTIONBROKEN
- Před IBM MQ 8.0.0 Fix Pack 2: WMQCONSTANTS.ASYNC_EXCEPTIONS_DEFAULT = ASYNC_EXCEPTIONS_ALL

Související pojmy

Výjimky v souboru IBM MQ classes for JMS

V 9.3.4 BALOPTIONS

Řídí, jak jsou aplikace IBM MQ classes for JMS , které používají přenos klienta, vyváženy v uniformních klastrech.

Použitelné objekty

`ConnectionFactory`, `QueueConnectionFactory`, `TopicConnectionFactory`, `XAConnectionFactory`, `XAQueueConnectionFactory`, `XATopicConnectionFactory`.

JMS dlouhý název nástroje pro administraci: **BALOPTIONS**

JMS krátký název nástroje pro administraci: **OPTIONS**

Programový přístup

Metody setter/getter

- `MQConnectionFactory.setBalancingOptions()`
- `MQConnectionFactory.getBalancingOptions()`

Hodnoty

IGNNONE
Použije se normální zpracování transakcí a aplikace se nepožadují, aby se během transakce přesunovaly.

Tato hodnota se mapuje na volbu IBM MQ *BalancingOption* `MQBNO_OPTIONS_NONE`.

IGNTRANS
Během transakce může být požadováno, aby se aplikace přesunovaly.

Tato hodnota se mapuje na volbu IBM MQ *BalancingOption* `MQBNO_OPTIONS_IGNORE_TRANS`.

V 9.3.4 BALTYPE

Řídí, jak mohou být aplikace IBM MQ classes for JMS , které používají přenos klienta, vyváženy v jednotném klastru.

Použitelné objekty

`ConnectionFactory`, `QueueConnectionFactory`, `TopicConnectionFactory`, `XAConnectionFactory`, `XAQueueConnectionFactory`, `XATopicConnectionFactory`.

JMS dlouhý název nástroje pro administraci: **BALTYPE**

JMS krátký název nástroje pro administraci: **TYPE**

Programový přístup

Metody setter/getter

- `MQConnectionFactory.setBalancingApplicationType()`
- `MQConnectionFactory.getBalancingApplicationType()`

Hodnoty

Jednoduché
Použije se výchozí zpracování aplikací v jednotném klastru.

Tato hodnota se mapuje na volbu IBM MQ *BalancingOption* MQBNO_BALTYPE_SIMPLE.

POŽADAVEK_ODPOVĚĎ
Aplikace nebude požádána o opětovné připojení, pokud **MQPUT** nebyl vyrovnán **MQGET**, pokud neuplynul časový limit.

Tato hodnota se mapuje na volbu IBM MQ *BalancingOption* MQBNO_BALTYPE_REQREP.

V 9.3.4 BALTIMEOUT

Řídí, jak jsou aplikace IBM MQ classes for JMS , které používají přenos klienta, vyváženy v jednotném klastru.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

JMS dlouhý název nástroje pro administraci: **BALTIMEOUT**

JMS krátký název nástroje pro administraci: **TIMEOUT**

Programový přístup

Metody setter/getter

- MQConnectionFactory.setBalancingTimeout()
- MQConnectionFactory.getBalancingTimeout()

Hodnoty

Nikdy
Aplikaci nikdy nevyprší časový limit pro účely nového vyvážení v jednotném klastru.

Tato hodnota se mapuje na volbu IBM MQ *BalancingOption* MQBNO_TIMEOUT_NEVER.

IMMEDIATE
Aplikaci okamžitě vyprší časový limit pro účely vyvážení v jednotném klastru.

Tato hodnota se mapuje na volbu IBM MQ *BalancingOption* MQBNO_TIMEOUT_IMMEDIATE.

DEFAULT
Časový limit aplikace vyprší pro účely nového vyvážení v jednotném klastru po uplynutí výchozí doby 10 sekund.

Tato hodnota se mapuje na volbu IBM MQ *BalancingOption* MQBNO_TIMEOUT_AS_DEFAULT.

nn
Časový limit aplikace vyprší pro účely nového vyvážení v jednotném klastru po *nn* sekundách.

nn může být mezi 1 a 9999999999.

BROKERCCDURSUBQ

Název fronty, ze které jsou načítány zprávy trvalého odběru pro ConnectionConsumer.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci JMS : BROKERCCDURSUBQ

Krátký název nástroje pro administraci JMS : CCDSUB

Programový přístup

Metody setter/getter

- MQTopic.setBrokerCCDurSubQueue()
- MQTopic.getBrokerCCDurSubQueue()

Hodnoty

SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERCCSUBQ

Název fronty, z níž jsou načítány zprávy netrvalého odběru pro objekt ConnectionConsumer.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : BROKERCCSUBQ

Krátký název nástroje pro administraci JMS : CCSUB

Programový přístup

Metody setter/getter

- MQConnectionFactory.setBrokerCCSubQueue()
- MQConnectionFactory.getBrokerCCSubQueue()

Hodnoty

SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERCONQ

Název řídicí fronty zprostředkovatele.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : BROKERCONQ

Krátký název nástroje pro administraci JMS : BCON

Programový přístup

Metody setter/getter

- MQConnectionFactory.setBrokerControlQueue()
- MQConnectionFactory.getBrokerControlQueue()

Hodnoty

SYSTEM.BROKER.CONTROL.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERDURSUBQ

Při použití parametru IBM MQ classes for JMS v režimu migrace poskytovatele systému zpráv IBM MQ tato vlastnost určuje název fronty, z níž jsou načítány zprávy trvalého odběru.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci JMS : BROKERDURSUBQ

Krátký název nástroje pro administraci JMS : BDSUB

Programový přístup

Metody setter/getter

- MQTopic.setBrokerDurSubQueue()
- MQTopic.getBrokerDurSubQueue()

Hodnoty

SYSTEM.JMS.D.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec
Začíná řetězcem SYSTEM.JMS.D

Související úlohy
Konfigurace vlastnosti JMS **PROVIDERVERSION**

BROKERPUBQ

Název fronty, kam jsou odesílány publikované zprávy (fronta proudu).

Použitelné objekty

ConnectionFactory, TopicConnectiontovárna, téma, XAConnectionFactory, XATopicConnectiontovárna

Dlouhý název nástroje pro administraci JMS : BROKERPUBQ

Krátký název nástroje pro administraci JMS : BPUB

Programový přístup

Metody setter/getter

- MQConnectionFactory.setBrokerPubQueue
- MQConnectionFactory.getBrokerPubQueue

Hodnoty

SYSTEM.BROKER.DEFAULT.STREAM
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERPUBQMGR

Název správce front, který vlastní frontu, do které jsou odesílány zprávy publikované v tématu.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci JMS : BROKERPUBQMGR

Krátký název nástroje pro administraci JMS : BPQM

Programový přístup

Metody setter/getter

- MQTopic.setBrokerPubQueueManager()
- MQTopic.getBrokerPubQueueManager()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERQMGR

Název správce front, v němž je zprostředkovatel spuštěn.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : BROKERQMGR

Krátký název nástroje pro administraci JMS : BQM

Programový přístup

Metody setter/getter

- MQConnectionFactory.setBrokerQueueManager()
- MQConnectionFactory.getBrokerQueueManager()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERSUBQ

Při použití parametru IBM MQ classes for JMS v režimu migrace poskytovatele systému zpráv IBM MQ tato vlastnost určuje název fronty, ze které jsou načítány zprávy netrvalého odběru.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : BROKERSUBQ

Krátký název nástroje pro administraci JMS : BSUB

Programový přístup

Metody setter/getter

- MQConnectionFactory.setBrokerSubQueue()
- MQConnectionFactory.getBrokerSubQueue()

Hodnoty

SYSTEM.JMS.ND.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec
Začíná řetězcem SYSTEM.JMS.ND

Související úlohy

Konfigurace vlastnosti JMS **PROVIDERVERSION**

BROKERVER

Verze používaného zprostředkovatele.

Použitelné objekty

ConnectionFactory, TopicConnectiontovárna, téma, XAConnectionFactory, XATopicConnectiontovárna

Dlouhý název nástroje pro administraci JMS : BROKERVER

Krátký název nástroje pro administraci JMS : BVER

Programový přístup

Metody setter/getter

- MQConnectionFactory.setBrokerVersion ()
- MQConnectionFactory.getBrokerVersion ()

Hodnoty

V1
Chcete-li použít zprostředkovatele publikování/odběru IBM MQ nebo zprostředkovatele IBM MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker nebo WebSphere Business Integration Message Broker v režimu kompatibility. Jedná se o výchozí hodnotu, pokud je volba TRANSPORT nastavena na hodnotu BIND nebo CLIENT.

V2
Chcete-li použít zprostředkovatele produktu IBM MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker nebo WebSphere Business Integration Message Broker v nativním režimu. Toto je výchozí hodnota, pokud je TRANSPORT nastaven na DIRECT nebo DIRECTHTTP.

nespecifikováno
Po migraci zprostředkovatele z verze V6 na verzi V7nastavte tuto vlastnost tak, aby se záhlaví RFH2 již nepoužívala. Po migraci již tato vlastnost není relevantní.

CCDTURL

Jednotný lokátor prostředků (URL), který identifikuje název a umístění souboru obsahujícího tabulku definic kanálů klienta a určuje způsob přístupu k souboru.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : CCDTURL

Krátký název nástroje pro administraci JMS : CCDT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setCCDTURL()
- MQConnectionFactory.getCCDTURL()

Hodnoty

null
Toto je výchozí hodnota.

Lokátor jednotného prostředku (URL)

CCSID

Pro továrny připojení tato vlastnost uvádí ID kódované znakové sady (CCSID), které se má použít pro interní datové toky se správcem front. Pro cíle vlastnost definuje CCSID, který se má použít k zakódování řetězcových dat v adresách MapMessages, StreamMessagesa TextMessages vložených do tohoto místa určení.

Poznámka: Obvykle není nutné měnit tuto vlastnost pro továrny připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : CCSID

Krátký název nástroje pro administraci JMS : CCS

Programový přístup

Metody setter/getter

- MQConnectionFactory.setCCSID()
- MQConnectionFactory.getCCSID()

Hodnoty

819
Výchozí hodnota pro továrnu připojení.

1208
Výchozí hodnota pro cíl.

Libovolné kladné celé číslo

Související pojmy
JMS převod zpráv

CHANNEL

Název používaného kanálu připojení klienta.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : CHANNEL

Krátký název nástroje pro administraci JMS : CHAN

Programový přístup

Metody setter/getter

- MQConnectionFactory.setChannel()
- MQConnectionFactory.getChannel()

Hodnoty

SYSTEM.DEF.SVRCONN
Toto je výchozí hodnota.

Libovolný platný řetězec

CLEANUP

Úroveň vyčištění pro úložiště odběrů BROKER nebo MIGRATE.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : CLEANUP

Krátký název nástroje pro administraci JMS : CL

Programový přístup

Metody setter/getter

- MQConnectionFactory.setCleanupÚroveň ()
- MQConnectionFactory.getCleanupÚroveň ()

Hodnoty

Bezpečný
Použijte bezpečné vyčištění. Toto je výchozí hodnota.

ASPROP
Použijte bezpečné, silné nebo žádné vyčištění podle vlastnosti nastavené na příkazovém řádku Java .

ŽÁDNÉ
Nepoužívat čištění.

velká
Použijte silné vyčištění.

CLEANUPINT

Interval, v milisekundách, mezi provedeními na pozadí obslužného programu vyčištění publikování/odběru.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : CLEANUPINT

Krátký název nástroje pro administraci JMS : CLINT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setCleanupInterval ()
- MQConnectionFactory.getCleanupInterval ()

Hodnoty

3600000
Toto je výchozí hodnota.

Libovolné kladné celé číslo

ConnectionNameList

Seznam názvů připojení TCP/IP. Seznam je vyzkoušen v pořadí, jednou za každý opakovaný pokus o připojení.

Použitelné objekty

ConnectionFactory, QueueConnectiontovárna, TopicConnectiontovárna

Dlouhý název nástroje pro administraci JMS : CONNECTIONNAMELIST

Krátký název nástroje pro administraci JMS : CNLIST

Programový přístup

Metody setter/getter

- MQConnectionFactory.setconnectionNameList ()
- MQConnectionFactory.getconnectionNameList ()

Hodnoty

Čárkami oddělený seznam HOSTNAME (PORT). HOSTNAME může být buď název DNS, nebo adresa IP.

Výchozí hodnota parametru PORT je 1414.

CLIENTRECONNECTOPTIONS

Volby, které řídí opětovné připojení.

Použitelné objekty

ConnectionFactory, QueueConnectiontovárna, TopicConnectiontovárna

Dlouhý název nástroje pro administraci JMS : CLIENTRECONNECTOPTIONS

Krátký název nástroje pro administraci JMS : CROPT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setClientReconnectOptions()
- MQConnectionFactory.getClientReconnectOptions()

Hodnoty

QMGR

Aplikace se může znovu připojit ke stejnému správci front, ke kterému je původně připojena.

Pokud má správce front, ke kterému se aplikace pokouší připojit, jak je uvedeno v seznamu názvů připojení, jiný identifikátor QMID než správce front, ke kterému se původně připojil, je vrácena chyba s kódem příčiny `MQRC_RECONNECT_QMID_MISMATCH` .

Tuto hodnotu použijte, pokud lze aplikaci znovu připojit, ale existuje afinita mezi aplikací IBM MQ classes for JMS a správcem front, ke kterému poprvé navázala připojení.

Tuto hodnotu vyberte, chcete-li, aby se aplikace automaticky znovu připojovala k rezervní instanci správce front s vysokou dostupností.

Chcete-li tuto hodnotu použít programově, použijte konstantu WMQConstants.WMQ_CLIENT_RECONNECT_Q_MGR.

ANY

Aplikace se může znovu připojit k libovolnému správci front zadanému v seznamu názvů připojení.

Volbu opětovného připojení použijte pouze v případě, že mezi třídami IBM MQ pro aplikaci JMS a správcem front, se kterým původně navázala připojení, neexistuje žádná afinita.

Chcete-li použít tuto hodnotu z programu, použijte konstantu WMQConstants.WMQ_CLIENT_RECONNECT.

VYPNUTO

Aplikace nebude opakovat připojení.

Chcete-li tuto hodnotu použít programově, použijte konstantu WMQConstants.WMQ_CLIENT_RECONNECT_DISABLED.

ASDEF

Zda se aplikace automaticky znovu připojí, závisí na hodnotě IBM MQ atributu kanálu DefReconnect.

Toto je výchozí hodnota.

Chcete-li použít tuto hodnotu z programu, použijte konstantu WMQConstants.WMQ_CLIENT_RECONNECT_AS_DEF.

CLIENTRECONNECTTIMEOUT

Doba před ukončením opakovaných pokusů o připojení.

Použitelné objekty

ConnectionFactory, QueueConnectiontovárna, TopicConnectiontovárna

Dlouhý název nástroje pro administraci JMS : CLIENTRECONNECTTIMEOUT

Krátký název nástroje pro administraci JMS : CRT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setClientReconnectTimeout()
- MQConnectionFactory.setClientReconnectTimeout()

Hodnoty

Interval v sekundách. Výchozí hodnota 1800 (30 minut).

CLIENTID

Identifikátor klienta je použit k jedinečné identifikaci připojení aplikace k trvalým odběrům.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : CLIENTID

Krátký název nástroje pro administraci JMS : CID

Programový přístup

Metody setter/getter

- MQConnectionFactory.setClientId ()
- MQConnectionFactory.getClientId ()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec

CLONESUPP

Zda mohou být dvě nebo více instancí stejného trvalého odběratele tématu spuštěny současně.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : CLONESUPP

Krátký název nástroje pro administraci JMS : CLS

Programový přístup

Metody setter/getter

- MQConnectionFactory.setCloneSupport ()
- MQConnectionFactory.getCloneSupport ()

Hodnoty

VYPNUTO
V daném okamžiku může být spuštěna pouze jedna instance trvalého odběratele tématu. Toto je výchozí hodnota.

POVOLENO
Dvě nebo více instancí stejného trvalého odběratele tématu lze spustit současně, ale každá instance musí být spuštěna v samostatném prostředí JVM (Java Virtual Machine).

COMPHDR

Seznam technik, které lze použít pro kompresi dat záhlaví na připojení.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : COMPHDR

Krátký název nástroje pro administraci JMS : HC

Programový přístup

Metody setter/getter

- MQConnectionFactory.setHdrCompList()
- MQConnectionFactory.getHdrCompList()

Hodnoty

ŽÁDNÉ
Toto je výchozí hodnota.

SYSTÉM
Provede se komprese záhlaví zprávy RLE.

COMPMSG

Seznam technik, které lze použít pro kompresi dat zprávy na připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : COMPMSG

Krátký název nástroje pro administraci JMS : MC

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMsgCompList()
- MQConnectionFactory.getMsgCompList()

Hodnoty

ŽÁDNÉ
Toto je výchozí hodnota.

Seznam jedné nebo více následujících hodnot oddělených prázdnými znaky:
RLE ZLIBFAST ZLIBHIGH

CONNOPT

Určuje způsob připojení aplikací IBM MQ classes for JMS , které používají přenos vazeb, ke správci front.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

Dlouhý název nástroje pro administraci JMS : CONNOPT

Krátký název nástroje pro administraci JMS : CNOPT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMQConnectionVolby ()
- MQConnectionFactory.getMQConnectionVolby ()

Hodnoty

STANDARD
Povaha vazby mezi aplikací a správcem front závisí na hodnotě atributu *DefaultBindType* správce front. Hodnota STANDARD se mapuje na volbu IBM MQ *ConnectOption* MQCNO_STANDARD_BINDING.

SHARED
Aplikace a lokální agent správce front jsou spuštěny v oddělených jednotkách provedení, ale sdílejí některé prostředky. Tato hodnota se mapuje na volbu IBM MQ *ConnectOption* MQCNO_SHARED_BINDING.

Izolovaný
Aplikace a lokální agent správce front jsou spouštěny v samostatných jednotkách provedení a nesdílejí žádné prostředky. Hodnota ISOLATED je mapována na volbu IBM MQ *ConnectOption* MQCNO_ISOLATED_BINDING.

Rychlý
Aplikace a lokální agent správce front jsou spuštěny ve stejné jednotce provedení. Tato hodnota se mapuje na volbu IBM MQ *ConnectOption* MQCNO_FASTPATH_BINDING.

SERIALQM-řízení kvality
Aplikace vyžaduje výhradní použití značky připojení v rámci oboru správce front. Tato hodnota se mapuje na IBM MQ *ConnectOption* MQCNO_SERIALIZE_CONN_TAG_Q_MGR.

SERIALQSG
Aplikace vyžaduje výhradní použití značky připojení v rámci oboru skupiny sdílení front, do které patří daný správce front. Hodnota SERIALQSG se mapuje na volbu IBM MQ *ConnectOption* MQCNO_SERIALIZE_CONN_TAG_QSG.

RESTRICTQM
Aplikace požaduje sdílené použití značky připojení, existují však omezení pro sdílené použití značky připojení v rámci oboru správce front. Tato hodnota se mapuje na volbu IBM MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_Q_MGR.

RESTRICTQSG
Aplikace požaduje sdílené použití značky připojení, existují však omezení pro sdílené použití značky připojení v oboru skupiny sdílení front, do které patří správce front. Tato hodnota se mapuje na volbu IBM MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_QSG.

Další informace o volbách připojení IBM MQ naleznete v tématu Připojení ke správci front pomocí volání MQCONNX.

CONNTAG

Značka, kterou správce front přidružuje k prostředkům aktualizovaným aplikací v rámci pracovní jednotky, zatímco je aplikace připojena ke správci front.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : CONNTAG

Krátký název nástroje pro administraci JMS : CNTAG

Programový přístup

Metody setter/getter

- MQConnectionFactory.setConnTag ()
- MQConnectionFactory.getConnTag ()

Hodnoty

Bajtové pole 128 prvků, kde každý prvek je 0
Toto je výchozí hodnota.

Libovolný řetězec
Hodnota je oříznuta, pokud je delší než 128 bajtů.

DESCRIPTION

Popis uloženého objektu.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : DESCRIPTION

Krátký název nástroje pro administraci JMS : DESC

Programový přístup

Metody setter/getter

- MQConnectionFactory.setDescription()
- MQConnectionFactory.getDescription()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec
`

DIRECTAUTH

Zda se ověření TLS používá na připojení v reálném čase ke zprostředkovateli.

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : DIRECTAUTH

Krátký název nástroje pro administraci JMS : DAUTH

Programový přístup

Metody setter/getter

- MQConnectionFactory.setDirectAuth ()
- MQConnectionFactory.getDirectAuth () ()

Hodnoty

ZÁKLADNÍ
Žádné ověření, ověření pomocí jména uživatele nebo ověření pomocí hesla. Toto je výchozí hodnota.

Certifikát
Ověření certifikátu veřejného klíče.

ENCODING

Způsob reprezentace číselných dat v těle zprávy při odeslání zprávy do tohoto místa určení. Tato vlastnost určuje reprezentaci binárních celých čísel, pakovaných desetinných celých čísel a čísel s pohyblivou řádovou čárkou.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : ENCODING

Krátký název nástroje pro administraci JMS : ENC

Programový přístup

Metody setter/getter

- MQDestination.setEncoding()
- MQDestination.getEncoding()

Hodnoty

Vlastnost ENCODING

Platné hodnoty, které může vlastnost ENCODING převzít, jsou sestaveny ze tří dílčích vlastností:

Kódování celých čísel
Buď normální, nebo obrácené

Kódování desetinných čísel
Buď normální, nebo obrácené

kódování s pohyblivou řádovou čárkou
IEEE normální, IEEE obrácené nebo z/OS

Vlastnost ENCODING je vyjádřena jako tříznakový řetězec s následující syntaxí:

```
{N|R}{N|R}{N|R|3}
```

V tomto řetězci:

- N označuje normální
- R označuje obrácené
- 3 označuje z/OS
- První znak představuje *kódování celých čísel*
- Druhý znak představuje *kódování desetinných čísel* .
- Třetí znak představuje *kódování s pohyblivou řádovou čárkou*

To poskytuje sadu dvanácti možných hodnot pro vlastnost ENCODING .

Existuje další hodnota, řetězec NATIVE, který nastavuje odpovídající hodnoty kódování pro platformu Java .

Následující příklady ukazují platné kombinace pro ENCODING:

```
ENCODING(NNR)
ENCODING(NATIVE)
ENCODING(RR3)
```

EXPIRY

Čas, po kterém vyprší platnost zpráv v místě určení.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : EXPIRAČNÍ

Krátký název nástroje pro administraci JMS : EXP

Programový přístup

Metody setter/getter

- MQDestination.setExpiry()
- MQDestination.getExpiry()

Hodnoty

Aplikace
Vypršení platnosti může být definováno aplikací JMS . Toto je výchozí hodnota.

UNLIM
Nevyskytne se žádné vypršení platnosti.

0
Nevyskytne se žádné vypršení platnosti.

Jakékoli kladné celé číslo představující vypršení platnosti v milisekundách.

FAILIFQUIESCE

Tato vlastnost určuje, zda se volání určitých metod nezdaří, pokud se správce front nachází ve stavu uvedení do klidového stavu nebo pokud se aplikace připojuje ke správci front pomocí transportu CLIENT a kanál, který aplikace používá, byl uveden do klidového stavu, například pomocí příkazu **STOP CHANNEL** nebo **STOP CHANNEL MODE(QUIESCE)** MQSC.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : FAILIFQUIESCE

Krátký název nástroje pro administraci JMS : FIQ

Programový přístup

Metody setter/getter

- MQConnectionFactory.setFailIfQuiesce()
- MQConnectionFactory.getFailIfQuiesce()

Hodnoty

YES
Volání určitých metod se nezdaří, pokud se správce front nachází ve stavu uvedení do klidového stavu nebo pokud je kanál používaný pro připojení ke správci front uveden do klidového stavu. Pokud aplikace zjistí některou z těchto podmínek, může dokončit svou okamžitou úlohu a zavřít připojení, což umožní zastavení správce front nebo instance kanálu. Toto je výchozí hodnota.

NO
Žádné volání metody neselhává, protože buď správce front, nebo kanál používaný pro připojení ke správci front se nachází ve stavu uvedení do klidového stavu. Zadáte-li tuto hodnotu, aplikace nemůže zjistit, že správce front nebo kanál jsou uvedeny do klidového stavu. Aplikace může pokračovat v provádění operací se správcem front, a zabránit tak zastavení správce front.

HOSTNAME

V případě připojení ke správci front se jedná o název hostitele nebo adresu IP systému, v němž je spuštěn správce front, nebo v případě připojení v reálném čase ke zprostředkovateli o název hostitele nebo adresu IP systému, v němž je zprostředkovatel spuštěn.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : HOSTNAME

Krátký název nástroje pro administraci JMS : HOST

Programový přístup

Metody setter/getter

- MQConnectionFactory.setHostName ()
- MQConnectionFactory.getHostName ()

Hodnoty

lokální hostitel
Toto je výchozí hodnota.

Libovolný platný řetězec

LOCALADDRESS

Pro připojení ke správci front tato vlastnost uvádí buď lokální síťové rozhraní, které se má použít, nebo lokální port nebo rozsah lokálních portů, které se mají použít.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : LOCALADDRESS

Krátký název nástroje pro administraci JMS : LA

Programový přístup

Metody setter/getter

- MQConnectionFactory.setLocalAddress ()

- MQConnectionFactory.getLocalAddress ()

Hodnoty

"" (prázdný řetězec)
Toto je výchozí hodnota.

Řetězec ve formátu [ip-addr] [(low-port [, high-port])]
Několik příkladů:

`192.0.2.0`

Kanál se lokálně váže na adresu 192.0.2.0 .

`192.0.2.0(1000)`

Kanál je lokálně svázán s adresou 192.0.2.0 a používá port 1000.

`192.0.2.0(1000,2000)`

Kanál je lokálně svázán s adresou 192.0.2.0 a používá port v rozsahu 1000 až 2000.

`(1000)`

Kanál se lokálně váže na port 1000.

`(1000,2000)`

Kanál se lokálně váže na port v rozsahu 1000 až 2000.

Místo adresy IP můžete zadat název hostitele. Pro připojení zprostředkovatele v reálném čase je tato vlastnost relevantní pouze při použití výběrového vysílání a hodnota vlastnosti nesmí obsahovat číslo portu nebo rozsah čísel portů. Jediné platné hodnoty vlastnosti v tomto případě jsou null, adresa IP nebo název hostitele.

MAPNAMESTYLE

Umožňuje použití stylu kompatibility pro názvy prvků MapMessage .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : MAPNAMESTYLE

Krátký název nástroje pro administraci JMS : MNST

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMapNameStyle()
- MQConnectionFactory.getMapNameStyle()

Hodnoty

STANDARD
Použije se standardní formát pojmenování prvku com.ibm.jms.JMSMapMessage . Toto je výchozí hodnota a umožňuje použití nepřípustných identifikátorů Java jako názvu prvku.

Kompatibilní
Použije se starší formát pojmenování prvku com.ibm.jms.JMSMapMessage . Jako název prvku lze použít pouze povolené identifikátory Java . To je zapotřebí pouze v případě, že jsou zprávy mapy odesílány do aplikace, která používá starší verzi produktu IBM MQ classes for JMS než 5.3.

MAXBUFFSIZE

Maximální počet přijatých zpráv, které lze uložit do vnitřní vyrovnávací paměti zpráv při čekání na zpracování aplikací. Tato vlastnost platí pouze v případě, že má vlastnost TRANSPORT hodnotu DIRECT nebo DIRECTHTTP.

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : MAXBUFFSIZE

Krátký název nástroje pro administraci JMS : MBSZ

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMaxBufferSize()
- MQConnectionFactory.getMaxBufferSize()

Hodnoty

1000
Toto je výchozí hodnota.

Libovolné kladné celé číslo

MDREAD

Tato vlastnost určuje, zda může aplikace JMS extrahovat hodnoty polí MQMD.

Použitelné objekty

Dlouhý název nástroje pro administraci JMS : MDREAD

Krátký název nástroje pro administraci JMS : MDR

Programový přístup

Metody setter/getter

- MQDestination.setMQMDReadEnabled()
- MQDestination.getMQMDReadEnabled()

Hodnoty

NO
Při odesílání zpráv nejsou vlastnosti JMS_IBM_MQMD* pro odeslanou zprávu aktualizovány tak, aby odrážely aktualizované hodnoty polí v deskriptoru MQMD. Při příjmu zpráv nejsou dostupné žádné z vlastností JMS_IBM_MQMD* v přijaté zprávě, i když odesilatel některé či všechny tyto vlastnosti nastavil. Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte hodnotu False.

Ano
Při odesílání zpráv jsou všechny vlastnosti JMS_IBM_MQMD* pro odeslanou zprávu aktualizovány tak, aby odrážely aktualizované hodnoty polí v deskriptoru MQMD, včetně vlastností, které odesilatel explicitně nenastavil. Při příjmu zpráv jsou pro přijatou zprávu k dispozici všechny vlastnosti JMS_IBM_MQMD*, včetně vlastností, které odesilatel explicitně nenastavil.

Pro programy použijte hodnotu True.

MDWRITE

Tato vlastnost určuje, zda může aplikace JMS nastavit hodnoty polí MQMD.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : MDWRITE

Krátký název nástroje pro administraci JMS : MDR

Programový přístup

Metody setter/getter

- MQDestination.setMQMDWriteEnabled()
- MQDestination.getMQMDWriteEnabled()

Hodnoty

NO
Všechny vlastnosti JMS_IBM_MQMD* jsou ignorovány a jejich hodnoty nejsou zkopírovány do základní struktury MQMD. Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte hodnotu False.

YES
Vlastnosti JMS_IBM_MQMD* jsou zpracovány. Jejich hodnoty jsou zkopírovány do podkladové struktury MQMD.

Pro programy použijte hodnotu True.

MDMSGCTX

Jaká úroveň kontextu zprávy má být nastavena aplikací JMS . Aby tato vlastnost mohla nabýt účinnosti, musí aplikace běžet s příslušným oprávněním kontextu.

Použitelné objekty

Dlouhý název nástroje pro administraci JMS : MDMSGCTX

Krátký název nástroje pro administraci JMS : MDCTX

Programový přístup

Metody setter/getter

- MQDestination.setMQMDMessageContext()
- MQDestination.getMQMDMessageContext()

Hodnoty

DEFAULT
Volání rozhraní API MQOPEN a struktura MQPMO neuvádějí žádné explicitní volby kontextu zprávy. Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte WMQ_MDCTX_DEFAULT.

KONTEX_IDENTITY_SET_IDENTITY_CONTEXT
Volání rozhraní API MQOPEN určuje volbu kontextu zprávy MQOO_SET_IDENTITY_CONTEXT a struktura MQPMO určuje volbu MQPMO_SET_IDENTITY_CONTEXT.

Pro programy použijte WMQ_MDCTX_SET_IDENTITY_CONTEXT.

NASTAVIT_VŠECHNY_KONTEXT
Volání MQOPEN API určuje volbu kontextu zprávy MQOO_SET_ALL_CONTEXT a struktura MQPMO určuje volbu MQPMO_SET_ALL_CONTEXT.

Pro programy použijte WMQ_MDCTX_SET_ALL_CONTEXT.

MSGBATCHSZ

Maximální počet zpráv, které mají být převzaty z fronty v jednom paketu při použití asynchronního doručování zpráv.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : MAXBUFFSIZE

Krátký název nástroje pro administraci JMS : MBSZ

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMsgBatchSize()
- MQConnectionFactory.getMsgBatchSize()

Hodnoty

10
Toto je výchozí hodnota.

Libovolné kladné celé číslo

MSGBODY

Určuje, zda aplikace JMS přistupuje k serveru MQRFH2 zprávy IBM MQ jako k části informačního obsahu zprávy.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : WMQ_MESSAGE_BODY

Krátký název nástroje pro administraci JMS : MBODY

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMessageBodyStyle()
- MQConnectionFactory.getMessageBodyStyle()

Hodnoty

Neuvedeno
Při odesílání produkt IBM MQ classes for JMS generuje nebo negeneruje a nezahrne záhlaví MQRFH2 v závislosti na hodnotě WMQ_TARGET_CLIENT. Při příjmu se chová jako hodnota JMS.

JMS
Při odesílání produkt IBM MQ classes for JMS automaticky vygeneruje záhlaví MQRFH2 a zahrne je do zprávy IBM MQ .

Při příjmu IBM MQ classes for JMS nastavte vlastnosti zprávy JMS podle hodnot v MQRFH2 (je-li k dispozici); neuvádí MQRFH2 jako součást těla zprávy JMS .

MQ
Při odesílání produkt IBM MQ classes for JMS negeneruje položku MQRFH2.

Při příjmu produkt IBM MQ classes for JMS prezentuje MQRFH2 jako součást těla zprávy JMS .

MSGRETENTION

Určuje, zda spotřebitel připojení uchovává nedoručené zprávy ve vstupní frontě.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory,

Dlouhý název nástroje pro administraci JMS : MSGRETENTION

Krátký název nástroje pro administraci JMS : MRET

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMessageUchování ()
- MQConnectionFactory.getMessageUchování ()

Hodnoty

Ano
Nedoručené zprávy zůstávají ve vstupní frontě. Toto je výchozí hodnota.

Ne
Nedoručené zprávy jsou zpracovávány v závislosti na jejich možnostech odebrání.

MSGSELECTION

Určuje, zda je výběr zpráv proveden agentem IBM MQ classes for JMS nebo zprostředkovatelem. Pokud má TRANSPORT hodnotu DIRECT, výběr zprávy je vždy proveden zprostředkovatelem a hodnota MSGSELECTION se ignoruje. Výběr zpráv zprostředkovatelem není podporován, pokud má BROKERVER hodnotu V1.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : MSGSELECTION

Krátký název nástroje pro administraci JMS : MSEL

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMessageSelection ()
- MQConnectionFactory.getMessageSelection ()

Hodnoty

CLIENT
Výběr zpráv provádí IBM MQ classes for JMS. Toto je výchozí hodnota.

BROKER
Výběr zpráv provádí zprostředkovatel.

MULTICAST

Chcete-li povolit výběrové vysílání na připojení v reálném čase ke zprostředkovateli a je-li tato volba povolena, určit přesný způsob, jakým se výběrové vysílání používá k doručování zpráv ze zprostředkovatele spotřebiteli zpráv. Vlastnost nemá žádný vliv na to, jak producent zpráv odesílá zprávy zprostředkovateli.

Použitelné objekty

ConnectionFactory, TopicConnectionTovárna, Téma

Dlouhý název nástroje pro administraci JMS : MULTICAST

Krátký název nástroje pro administraci JMS : MCAST

Programový přístup

Metody setter/getter

- MQConnectionFactory.setMulticast()
- MQConnectionFactory.getMulticast()

Hodnoty

VYPNUTO
Zprávy nejsou doručeny spotřebiteli zpráv pomocí přenosu výběrového vysílání. Jedná se o výchozí hodnotu pro objekty ConnectionFactory a TopicConnectionFactory.

ASCF
Zprávy jsou doručovány spotřebiteli zpráv v souladu s nastavením výběrového vysílání pro továrnu připojení přidruženou ke spotřebiteli zpráv. Nastavení výběrového vysílání pro továrnu připojení je zaznamenáno v době vytvoření spotřebitele zpráv. Tato hodnota je platná pouze pro objekty tématu a je výchozí hodnotou pro objekty tématu.

POVOLENO
Pokud je téma konfigurováno pro výběrové vysílání ve zprostředkovateli, zprávy jsou doručeny spotřebiteli zpráv pomocí výběrového vysílání. Spolehlivá kvalita služby se používá, pokud je téma nakonfigurováno pro spolehlivé výběrové vysílání.

Spolehlivé
Je-li téma konfigurováno pro spolehlivé výběrové vysílání ve zprostředkovateli, jsou zprávy doručeny spotřebiteli zpráv pomocí výběrového vysílání se spolehlivou kvalitou služby. Pokud téma není konfigurováno pro spolehlivé výběrové vysílání, nemůžete pro toto téma vytvořit spotřebitele zpráv.

NOTR
Je-li téma konfigurováno pro výběrové vysílání ve zprostředkovateli, budou zprávy doručeny spotřebiteli zpráv pomocí výběrového vysílání. Spolehlivá kvalita služby se nepoužívá, i když je téma konfigurováno pro spolehlivé výběrové vysílání.

OPTIMISTICPUBLICATION

Tato vlastnost určuje, zda produkt IBM MQ classes for JMS vrátí řízení okamžitě vydavateli, který publikoval zprávu, nebo zda vrátí řízení pouze po dokončení veškerého zpracování přidruženého k volání a může ohlásit výsledek vydavateli.

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : OPTIMISTICPUBLICATION

Krátký název nástroje pro administraci JMS : OPTPUB

Programový přístup

Metody setter/getter

- MQConnectionFactory.setOptimisticPublication ()
- MQConnectionFactory.getOptimisticPublication ()

Hodnoty

NO
Když vydavatel publikuje zprávu, produkt IBM MQ classes for JMS nevrátí řízení vydavateli, dokud nedokončí veškeré zpracování přidružené k volání a nebude moci oznámit výsledek vydavateli. Toto je výchozí hodnota.

YES
Když vydavatel publikuje zprávu, produkt IBM MQ classes for JMS vrátí řízení vydavateli okamžitě před dokončením veškerého zpracování přidruženého k volání a může oznámit výsledek vydavateli. Produkt IBM MQ classes for JMS ohlásí výsledek pouze v případě, že vydavatel zprávu potvrdí.

OUTCOMENOTIFICATION

Tato vlastnost určuje, zda má produkt IBM MQ classes for JMS okamžitě vrátit řízení odběrateli, který právě potvrdil nebo potvrdil zprávu, nebo zda vrací řízení pouze po dokončení veškerého zpracování přidruženého k volání a může oznámit výsledek odběrateli.

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : OUTCOMENOTIFICATION

Krátký název nástroje pro administraci JMS : NOTIFY

Programový přístup

Metody setter/getter

- MQConnectionFactory.setOutcomeNotification ()
- MQConnectionFactory.getOutcomeNotification ()

Hodnoty

YES
Když odběratel potvrdí nebo potvrdí zprávu, produkt IBM MQ classes for JMS nevrátí řízení odběrateli, dokud nedokončí veškeré zpracování přidružené k volání a nebude moci oznámit výsledek odběrateli. Toto je výchozí hodnota.

NO
Když odběratel potvrdí nebo potvrdí zprávu, produkt IBM MQ classes for JMS vrátí řízení odběrateli okamžitě před tím, než dokončí veškeré zpracování přidružené k volání, a může oznámit výsledek odběrateli.

PERSISTENCE

Perzistence zpráv odeslaných do místa určení.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : PERSISTENCE

Krátký název nástroje pro administraci JMS : PER

Programový přístup

Metody setter/getter

- MQDestination.setPersistence()
- MQDestination.getPersistence()

Hodnoty

Aplikace
Perzistence je definována aplikací JMS . Toto je výchozí hodnota.

QDEF
Perzistence přebírá výchozí hodnotu fronty.

PERS
Zprávy jsou trvalé.

JINÉ NEŽ
Zprávy jsou přechodné.

VYSOKÁ
Další informace o použití této hodnoty viz JMS trvalé zprávy .

POLLINGINT

Pokud každý modul listener pro zprávy v rámci relace nemá ve své frontě žádnou vhodnou zprávu, jedná se o maximální interval (v milisekundách), který uplyne, než se každý modul listener pro zprávy znovu pokusí získat zprávu ze své fronty. Pokud se často stává, že pro žádný z modulů listener pro zprávy v relaci není k dispozici žádná vhodná zpráva, zvažte zvýšení hodnoty této vlastnosti. Tato vlastnost je relevantní pouze v případě, že má volba TRANSPORT hodnotu BIND nebo CLIENT.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : POLLINGINT

Krátký název nástroje pro administraci JMS : PINT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setPollingInterval ()
- MQConnectionFactory.getPollingInterval ()

Hodnoty

5000
Toto je výchozí hodnota.

Libovolné kladné celé číslo

PORT

V případě připojení ke správci front se jedná o číslo portu, na kterém správce front naslouchá, nebo v případě připojení zprostředkovatele v reálném čase o číslo portu, na kterém zprostředkovatel naslouchá připojením v reálném čase.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : PORT

Krátký název nástroje pro administraci JMS : PORT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setPort()
- MQConnectionFactory.getPort()

Hodnoty

1414
Jedná se o výchozí hodnotu, pokud je volba TRANSPORT nastavena na hodnotu CLIENT.

1506
Toto je výchozí hodnota, pokud je TRANSPORT nastaven na DIRECT nebo DIRECTHTTP.

Libovolné kladné celé číslo

PRIORITY

Priorita pro zprávy odeslané do cíle.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : PRIORITY

Krátký název nástroje pro administraci JMS : PRI

Programový přístup

Metody setter/getter

- MQDestination.setPriority()
- MQDestination.getPriority()

Hodnoty

Aplikace
Priorita je definována aplikací JMS . Toto je výchozí hodnota.

QDEF
Priorita přebírá hodnotu předvolby fronty.

Libovolné celé číslo v rozsahu 0-9
Od nejnižší k nejvyšší.

PROCESSDURATION

Tato vlastnost určuje, zda odběratel zaručuje rychlé zpracování všech zpráv, které obdrží, před vrácením řízení do produktu IBM MQ classes for JMS.

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : PROCESSDURATION

Krátký název nástroje pro administraci JMS : PROCDUR

Programový přístup

Metody setter/getter

- MQConnectionFactory.setProcessDoba trvání ()
- MQConnectionFactory.getProcessDoba trvání ()

Hodnoty

NEZNÁMÉ
Odběratel nemůže poskytnout žádnou záruku o tom, jak rychle může zpracovat jakoukoli zprávu, kterou obdrží. Toto je výchozí hodnota.

Krátký
Odběratel zaručuje rychlé zpracování všech zpráv, které obdrží před vrácením řízení do produktu IBM MQ classes for JMS.

PROVIDERVERSION

Tato vlastnost rozlišuje tři provozní režimy systému zpráv IBM MQ : IBM MQ normální režim poskytovatele systému zpráv, IBM MQ normální režim poskytovatele systému zpráv s omezeními a IBM MQ režim migrace poskytovatele systému zpráv.

Normální režim poskytovatele systému zpráv IBM MQ používá k implementaci všechny funkce IBM MQ správce front JMS. Tento režim je optimalizován pro použití rozhraní API a funkcí produktu JMS 2.0 . Normální režim poskytovatele systému zpráv IBM MQ s omezeními používá rozhraní API produktu JMS 2.0 , ale ne nové funkce, jako jsou sdílené odběry, zpožděné doručení nebo asynchronní odeslání.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnection Factory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : PROVIDERVERSION

Krátký název nástroje pro administraci JMS : PVER

Programový přístup

Metody setter/getter

- MQConnectionFactory.setProviderVersion ()
- MQConnectionFactory.getProviderVersion ()

Hodnoty

Vlastnost **PROVIDERVERSION** lze nastavit na některou z hodnot 8 (normální režim), 7 (normální režim s omezeními), 6 (režim migrace) nebo `nespecifikováno` (výchozí hodnota). Hodnota, kterou zadáte

pro vlastnost **PROVIDERVERSION**, musí být řetězec. Pokud zadáte volbu 8, 7 nebo 6, můžete to provést v některém z následujících formátů:

- V.R.M.F
- V.R.M
- V.R
- V

kde V, R, M a F jsou celá čísla větší nebo rovná nule. Hodnoty R, M a F jsou nepovinné a můžete je použít k upřesnění v případě potřeby řízení s vysokou úrovní granularity. Chcete-li například použít úroveň **PROVIDERVERSION** 7, můžete nastavit **PROVIDERVERSION**=7, `7.0`, `7.0.0` nebo `7.0.0.0`.

8 - Normální režim
Aplikace JMS používá normální režim poskytovatele systémů zpráv IBM MQ. Normální režim používá všechny funkce správce front IBM MQ k implementaci služby JMS. Tento režim je optimalizovaný pro použití rozhraní API a funkčnosti JMS 2.0.

Pokud se připojujete ke správci front s úrovní příkazu 800 nebo novější, lze použít všechna rozhraní JMS 2.0 API a funkce, jako např. asynchronní odesílání, zpožděné doručení nebo sdílený odběr.

Pokud správce front určený v nastavení továrny připojení není správcem front IBM MQ 8.0.0 nebo novějším, metoda `createConnection` se nezdaří s výjimkou `JMSFMQ0003`.

Normální režim poskytovatele systému zpráv IBM MQ používá funkci sdílení konverzací a počet konverzací, které lze sdílet, je řízen vlastností **SHARECNV()** v kanálu připojení k serveru. Je-li tato vlastnost nastavena na hodnotu `0`, nemůžete používat normální režim poskytovatele systému zpráv IBM MQ a metoda `createConnection` se nezdaří s výjimkou `JMSCC5007`.

7 - Normální režim s omezeními
Aplikace JMS používá normální režim s omezeními poskytovatele systému zpráv IBM MQ. Tento režim používá rozhraní JMS 2.0 API, ale ne nové funkce, jako sdílení odběrů, odložené doručení nebo asynchronní odeslání.

Pokud jste nastavili hodnotu **PROVIDERVERSION** na 7, je k dispozici pouze normální režim poskytovatele systému zpráv IBM MQ s omezeními.

Pokud se připojujete s použitím normálního režimu s omezeními ke správci front s úrovní příkazu nižší než 800, můžete použít rozhraní JMS 2.0 API, nikoli však funkce asynchronního odesílání, zpožděného doručení nebo sdíleného odběru.

Normální režim poskytovatele systému zpráv IBM MQ s omezeními používá funkci sdílení konverzací a počet konverzací, které lze sdílet, je řízen vlastností **SHARECNV()** v kanálu připojení k serveru. Je-li tato vlastnost nastavena na hodnotu `0`, nemůžete používat normální režim s omezeními poskytovatele systému zpráv IBM MQ a metoda `createConnection` se nezdaří s výjimkou `JMSCC5007`.

6 - Režim migrace
Aplikace JMS používá režim migrace poskytovatele systémů zpráv IBM MQ.

Pomocí tohoto režimu se můžete připojit ke správci front IBM MQ 8.0nebo novějšímu, ale nepoužívají se žádné nové funkce správce front IBM MQ classes for JMS , například dopředné čtení nebo proudové zpracování.

Máte-li klienta IBM MQ 8.0 nebo novějšího, který se připojuje ke správci front IBM MQ 8.0 nebo novějšímu, pak je výběr zpráv proveden správcem front spíše než v systému klienta.

Je-li zadán režim migrace poskytovatele systému zpráv IBM MQ a vy se pokusíte použít některé z rozhraní JMS 2.0 API, volání metody API se nezdaří s výjimkou `JMSCC5007`.

neuvedeno (výchozí)
Vlastnost **PROVIDERVERSION** je standardně nastavena na hodnotu *nespecifikováno*.

Továrna připojení, která byla vytvořena v předchozí verzi produktu IBM MQ classes for JMS v JNDI, převezme tuto hodnotu, když se použije továrna připojení novou verzí IBM MQ classes for JMS. Při určování použitého režimu operací se používá následující algoritmus. Tento algoritmu se použije

při volání metody `createConnection` a používá další aspekty továrny připojení k určení, zda je potřeba normální režim poskytovatele systému zpráv IBM MQ, normální režim s omezeními nebo režim migrace poskytovatele systému zpráv IBM MQ.

1. Nejprve se pokusí použít normální režim poskytovatele systému zpráv IBM MQ.
2. Pokud připojený správce front není IBM MQ 8.0 nebo novější, je proveden pokus o použití normálního režimu poskytovatele systémů zpráv IBM MQ s omezeními.
3. Pokud připojený správce front není verze IBM WebSphere MQ 7.0.1 nebo novější, připojení se zavře a použije se režim migrace poskytovatele systému zpráv IBM MQ.
4. Je-li vlastnost **SHARECNV** kanálu připojení k serveru nastavena na hodnotu 0, je připojení uzavřeno a místo něj bude použit režim migrace poskytovatele systému zpráv IBM MQ.
5. Je-li volba **BROKERVER** nastavena na hodnotu V1 nebo na výchozí *nespecifikovanou* hodnotu, bude nadále používán normální režim poskytovatele systému zpráv IBM MQ .

   Informace o parametru PSMPDE příkazu ALTER QMGR a další informace o kompatibilitě viz ALTER QMGR.
6. Je-li vlastnost **BROKERVER** nastavena na hodnotu V2, provedená akce bude záviset na hodnotě **BROKERQMGR**:
   - Pokud je hodnota **BROKERQMGR** `prázdná`:

     Pokud je frontu uvedenou vlastností **BROKERCONQ** možné otevřít pro výstup (tj. akce `MQOPEN` bude pro výstup úspěšná), a pokud je vlastnost **PSMODE** na správci front nastavena na hodnotu `COMPAT` nebo `DISABLED`, pak se použije režim migrace poskytovatele systému zpráv IBM MQ.
   - Pokud frontu určenou pomocí vlastnosti **BROKERCONQ** nelze otevřít pro výstup, nebo je atribut **PSMODE** nastaven na hodnotu `ENABLED`:

     použije se normální režim poskytovatele systému zpráv IBM MQ.
   - Pokud je hodnota **BROKERQMGR** `neprázdná`:

     použije se režim migrace poskytovatele systému zpráv IBM MQ.

Pokud nemůžete změnit továrnu připojení, kterou používáte, můžete použít vlastnost `com.ibm.msg.client.wmq.overrideProviderVersion` k potlačení jakéhokoli nastavení v továrně připojení. Tento přepis platí pro všechny továrny připojení v prostředí JVM, ale skutečné objekty továrny připojení se nezmění.

Související úlohy

Konfigurace vlastnosti JMS **PROVIDERVERSION**

PROXYHOSTNAME

Název hostitele nebo adresa IP systému, na kterém je spuštěn server proxy při použití připojení v reálném čase ke zprostředkovateli prostřednictvím serveru proxy.

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : PROXYHOSTNAME

Krátký název nástroje pro administraci JMS : PHOST

Programový přístup

Metody setter/getter

- MQConnectionFactory.setProxyHostName()
- MQConnectionFactory.getProxyHostName()

Hodnoty

null
Název hostitele serveru proxy. Toto je výchozí hodnota.

PROXYPORT

Číslo portu, na kterém server proxy naslouchá při použití připojení v reálném čase ke zprostředkovateli prostřednictvím serveru proxy.

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : PROXYPORT

Krátký název nástroje pro administraci JMS : PPORT

Programový přístup

Metody setter/getter

MQConnectionFactory.setProxyPort ()

MQConnectionFactory.getProxyPort ()

Hodnoty

443
Číslo portu serveru proxy. Toto je výchozí hodnota.

PUBACKINT

Počet zpráv publikovaných vydavatelem, než produkt IBM MQ classes for JMS vyžádá potvrzení od zprostředkovatele.

Když snížíte hodnotu této vlastnosti, produkt IBM MQ classes for JMS bude požadovat potvrzení častěji, a proto se výkon vydavatele sníží. Když zvýšíte hodnotu, IBM MQ classes for JMS trvá delší dobu, než vygeneruje výjimku, pokud se zprostředkovatel nezdaří. Tato vlastnost je relevantní pouze v případě, že má volba TRANSPORT hodnotu BIND nebo CLIENT.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : PROXYPORT

Krátký název nástroje pro administraci JMS : PPORT

Programový přístup

Metody setter/getter

MQConnectionFactory.setPubAckInterval()

MQConnectionFactory.getPubAckInterval()

Hodnoty

25
Výchozí hodnotou může být libovolné kladné celé číslo.

PUTASYNCALLOWED

Tato vlastnost určuje, zda producenti zpráv mohou používat k odesílání zpráv do tohoto místa určení asynchronní operace put.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : PUTASYNCALLOWED

Krátký název nástroje pro administraci JMS : PAALD

Programový přístup

Metody setter/getter

MQDestination.setPutAsyncAllowed()

MQDestination.getPutAsyncAllowed()

Hodnoty

AS_DEST
Zjistěte, zda jsou povolena asynchronní vložení, odkazem na definici fronty nebo tématu. Toto je výchozí hodnota.

AS_Q_DEF
Určete, zda jsou povolena asynchronní vložení, odkazem na definici fronty.

AS_TOPIC_DEF
Zjistěte, zda jsou povolena asynchronní vložení, na základě definice tématu.

NO
Asynchronní vložení nejsou povolena.

YES
Asynchronní vložení jsou povolena.

QMANAGER

Název správce front, s nímž má být navázáno připojení.

Pokud však vaše aplikace používá tabulku definic kanálů klienta pro připojení ke správci front, přečtěte si téma Použití tabulky definic kanálů klienta s produktem IBM MQ classes for JMS.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, XAConnectionFactory, XAQueueConnection Factory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : QMANAGER

Krátký název nástroje pro administraci JMS : QMGR

Programový přístup

Metody setter/getter

- MQConnectionFactory.setQueueManager ()
- MQConnectionFactory.getQueueManager ()

Hodnoty

"" " (prázdný řetězec)
Výchozí hodnotou může být libovolný řetězec.

QUEUE

Název cíle fronty JMS . Tato volba odpovídá názvu fronty používané správcem front.

Použitelné objekty

Fronta

Dlouhý název nástroje pro administraci JMS : QUEUE

Krátký název nástroje pro administraci JMS : QU

Hodnoty

Libovolný řetězec
Libovolný platný název fronty IBM MQ .

Související odkazy
Pravidla pro pojmenování objektů IBM MQ >

READAHEADALLOWED

Tato vlastnost určuje, zda mohou spotřebitelé zpráv a prohlížeče front používat dopředné čtení k získání přechodných zpráv z tohoto místa určení do interní vyrovnávací paměti před jejich přijetím.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : READAHEADALLOWED

Krátký název nástroje pro administraci JMS : RAALD

Programový přístup

Metody setter/getter

- MQDestination.setReadAheadAllowed()
- MQDestination.getReadAheadAllowed()

Hodnoty

AS_DEST
Určete, zda je dopředné čtení povoleno odkazem na definici fronty nebo tématu. Jedná se o výchozí hodnotu v administrativních nástrojích.

V programech použijte WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_DEST .

AS_Q_DEF
Určete, zda je dopředné čtení povoleno odkazem na definici fronty.

V programech použijte WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_Q_DEF .

AS_TOPIC_DEF
Určete, zda je dopředné čtení povoleno odkazem na definici tématu.

V programech použijte WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_TOPIC_DEF .

NO
Dopředné čtení není povoleno.

V programech použijte WMQConstants.WMQ_READ_AHEAD_ALLOWED_DISABLED .

YES
Dopředné čtení je povoleno.

V programech použijte WMQConstants.WMQ_READ_AHEAD_ALLOWED_ENABLED .

READAHEADCLOSEPOLICY

U zpráv doručených do asynchronního modulu listener pro zprávy se jedná o zprávy v interní vyrovnávací paměti dopředného čtení při zavření spotřebitele zpráv.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : READAHEADCLOSEPOLICY

Krátký název nástroje pro administraci JMS : RACP

Programový přístup

Metody setter/getter

- MQDestination.setReadAheadClosePolicy()
- MQDestination.getReadAheadClosePolicy()

Hodnoty

DORUČENÍ_VŠE
Všechny zprávy v interní vyrovnávací paměti dopředného čtení jsou doručeny modulu listener pro zprávy aplikace před návratem. Jedná se o výchozí hodnotu v administrativních nástrojích.

V programech použijte WMQConstants.WMQ_READ_AHEAD_DELIVERALL .

DORUČENÍ_AKTUÁLNÍ
Před návratem se dokončí pouze aktuální vyvolání modulu listener pro zprávy, což potenciálně zanechá zprávy ve vnitřní vyrovnávací paměti dopředného čtení, které se pak vyřadí.

V programech použijte WMQConstants.WMQ_READ_AHEAD_DELIVERCURRENT .

RECEIVECCSID

Cílová vlastnost, která nastavuje cílový CCSID pro převod zpráv správce front. Hodnota je ignorována, pokud není parametr RECEIVECONVERSION nastaven na hodnotu `WMQ_RECEIVE_CONVERSION_QMGR` .

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : RECEIVECCSID

Krátký název nástroje pro administraci JMS : RCCS

Programový přístup

Metody setter/Getter

- `MQDestination.setReceiveCCSID`
- `MQDestination.getReceiveCCSID`

Hodnoty

WMQConstants.WMQ_RECEIVE_CCSID_JVM_DEFAULT
0 -Použít prostředí JVM `Charset.defaultCharset`

1208
UTF-8

CCSID
Podporovaný identifikátor kódované znakové sady.

RECEIVECONVERSION

Vlastnost místa určení, která určuje, zda má správce front provést převod dat.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : RECEIVECONVERSION

Krátký název nástroje pro administraci JMS : RCNV

Programový přístup

Metody setter/Getter

- `MQDestination.setReceiveConversion`
- `MQDestination.getReceiveConversion`

Hodnoty

WMQConstants.WMQ_RECEIVE_CONVERSION_CLIENT_MSG
1 -provádět převod dat pouze na klientu JMS . Výchozí hodnota až od V7.0a od 7.0.1.5včetně.

WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR
2 -Před odesláním zprávy klientovi proveďte převod dat ve správci front. Výchozí (a pouze) hodnota z V7.0 na V7.0.1.4 včetně, kromě případu, kdy je použita oprava APAR IC72897 .

RECEIVEISOLATION

Tato vlastnost určuje, zda odběratel může přijímat zprávy, které nebyly potvrzeny ve frontě odběratele.

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : RECEIVEISOLATION

Krátký název nástroje pro administraci JMS : RCVISOL

Hodnoty

POTVRZENO
Odběratel přijímá pouze ty zprávy ve frontě odběratele, které byly potvrzeny. Jedná se o výchozí hodnotu v administrativních nástrojích.

V programech použijte WMQConstants.WMQ_RCVISOL_COMMITTED .

NEPOTVRZENO
Odběratel může přijímat zprávy, které nebyly potvrzeny ve frontě odběratele.

V programech použijte WMQConstants.WMQ_RCVISOL_UNCOMMITTED .

RECEXIT

Identifikuje uživatelskou proceduru pro příjem kanálu nebo posloupnost uživatelských procedur pro příjem, které mají být spuštěny v posloupnosti.

Aby mohl server IBM MQ classes for JMS vyhledat uživatelské procedury pro příjem, může být vyžadována další konfigurace. Další informace naleznete v tématu Konfigurace tříd IBM MQ pro platformu JMS pro použití uživatelských procedur kanálu.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : RECEXIT

Krátký název nástroje pro administraci JMS : RCX

Programový přístup

Metody setter/getter

- MQConnectionFactory.setReceiveExit ()
- MQConnectionFactory.getReceiveExit ()

Hodnoty

- `null`. Toto je výchozí hodnota.
- Řetězec obsahující jednu nebo více položek oddělených čárkami, kde každá položka je buď:
  - Název třídy, která implementuje rozhraní `WMQReceiveExit` (pro uživatelskou proceduru pro příjem kanálu zapsanou v souboru Java).
  - Řetězec ve formátu *libraryName(entryPointName)* (pro uživatelskou proceduru pro příjem kanálu, která není zapsána v adresáři Java).

RECEXITINIT

Uživatelská data, která jsou při volání předána uživatelským procedur pro příjem kanálu.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : RECEXITINIT

Krátký název nástroje pro administraci JMS : RCXI

Programový přístup

Metody setter/getter

- MQConnectionFactory.setReceiveExitInit()
- MQConnectionFactory.getReceiveExitInit()

Hodnoty

null
Řetězec obsahující jednu nebo více položek uživatelských dat oddělených čárkami. Toto je výchozí hodnota.

REPLYTOSTYLE

Určuje, jak je sestaveno pole JMSReplyTo v přijaté zprávě.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : REPLYTOSTYLE

Krátký název nástroje pro administraci JMS : RTOST

Programový přístup

Metody setter/getter

- MQConnectionFactory.setReplyToStyle()
- MQConnectionFactory.getReplyToStyle()

Hodnoty

DEFAULT
Ekvivalentní příkazu `MQMD`.

RFH2
Použijte hodnotu zadanou v záhlaví RFH2 . Pokud byla v odesílající aplikaci nastavena hodnota JMSReplyTo , použijte tuto hodnotu.

MQMD
Použijte zadanou hodnotu MQMD. Toto chování je ekvivalentní výchozímu chování IBM WebSphere MQ 6.0.2.4 a 6.0.2.5.

Pokud hodnota JMSReplyTo nastavená odesílající aplikací neobsahuje název správce front, přijímající správce front vloží do deskriptoru MQMD svůj vlastní název. Nastavíte-li tento parametr na hodnotu `MQMD`, bude fronta pro odpověď použita v přijímajícím správci front. Nastavíte-li tento parametr na hodnotu RFH2, fronta pro odpověď, kterou používáte, se bude nacházet ve správci front určeném v RFH2 odeslané zprávy, jak bylo původně nastaveno odesílající aplikací.

Pokud hodnota JMSReplyTo nastavená odesílající aplikací obsahuje název správce front, hodnota tohoto parametru není důležitá, protože MQMD i RFH2 obsahují stejnou hodnotu.

RESCANINT

Pokud spotřebitel zpráv v doméně typu point-to-point používá selektor zpráv k výběru zpráv, které chce přijmout, IBM MQ classes for JMS vyhledejte ve frontě IBM MQ vhodné zprávy v pořadí určeném atributem `MsgDeliverySequence` fronty.

Poté, co IBM MQ classes for JMS najde vhodnou zprávu a doručí ji spotřebiteli, IBM MQ classes for JMS pokračujte v hledání další vhodné zprávy z její aktuální pozice ve frontě. Produkt IBM MQ classes for JMS pokračuje v prohledávání fronty tímto způsobem, dokud nedosáhne konce fronty nebo dokud nevyprší časový interval v milisekundách určený hodnotou této vlastnosti. V každém případě se produkt IBM MQ classes for JMS vrátí na začátek fronty, aby mohl pokračovat v hledání, a začne nový časový interval.

Použitelné objekty

ConnectionFactory, QueueConnectiontovárna, XAConnectionFactory, XAQueueConnectiontovárna

Dlouhý název nástroje pro administraci JMS : RESCANINT

Krátký název nástroje pro administraci JMS : RINT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setRescanInterval ()
- MQConnectionFactory.getRescanInterval ()

Hodnoty

5000
Výchozí hodnotou může být libovolné kladné celé číslo.

SECEXIT

Identifikuje uživatelskou proceduru pro zabezpečení zprávy kanálu.

Může být požadována další konfigurace, aby IBM MQ classes for JMS vyhledala uživatelské procedury zabezpečení. Další informace naleznete v tématu Konfigurace tříd IBM MQ pro platformu JMS pro použití uživatelských procedur kanálu.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SECEXIT

Krátký název nástroje pro administraci JMS : SXC

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSecurityExit ()
- MQConnectionFactory.getSecurityExit ()

Hodnoty

- `null`. Toto je výchozí hodnota.
- Řetězec obsahující jednu nebo více položek oddělených čárkami, kde každá položka je buď:
  - Název třídy, která implementuje rozhraní `WMQSecurityExit` (pro uživatelskou proceduru pro zabezpečení zprávy kanálu napsanou v souboru Java).
  - Řetězec ve formátu *libraryName(entryPointName)* (pro uživatelskou proceduru zabezpečení kanálu, která není zapsána v adresáři Java).

SECEXITINIT

Uživatelská data, která jsou předána uživatelské proceduře pro zabezpečení zprávy kanálu při volání.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SECEXITINIT

Krátký název nástroje pro administraci JMS : SCXI

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSecurityExitInit()
- MQConnectionFactory.getSecurityExitInit()

Hodnoty

null
Výchozí hodnotou může být libovolný řetězec.

SENDCHECKCOUNT

Počet volání odeslání, která mají být povolena mezi kontrolou asynchronních chyb vložení v rámci jedné netransakční relace JMS .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SENDCHECKCOUNT

Krátký název nástroje pro administraci JMS : SCC

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSendCheckCount()
- MQConnectionFactory.getSendCheckCount()

Hodnoty

null
Výchozí hodnotou může být libovolný řetězec.

SENDEXIT

Identifikuje uživatelskou proceduru odeslání kanálu nebo posloupnost uživatelských procedur odeslání, které mají být spuštěny v posloupnosti.

Může být požadována další konfigurace, aby IBM MQ classes for JMS vyhledal uživatelské procedury odeslání. Další informace naleznete v tématu Konfigurace tříd IBM MQ pro platformu JMS pro použití uživatelských procedur kanálu.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SENDEXIT

Krátký název nástroje pro administraci JMS : SDX

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSendExit ()
- MQConnectionFactory.getSendExit ()

Hodnoty

- `null`. Toto je výchozí hodnota.
- Řetězec obsahující jednu nebo více položek oddělených čárkami, kde každá položka je buď:
  - Název třídy, která implementuje rozhraní `WMQSendExit` (pro uživatelskou proceduru odeslání zprávy kanálu napsanou v souboru Java).
  - Řetězec ve formátu *libraryName(entryPoint)* (pro uživatelskou proceduru odeslání zprávy kanálu, která není zapsána v produktu Java).

SENDEXITINIT

Uživatelská data, která jsou předána uživatelským procedurám pro odeslání zprávy kanálu při volání.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SENDEXITINIT

Krátký název nástroje pro administraci JMS : SDXI

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSendExitInit()
- MQConnectionFactory.getSendExitInit()

Hodnoty

null
Výchozí hodnotou může být libovolný řetězec obsahující jednu nebo více položek uživatelských dat oddělených čárkami.

SHARECONVALLOWED

U aplikací, které používají normální režim nebo normální režim poskytovatele systému zpráv IBM MQ s omezeními, tato vlastnost určuje, zda se funkce sdílení konverzací používá pro připojení, relace a kontexty produktu JMS vytvořené z továrny připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SHARECONVALLOWED

Krátký název nástroje pro administraci JMS : SCALD

Programový přístup

Metody setter/getter

- MQConnectionFactory.setShareConvAllowed()
- MQConnectionFactory.getShareConvAllowed()

Hodnoty

YES
JMS připojení, relace a kontexty vytvořené z továrny připojení v rámci stejného prostředí JVM mohou v případě potřeby sdílet instanci kanálu (která se mapuje na připojení TCP/IP).

Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_SHARE_CONV_ALLOWED_YES.

NO
Každé připojení JMS vytvořené z továrny připojení a každá relace JMS vytvořená z těchto připojení JMSmá vlastní instanci kanálu (připojení TCP/IP) ke správci front.

Pro kontexty JMS první kontext vytvořený z továrny připojení vytváří dvě instance kanálu (připojení TCP/IP). Další kontexty JMS vytvořené z prvního kontextu mají svou vlastní instanci kanálu (připojení TCP/IP).

Pro programy použijte WMQConstants.WMQ_SHARE_CONV_ALLOWED_NO.

Související pojmy
IBM MQ provozní režimy poskytovatele systému zpráv
Sdílení připojení TCP/IP ve třídách IBM MQ pro JMS

SPARSESUBS

Řídí zásadu načítání zpráv objektu TopicSubscriber .

Použitelné objekty

ConnectionFactory, TopicConnection, továrna

Dlouhý název nástroje pro administraci JMS : SPARSESUBS

Krátký název nástroje pro administraci JMS : SSUBS

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSparseOdběry ()
- MQConnectionFactory.getSparseOdběry ()

Hodnoty

NO
Odběry přijímají často odpovídající zprávy. Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte hodnotu false.

YES
Odběry přijímají zřídka odpovídající zprávy. Tato hodnota vyžaduje, aby frontu odběrů bylo možné otevřít pro procházení.

Pro programy použijte hodnotu true.

SSLCIPHERSUITE

CipherSuite , která má být použita pro připojení TLS.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SSLCIPHERSUITE

Krátký název nástroje pro administraci JMS : SCPHS

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSSLCipherSuite ()
- MQConnectionFactory.getSSLCipherSuite ()

Hodnoty

null
Toto je výchozí hodnota. Další informace viz Vlastnosti TLS JMS objektů.

SSLCRL

Servery CRL pro kontrolu odvolání certifikátu TLS.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SSLCRL

Krátký název nástroje pro administraci JMS : SCRL

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSSLCertStores ()
- MQConnectionFactory.getSSLCertStores ()

Hodnoty

null
Seznam adres URL LDAP oddělených mezerami. Toto je výchozí hodnota. Další informace viz Vlastnosti TLS JMS objektů.

SSLFIPSREQUIRED

Tato vlastnost určuje, zda připojení TLS musí používat sadu CipherSuite , kterou podporuje poskytovatel JSSE FIPS produktu IBM Java (IBMJSSEFIPS).

Poznámka: V systému AIX, Linux, and Windowsposkytuje produkt IBM MQ kompatibilitu se standardem FIPS 140-2 prostřednictvím šifrovacího modulu IBM Crypto for C (ICC) . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C (ICC) certifikát a měli by si být vědomi všech doporučení poskytnutých NIST. Náhradní modul FIPS 140-3 momentálně probíhá a jeho stav lze zobrazit jeho vyhledáním v modulech NIST CMVP v seznamu procesů.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SSLFIPSREQUIRED

Krátký název nástroje pro administraci JMS : SFIPS

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSSLFipsRequired ()
- MQConnectionFactory.getSSLFipsRequired ()

Hodnoty

NO
Připojení TLS může používat libovolnou sadu CipherSuite , která není podporována poskytovatelem FIPS JSSE IBM Java (IBMJSSEFIPS).

Toto je výchozí hodnota. V programech použijte hodnotu false.

YES
Připojení TLS musí používat sadu CipherSuite , kterou podporuje produkt IBMJSSEFIPS.

V programech použijte hodnotu true.

SSLPEERNAME

V případě protokolu TLS se jedná o kostru *rozlišujícího názvu* , která se musí shodovat s kostrou poskytnutým správcem front.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SSLPEERNAME

Krátký název nástroje pro administraci JMS : SPEER

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSSLPeerName ()
- MQConnectionFactory.getSSLPeerName ()

Hodnoty

null
Toto je výchozí hodnota. Další informace viz Vlastnosti TLS JMS objektů.

SSLRESETCOUNT

Pro TLS-celkový počet bajtů odeslaných a přijatých připojením, než bude znovu vyjednán tajný klíč, který se používá pro šifrování.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SSLRESETCOUNT

Krátký název nástroje pro administraci JMS : SRC

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSSLResetCount ()
- MQConnectionFactory.getSSLResetCount ()

Hodnoty

0
Nula nebo jakékoli kladné celé číslo menší nebo rovno 999, 999, 999. Toto je výchozí hodnota. Další informace viz Vlastnosti TLS JMS objektů.

STATREFRESHINT

Interval (v milisekundách) mezi aktualizacemi dlouho běžící transakce, která zjistí, když odběratel ztratí připojení ke správci front.

Tato vlastnost je relevantní pouze v případě, že má SUBSTORE hodnotu QUEUE.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : STATREFRESHINT

Krátký název nástroje pro administraci JMS : SRI

Programový přístup

Metody setter/getter

- MQConnectionFactory.setStatusRefreshInterval()
- MQConnectionFactory.getStatusRefreshInterval()

Hodnoty

60000
Výchozí hodnotou může být libovolné kladné celé číslo. Další informace viz Vlastnosti TLS JMS objektů.

SUBSTORE

Kde produkt IBM MQ classes for JMS ukládá trvalá data související s aktivními odběry.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SUBSTORE

Krátký název nástroje pro administraci JMS : SS

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSubscriptionStore ()
- MQConnectionFactory.getSubscriptionStore ()

Hodnoty

BROKER
K uchování podrobností o odběrech použijte úložiště odběrů založené na zprostředkovateli. Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_SUBSTORE_BROKER.

MIGRATE
Přeneste informace o odběru z úložiště odběrů založeného na frontě do úložiště odběrů založeného na zprostředkovateli.

Pro programy použijte WMQConstants.WMQ_SUBSTORE_MIGRATE.

QUEUE
K uchování podrobností o odběrech použijte úložiště odběrů založené na frontě.

Pro programy použijte WMQConstants.WMQ_SUBSTORE_QUEUE.

SYNCPOINTALLGETS

Tato vlastnost určuje, zda mají být všechny operace get prováděny pod synchronizačním bodem.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : SYNCPOINTALLGETS

Krátký název nástroje pro administraci JMS : SPAG

Programový přístup

Metody setter/getter

- MQConnectionFactory.setSyncpointAllGets()
- MQConnectionFactory.getSyncpointAllGets()

Hodnoty

Ne
Toto je výchozí hodnota.

Ano

TARGCLIENT

Tato vlastnost určuje, zda se formát IBM MQ RFH2 používá k výměně informací s cílovými aplikacemi.

Použitelné objekty

Fronta, téma

Dlouhý název nástroje pro administraci JMS : TARGCLIENT

Krátký název nástroje pro administraci JMS : TC

Programový přístup

Metody setter/getter

- MQDestination.setTargetClient()
- MQDestination.getTargetClient()

Hodnoty

JMS
Cílem zprávy je aplikace JMS . Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_CLIENT_JMS_COMPLIANT.

MQ
Cílem zprávy je jiná aplikace nežJMS IBM MQ .

Pro programy použijte WMQConstants.WMQ_CLIENT_NONJMS_MQ.

TARGCLIENTMATCHING

Tato vlastnost určuje, zda má zpráva odpovědi odeslaná do fronty určené polem záhlaví JMSReplyTo příchozí zprávy záhlaví MQRFH2 pouze v případě, že příchozí zpráva má záhlaví MQRFH2 .

Použitelné objekty

ConnectionFactory, QueueConnectiontovárna, XAConnectionFactory, XAQueueConnectiontovárna

Dlouhý název nástroje pro administraci JMS : TARGCLIENTMATCHING

Krátký název nástroje pro administraci JMS : TCM

Programový přístup

Metody setter/getter

- MQConnectionFactory.setTargetClientMatching()
- MQConnectionFactory.getTargetClientMatching()

Hodnoty

YES
Pokud příchozí zpráva nemá záhlaví MQRFH2 , vlastnost TARGCLIENT objektu fronty odvozenou z pole záhlaví JMSReplyTo zprávy se odešle do produktu MQ. Pokud zpráva obsahuje záhlaví MQRFH2 , je vlastnost TARGCLIENT nastavena na hodnotu JMS . Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte hodnotu true.

NO
Vlastnost TARGCLIENT objektu fronty odvozená z pole záhlaví JMSReplyTo příchozí zprávy je vždy nastavena na hodnotu JMS.

Pro programy použijte hodnotu false.

TEMPMODEL

Název modelové fronty, ze které jsou vytvořeny dočasné fronty JMS .

Použitelné objekty

ConnectionFactory, QueueConnectiontovárna, XAConnectionFactory, XAQueueConnectiontovárna

Dlouhý název nástroje pro administraci JMS : TEMPMODEL

Krátký název nástroje pro administraci JMS : TM

Programový přístup

Metody setter/getter

- MQConnectionFactory.setTemporaryModel ()
- MQConnectionFactory.getTemporaryModel ()

Hodnoty

SYSTEM.DEFAULT.MODEL.QUEUE
Výchozí hodnotou může být libovolný řetězec.

TEMPQPREFIX

Předpona, která se používá k vytvoření názvu dynamické fronty IBM MQ .

Použitelné objekty

ConnectionFactory, QueueConnectiontovárna, XAConnectionFactory, XAQueueConnectiontovárna

Dlouhý název nástroje pro administraci JMS : TEMPQPREFIX

Krátký název nástroje pro administraci JMS : TQP

Programový přístup

Metody setter/getter

- MQConnectionFactory.setTempQPrefix ()
- MQConnectionFactory.getTempQPrefix ()

Hodnoty

'' '' (prázdný řetězec)
Použitá předpona je `CSQ.*` na z/OS a `AMQ.*` na všech ostatních platformách. Jedná se o výchozí hodnoty.

předpona fronty
Předpona fronty je libovolný řetězec, který odpovídá pravidlům pro vytváření obsahu pole *DynamicQName* v deskriptoru objektu IBM MQ (struktura MQOD), ale poslední neprázdný znak musí být hvězdička.

TEMPTOPICPREFIX

Při vytváření dočasných témat produkt JMS vygeneruje řetězec tématu ve tvaru '' TEMP */ TEMPTOPICPREFIX/unique_id* '', nebo pokud je tato vlastnost ponechána s výchozí hodnotou, pouze '' TEMP */unique_id* ''. Zadání neprázdné předpony TEMPTOPICPREFIX umožňuje definovat specifické modelové fronty pro vytváření spravovaných front pro odběratele dočasných témat vytvořených v rámci tohoto připojení.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : TEMPTOPICPREFIX

Krátký název nástroje pro administraci JMS : TTP

Programový přístup

Metody setter/getter

- MQConnectionFactory.setTempTopicPrefix()
- MQConnectionFactory.getTempTopicPrefix()

Hodnoty

Jakýkoli nenulový řetězec sestávající pouze z platných znaků pro řetězec tématu IBM MQ . Výchozí hodnota je '' '' (prázdný řetězec).

TOPIC

Název místa určení tématu JMS . Tuto hodnotu používá správce front jako řetězec tématu publikování nebo odběru.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci JMS : TOPIC

Krátký název nástroje pro administraci JMS : TOP

Hodnoty

Libovolný řetězec
Řetězec, který tvoří platný řetězec tématu IBM MQ . Používáte-li produkt IBM MQ jako poskytovatele systému zpráv s produktem WebSphere Application Server, zadejte hodnotu, která odpovídá názvu, pod kterým je téma známé pro administrativní účely v rámci produktu WebSphere Application Server.

Související odkazy
Řetězce tématu

TRANSPORT

Povaha připojení ke správci front nebo zprostředkovateli.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : TRANSPORT

Krátký název nástroje pro administraci JMS : TRAN

Programový přístup

Metody setter/getter

- MQConnectionFactory.setTransportTyp ()
- MQConnectionFactory.getTransportTyp ()

Hodnoty

BIND
Pro připojení ke správci front v režimu vazeb. Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_CM_BINDINGS.

CLIENT
Pro připojení ke správci front v režimu klienta.

Pro programy použijte WMQConstants.WMQ_CM_CLIENT.

Přímý
Pro připojení v reálném čase ke zprostředkovateli, který nepoužívá tunelové propojení HTTP .

Pro programy použijte WMQConstants.WMQ_CM_DIRECT_TCPIP.

DIRECTHTTP
Pro připojení v reálném čase ke zprostředkovateli pomocí tunelového propojení HTTP . Podporován je pouze protokol HTTP 1.0 .

Pro programy použijte WMQConstants.WMQ_CM_DIRECT_HTTP.

Související pojmy

“Závislosti mezi vlastnostmi objektů IBM MQ classes for JMS” na stránce 1891
Platnost některých vlastností závisí na konkrétních hodnotách jiných vlastností.

WILDCARDFORMAT

Tato vlastnost určuje, která verze syntaxe zástupných znaků se má použít.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS : WILDCARDFORMAT

Krátký název nástroje pro administraci JMS : WCFMT

Programový přístup

Metody setter/getter

- MQConnectionFactory.setWildCardFormat()
- MQConnectionFactory.getWildCardFormat()

Hodnoty

TOPIC_ONLY
Rozpozná pouze zástupné znaky na úrovni tématu, které se používají ve zprostředkovateli verze 2. Toto je výchozí hodnota pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_WILDCARD_TOPIC_ONLY.

POUZE_V_SOUBORU
Rozpozná pouze zástupné znaky znaků, které se používají ve verzi zprostředkovatele 1.

Pro programy použijte WMQConstants.WMQ_WILDCARD_CHAR_ONLY.

Vlastnost ENCODING

Vlastnost ENCODING se skládá ze tří dílčích vlastností, a to ve dvanácti možných kombinacích.

Platné hodnoty, které může vlastnost `ENCODING` převzít, jsou sestaveny ze tří dílčích vlastností:

Kódování celých čísel
Buď normální, nebo obrácené

Kódování desetinných čísel
Buď normální, nebo obrácené

kódování s pohyblivou řádovou čárkou
IEEE normální, IEEE obrácené nebo z/OS

Vlastnost `ENCODING` je vyjádřena jako tříznakový řetězec s následující syntaxí:

```
{N|R}{N|R}{N|R|3}
```

V tomto řetězci:

- N označuje normální
- R označuje obrácené
- 3 označuje z/OS
- První znak představuje *kódování celých čísel*
- Druhý znak představuje *kódování desetinných čísel* .
- Třetí znak představuje *kódování s pohyblivou řádovou čárkou*

To poskytuje sadu dvanácti možných hodnot pro vlastnost ENCODING .

Existuje další hodnota, řetězec NATIVE, který nastavuje odpovídající hodnoty kódování pro platformu Java .

Následující příklady ukazují platné kombinace pro ENCODING:

```
ENCODING(NNR)
ENCODING(NATIVE)
ENCODING(RR3)
```

Vlastnosti TLS objektů JMS

Povolte šifrování TLS (Transport Layer Security) pomocí vlastnosti SSLCIPHERSUITE. Poté můžete změnit charakteristiky šifrování TLS pomocí několika dalších vlastností.

Zadáte-li volbu TRANSPORT (CLIENT), můžete povolit šifrovanou komunikaci TLS pomocí vlastnosti SSLCIPHERSUITE. Nastavte tuto vlastnost na platnou sadu CipherSuite poskytovanou vaším poskytovatelem JSSE. Musí odpovídat specifikaci CipherSpec pojmenované v kanálu SVRCONN pojmenovaném vlastností CHANNEL.

Specifikace CipherSpecs (určené v kanálu SVRCONN) a CipherSuites (určené v objektech ConnectionFactory) však používají k reprezentaci stejných šifrovacích algoritmů TLS různá schémata pojmenování. Pokud je ve vlastnosti SSLCIPHERSUITE zadán rozpoznaný název CipherSpec , správce JMSAdmin vydá varování a namapuje specifikaci CipherSpec na ekvivalentní sadu CipherSuite. Seznam CipherSpecs rozpoznaných IBM MQ a JMSAdmin viz TLS CipherSpecs a CipherSuites v části IBM MQ classes for JMS .

Pokud požadujete připojení k použití CipherSuite , která je podporována poskytovatelem JSSE FIPS produktu IBM Java (IBMJSSEFIPS), nastavte vlastnost SSLFIPSREQUIRED továrny připojení na hodnotu YES. Výchozí hodnota této vlastnosti je NO, což znamená, že připojení může používat libovolnou podporovanou sadu CipherSuite. Vlastnost je ignorována, pokud není nastaven parametr SSLCIPHERSUITE.

Hodnota SSLPEERNAME odpovídá formátu parametru SSLPEER, který lze nastavit v definicích kanálů. Jedná se o seznam dvojic název-hodnota atributu oddělených čárkami nebo středníky. Příklad:

```
SSLPEERNAME(CN=QMGR.*, OU=IBM, OU=WEBSPHERE)
```

Sada názvů a hodnot tvoří *rozlišující název*. Další podrobnosti o rozlišujících názvech a jejich použití s produktem IBM MQviz Zabezpečení IBM MQ.

Uvedený příklad zkontroluje identifikační certifikát předložený serverem v době připojení. Aby bylo připojení úspěšné, musí mít certifikát obecný název začínající QMGR., a musí mít alespoň dva názvy organizačních jednotek, z nichž první je IBM a druhý WEBSPHERE. Kontrola nerozlišuje velká a malá písmena.

Není-li parametr SSLPEERNAME nastaven, žádná taková kontrola se neprovede. Není-li parametr SSLCIPHERSUITE nastaven, bude parametr SSLPEERNAME ignorován.

Vlastnost SSLCRL určuje žádný nebo více serverů CRL (Certificate Revocation List). Použití této vlastnosti vyžaduje prostředí JVM v Java 2 v1.4. Toto je seznam položek formuláře oddělených mezerami:

```
ldap:// hostname:[ port ]
```

volitelně následováno jedním/. Pokud je parametr *port* vynechán, předpokládá se výchozí port LDAP 389. V době připojení je certifikát TLS předložený serverem kontrolován proti uvedeným serverům CRL. Další informace o zabezpečení CRL naleznete v části Zabezpečení IBM MQ .

Pokud není nastaven SSLCRL, žádná taková kontrola se neprovede. Není-li parametr SSLCIPHERSUITE nastaven, je parametr SSLCRL ignorován.

Vlastnost SSLRESETCOUNT představuje celkový počet bajtů odeslaných a přijatých připojením, než bude znovu vyjednán tajný klíč použitý pro šifrování. Počet odeslaných bajtů je číslo před šifrováním a počet přijatých bajtů je číslo po dešifrování. Počet bajtů také zahrnuje řídicí informace odeslané a přijaté produktem IBM MQ classes for JMS.

Chcete-li například konfigurovat objekt ConnectionFactory , který lze použít k vytvoření připojení prostřednictvím kanálu MQI s povoleným zabezpečením TLS s tajným klíčem, který je znovu vyjednán po toku 4 MB dat, zadejte pro správce JMSAdmin následující příkaz:

```
ALTER CF(my.cf) SSLRESETCOUNT(4194304)
```

Je-li hodnota SSLRESETCOUNT nula, což je výchozí hodnota, tajný klíč se nikdy znovu nevyjednává. Vlastnost SSLRESETCOUNT je ignorována, pokud není nastaveno SSLCIPHERSUITE.

Odkaz na IBM MQ Message Service Client (XMS) for .NET

Tento referenční oddíl poskytuje informace o rozhraních tříd IBM MQ Message Service Client (XMS) for .NET (XMS .NET) a o vlastnostech objektů definovaných XMS.

.NETRozhraní

Tento oddíl popisuje rozhraní třídy .NET a jejich vlastnosti a metody.

Následující tabulka shrnuje rozhraní, která jsou definována v oboru názvů `IBM.XMS` .

Tabulka 871. Souhrn rozhraní třídy .NET

Rozhraní	**Popis**
“IBytesMessage” na stránce 1946	Bajtová zpráva je zpráva, jejíž tělo se skládá z proudu bajtů.
“IPřipojení” na stránce 1956	Objekt připojení představuje aktivní připojení aplikace k serveru systému zpráv.
“IConnectionFactory” na stránce 1958	Aplikace používá továrnu připojení k vytvoření připojení.
“IConnectionMetaData” na stránce 1960	Objekt ConnectionMetaData poskytuje informace o připojení.
“ICíl” na stránce 1960	Místo určení je místo, kam aplikace odesílá zprávy, nebo je to zdroj, ze kterého aplikace přijímá zprávy, nebo obojí.
“ExceptionListener” na stránce 1961	Aplikace používá modul listener pro výjimky, aby byla asynchronně upozorněna na problém s připojením.
“Výjimka IllegalState” na stránce 1962	XMS vyvolá tuto výjimku, pokud aplikace volá metodu v chybném nebo nevhodném čase, nebo pokud XMS není v odpovídajícím stavu pro požadovanou operaci.
“InitialContext” na stránce 1962	Aplikace používá objekt InitialContext k vytvoření objektů z definic objektů, které jsou načteny z úložiště spravovaných objektů.
“InvalidClientIDException” na stránce 1964	XMS vyvolá tuto výjimku, pokud se aplikace pokusí nastavit identifikátor klienta pro připojení, ale identifikátor klienta není platný nebo se již používá.

Tabulka 871. Souhrn rozhraní třídy .NET (pokračování)	
Rozhraní	**Popis**
“Výjimka InvalidDestination” na stránce 1964	XMS vyvolá tuto výjimku, pokud aplikace uvede místo určení, které není platné.
“Výjimka InvalidSelector” na stránce 1965	XMS vyvolá tuto výjimku, pokud aplikace poskytne výraz selektoru zpráv, jehož syntaxe je neplatná.
“IMapMessage” na stránce 1965	Zpráva mapy je zpráva, jejíž tělo se skládá ze sady dvojic název-hodnota, kde každá hodnota má přidružený datový typ.
“Zpráva IMessage” na stránce 1974	Objekt zprávy představuje zprávu, kterou aplikace odesílá nebo přijímá. IMessage je supertřída pro třídy zpráv, jako např. IMapMessage.
“IMessageConsumer” na stránce 1979	Aplikace používá spotřebitele zpráv k přijímání zpráv odeslaných do místa určení.
“MessageEOFException” na stránce 1982	XMS vyvolá tuto výjimku, pokud XMS zjistí konec proudu bajtových zpráv, když aplikace čte tělo bajtové zprávy.
“Výjimka MessageFormat” na stránce 1982	XMS vyvolá tuto výjimku, pokud XMS zjistí zprávu s neplatným formátem.
“IMessageListener (delegát)” na stránce 1982	Aplikace používá modul listener pro zprávy k asynchronnímu příjmu zpráv.
“MessageNotReadableException” na stránce 1983	XMS vyvolá tuto výjimku, pokud se aplikace pokusí přečíst tělo zprávy, které je pouze pro zápis.
“MessageNotWritableException” na stránce 1983	XMS vyvolá tuto výjimku, pokud se aplikace pokusí zapsat do těla zprávy, která je jen pro čtení.
“IMessageProducer” na stránce 1983	Aplikace používá producenta zpráv k odesílání zpráv do místa určení.
“IObjectMessage” na stránce 1989	Zpráva objektu je zpráva, jejíž tělo se skládá ze serializovaného objektu Java nebo .NET .
“IPropertyContext” na stránce 1990	IPropertyContext je abstraktní supertřída, která obsahuje metody, které získají a nastaví vlastnosti. Tyto metody jsou zděděny jinými třídami.
“IQueueBrowser” na stránce 1998	Aplikace používá prohlížeč front k procházení zpráv ve frontě bez jejich odebrání.
“Žadatel” na stránce 2000	Aplikace používá žadatele k odeslání zprávy požadavku a poté čeká na odpověď a obdrží ji.
“Výjimka ResourceAllocation” na stránce 2001	XMS vyvolá tuto výjimku, pokud XMS nemůže přidělit prostředky požadované metodou.
“SecurityException” na stránce 2002	XMS vyvolá tuto výjimku, pokud je identifikátor uživatele a heslo poskytnuté pro ověření aplikace odmítnuto. XMS také vyvolá tuto výjimku, pokud kontrola oprávnění selže a zabrání dokončení metody.
“IRelace” na stránce 2002	Relace je kontext s jedním podprocesem pro odesílání a příjem zpráv.

Tabulka 871. Souhrn rozhraní třídy .NET (pokračování)	
Rozhraní	**Popis**
“IStreamMessage” na stránce 2012	Zpráva proudu je zpráva, jejíž tělo se skládá z proudu hodnot, kde každá hodnota má přidružený datový typ.
“ITextMessage” na stránce 2021	Textová zpráva je zpráva, jejíž tělo obsahuje řetězec.
“TransactionInProgressException” na stránce 2022	XMS vyvolá tuto výjimku, pokud aplikace požaduje operaci, která není platná, protože probíhá transakce.
“TransactionRolledBackException” na stránce 2022	XMS vyvolá tuto výjimku, pokud aplikace volá funkci Session.commit() k potvrzení aktuální transakce, ale transakce je poté odvolána.
XMSC	Pro .NETjsou XMS názvy a hodnoty vlastností definovány v této třídě jako veřejné konstanty. Další podrobnosti viz “Vlastnosti objektů XMS” na stránce 2025.
“Výjimka XMSException” na stránce 2022	Pokud produkt XMS zjistí chybu při zpracování volání metody .NET , XMS vygeneruje výjimku. Výjimkou je objekt, který zapouzdřuje informace o chybě. Existují různé typy výjimky XMS a objekt XMSException je pouze jedním typem výjimky. Třída XMSException je však supertřídou ostatních tříd výjimek XMS . XMS vyvolá objekt XMSException v situacích, kdy žádný z ostatních typů výjimek není vhodný.
“XMSFactoryFactory” na stránce 2023	Pokud aplikace nepoužívá spravované objekty, použijte tuto třídu k vytvoření továren připojení, front a témat.

Definice každé metody vypisuje kódy výjimek, které může produkt XMS vrátit, pokud zjistí chybu při zpracování volání metody. Každý kód výjimky je reprezentován pojmenovanou konstantou, která má odpovídající výjimku.

IBytesMessage

Bajtová zpráva je zpráva, jejíž tělo se skládá z proudu bajtů.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IBytesMessage
```

.NET vlastnosti

BodyLength -Získat délku těla

Rozhraní:

```
Int64 BodyLength
    {
      get;
    }
```

Získat délku těla zprávy v bajtech, když je tělo zprávy jen pro čtení.

Vrácená hodnota je délka celého těla bez ohledu na to, kde je momentálně umístěn kurzor pro čtení zprávy.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException

metody

ReadBoolean -Čtení logické hodnoty

Rozhraní:

```
Boolean ReadBoolean();
```

Přečte logickou hodnotu z proudu zpráv v bajtech.

Parametry:
Není

Vrací:
Logická hodnota, která se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadSignedbajt-bajt pro čtení

Rozhraní:

```
Int16   ReadSignedByte();
```

Přečtěte další bajt z bajtového proudu zpráv jako podepsané 8bitové celé číslo.

Parametry:
Není

Vrací:
Bajt, který se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadBytes -Načtené bajty

Rozhraní:

```
Int32   ReadBytes(Byte[] array);
Int32   ReadBytes(Byte[] array, Int32 length);
```

Čte pole bajtů z proudu zpráv bajtů počínaje aktuální pozicí kurzoru.

Parametry:

pole (výstup)
Vyrovnávací paměť, která má obsahovat pole přečtených bajtů. Pokud je počet zbývajících bajtů, které mají být načteny z proudu před voláním, větší nebo roven délce vyrovnávací paměti, je vyrovnávací paměť vyplněna. Jinak je vyrovnávací paměť částečně vyplněna všemi zbývajícími bajty.

Zadáte-li na vstupu ukazatel null, metoda přeskočí bajty bez jejich čtení. Pokud zbývající počet bajtů, které mají být načteny z proudu před voláním, je větší nebo roven délce vyrovnávací paměti, počet přeskočených bajtů se rovná délce vyrovnávací paměti. Jinak budou všechny zbývající bajty vynechány. Kurzor zůstává na další pozici pro čtení v proudu bajtových zpráv.

délka (vstup)
Délka vyrovnávací paměti v bajtech

Vrací:
Počet bajtů přečtených do vyrovnávací paměti. Pokud je vyrovnávací paměť částečně vyplněna, hodnota je menší než délka vyrovnávací paměti, což znamená, že zbývají žádné další bajty ke čtení. Pokud z proudu před voláním nezbývají žádné bajty ke čtení, hodnota je `XMSC_END_OF_STREAM`.

Zadáte-li na vstupu ukazatel s hodnotou Null, metoda nevrátí žádnou hodnotu.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException

ReadChar -Přečtené znaky

Rozhraní:

```
Char    ReadChar();
```

Přečtěte další 2 bajty z proudu zpráv bajtů jako znak.

Parametry:
Není

Vrací:
Znak, který se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadDouble -Přečtěte si číslo s pohyblivou řádovou čárkou s dvojitou přesností.

Rozhraní:

```
Double  ReadDouble();
```

Přečtěte dalších 8 bajtů z proudu zpráv bajtů jako číslo s pohyblivou řádovou čárkou s dvojitou přesností.

Parametry:
Není

Vrací:
Přečtené číslo s pohyblivou řádovou čárkou s dvojitou přesností.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadFloat -Číslo pohyblivé řádové čárky čtení

Rozhraní:

```
Single  ReadFloat();
```

Čte další 4 bajty z proudu bajtů zpráv jako číslo s pohyblivou řádovou čárkou.

Parametry:
Není

Vrací:
Číslo s pohyblivou řádovou čárkou, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadInt -Číst celé číslo

Rozhraní:

```
Int32   ReadInt();
```

Přečtěte si další 4 bajty z proudu zpráv bajtů jako 32bitové celé číslo se znaménkem.

Parametry:
Není

Vrací:
Celé číslo, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadLong -Číst dlouhé celé číslo

Rozhraní:

```
Int64   ReadLong();
```

Přečtěte dalších 8 bajtů z proudu zpráv bajtů jako 64bitové celé číslo se znaménkem.

Parametry:
Není

Vrací:
Dlouhé celé číslo, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadShort -Číst krátké celé číslo

Rozhraní:

```
Int16   ReadShort();
```

Přečtěte si další 2 bajty z proudu zpráv bajtů jako 16bitové celé číslo se znaménkem.

Parametry:
Není

Vrací:
Krátké celé číslo, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadByte -Přečíst nepodepsaný bajt

Rozhraní:

```
Byte    ReadByte();
```

Přečtěte si další bajt z proudu bajtů zpráv jako osmibitové celé číslo bez znaménka.

Parametry:
Není

Vrací:
Bajt, který se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadUnsignedKrátké celé číslo bez znaménka pro čtení

Rozhraní:

```
Int32   ReadUnsignedShort();
```

Přečtěte si další 2 bajty z proudu zpráv bajtů jako 16bitové celé číslo bez znaménka.

Parametry:
Není

Vrací:
Krátké celé číslo bez znaménka, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadUTF -Čtení řetězce UTF

Rozhraní:

```
String  ReadUTF();
```

Čte řetězec kódovaný v kódování UTF-8z proudu zpráv v bajtech.

Poznámka: Před voláním ReadUTF() se ujistěte, že kurzor vyrovnávací paměti ukazuje na začátek bajtového proudu zpráv.

Parametry:
Není

Vrací:
Objekt typu String zapouzdřující načtený řetězec.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

Resetovat-Resetovat

Rozhraní:

```
void Reset();
```

Vložte tělo zprávy do režimu jen pro čtení a umístěte kurzor na začátek bajtových proudů zpráv.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotReadableException

WriteBoolean -Hodnota logického zápisu

Rozhraní:

```
void  WriteBoolean(Boolean value);
```

Zapište logickou hodnotu do proudu zpráv v bajtech.

Parametry:

hodnota (vstup)
Logická hodnota, která se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteByte -Bajt zápisu

Rozhraní:

```
void  WriteByte(Byte value);
void  WriteSignedByte(Int16 value);
```

Zapsat bajt do bajtového proudu zpráv.

Parametry:

hodnota (vstup)
Bajt, který se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteBytes -Zapsané bajty

Rozhraní:

```
void  WriteBytes(Byte[] value);
```

Zapsat pole bajtů do proudu zpráv bajtů.

Parametry:

hodnota (vstup)
Pole bajtů, které se mají zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteBytes -Pole dílčích bajtů pro zápis

Rozhraní:

```
void  WriteBytes(Byte[] value, int offset, int length);
```

Zapište dílčí pole bajtů do proudu zpráv bajtů, jak je definováno uvedenou délkou.

Parametry:

hodnota (vstup)
Pole bajtů, které se mají zapsat.

offset (vstup)
Počáteční bod pro pole bajtů, které se mají zapsat.

délka (vstup)
Počet bajtů pro zápis.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteChar -Znak zápisu

Rozhraní:

```
void  WriteChar(Char value);
```

Zapsat znak do bajtového proudu zpráv jako 2 bajty, nejprve jako bajt vysokého pořadí.

Parametry:

hodnota (vstup)
Znak, který se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteDouble -číslo s pohyblivou řádovou čárkou a dvojitou přesností pro zápis.

Rozhraní:

```
void  WriteDouble(Double value);
```

Převést číslo s pohyblivou řádovou čárkou s dvojitou přesností na dlouhé celé číslo a zapsat dlouhé celé číslo do proudu zpráv bajtů jako 8 bajtů, nejprve bajt s vysokým pořadovým číslem.

Parametry:

hodnota (vstup)
Číslo s pohyblivou řádovou čárkou s dvojitou přesností, které má být zapsáno.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteFloat -Číslo plovoucí řádové čárky pro zápis

Rozhraní:

```
void  WriteFloat(Single value);
```

Převést číslo s pohyblivou řádovou čárkou na celé číslo a zapsat celé číslo do proudu zpráv bajtů jako 4 bajty, nejprve bajt s vysokým pořadovým číslem.

Parametry:

hodnota (vstup)
Číslo s pohyblivou řádovou čárkou, které se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteInt -celé číslo pro zápis

Rozhraní:

```
void  WriteInt(Int32 value);
```

Zapsat celé číslo do bajtového proudu zpráv jako 4 bajty, nejprve jako bajt vysokého pořadí.

Parametry:

hodnota (vstup)
Celé číslo, které se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteLong -Zapsat dlouhé celé číslo

Rozhraní:

```
void  WriteLong(Int64 value);
```

Zapsat dlouhé celé číslo do bajtového proudu zpráv jako 8 bajtů, nejvyšší pořadí bajtů jako první.

Parametry:

hodnota (vstup)
Dlouhé celé číslo, které se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteObject -Zapsat objekt

Rozhraní:

```
void  WriteObject(Object value);
```

Zapište uvedený objekt do proudu bajtových zpráv.

Parametry:

hodnota (vstup)
Objekt, který má být zapsán, což musí být odkaz na primitivní typ.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteShort -Zapsat krátké celé číslo

Rozhraní:

```
void  WriteShort(Int16 value);
```

Zapsat krátké celé číslo do bajtového proudu zpráv jako 2 bajty, nejprve jako nejvyšší bajt.

Parametry:

hodnota (vstup)
Krátké celé číslo, které se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteUTF -Zapsat řetězec UTF

Rozhraní:

```
void  WriteUTF(String value);
```

Zapsat řetězec, kódovaný v UTF-8, do proudu zpráv bajtů.

Parametry:

hodnota (vstup)
Objekt typu String zapouzdřující řetězec, který má být zapsán.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Zděděné vlastnosti a metody

Níže uvedené vlastnosti jsou zděděny z rozhraní IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType,Properties

Níže uvedené metody jsou zděděny z rozhraní IMessage:

clearBody, clearProperties, PropertyExists

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IPřipojení

Objekt připojení představuje aktivní připojení aplikace k serveru systému zpráv.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnection
```

Seznam XMS definovaných vlastností objektu připojení viz “Vlastnosti připojení” na stránce 2026.

.NET vlastnosti

ClientID -Získat a nastavit ID klienta.

Rozhraní:

```
String ClientID
      {
        get;
        set;
      }
```

Získejte a nastavte identifikátor klienta pro připojení.

Identifikátor klienta může být předkonfigurován administrátorem v ConnectionFactorynebo přiřazen nastavením ClientID.

Identifikátor klienta se používá pouze pro podporu trvalých odběrů v doméně publikování/odběru a v doméně typu point-to-point se ignoruje.

Pokud aplikace nastaví identifikátor klienta pro připojení, musí tak učinit okamžitě po vytvoření připojení a před provedením dalších operací na připojení. Pokud se aplikace pokusí nastavit identifikátor klienta za tímto bodem, volání vyvolá výjimku IllegalStateException.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState
- InvalidClientIDException

ExceptionListener -Získat a nastavit modul listener pro výjimky

Rozhraní:

```
    ExceptionListener ExceptionListener
        {
          get;
          set;
        }
```

Získejte modul listener pro výjimky, který je registrován s připojením, a registrujte modul listener pro výjimky s připojením.

Není-li s připojením registrován žádný modul listener pro výjimky, metoda vrátí hodnotu null. Pokud je již v připojení registrován modul listener pro výjimky, můžete registraci zrušit zadáním hodnoty null namísto modulu listener pro výjimky.

Další informace o použití modulů listener pro výjimky naleznete v tématu Použití modulů listener pro zprávy a výjimky v produktu .NET.

Výjimky:

- Výjimka XMSException

Metadata-získat metadata

Rozhraní:

```
IConnectionMetaData MetaData
    {
      get;
    }
```

Získejte metadata pro připojení.

Výjimky:

- Výjimka XMSException

metody

Zavřít-zavřít připojení

Rozhraní:

```
void  Close();
```

Zavřete připojení.

Pokud se aplikace pokusí zavřít připojení, které je již zavřené, volání se ignoruje.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException

CreateSession -Vytvořit relaci

Rozhraní:

```
    ISession CreateSession(Boolean transacted,
                        AcknowledgeMode acknowledgeMode);
```

Vytvořte relaci.

Parametry:

transakční (vstup)
Hodnota `True` znamená, že relace je transakční. Hodnota `False` znamená, že relace není transakční.

Pro připojení v reálném čase ke zprostředkovateli musí být hodnota `False`.

acknowledgeMode (vstup)
Označuje, jak jsou potvrzeny zprávy přijaté aplikací. Hodnota musí být jedna z následujících z výčtového nástroje AcknowledgeMode :

`AcknowledgeMode.AutoAcknowledge`
`AcknowledgeMode.ClientAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

Pro připojení v reálném čase ke zprostředkovateli musí být hodnota `AcknowledgeMode.AutoAcknowledge` nebo `AcknowledgeMode.DupsOkAcknowledge` .

Tento parametr je ignorován, pokud je relace transakční. Další informace o režimech potvrzení naleznete v tématu Potvrzení zprávy.

Vrací:
Objekt relace

Výjimky:

- Výjimka XMSException

Spustit-Spustit připojení

Rozhraní:

```
void  Start();
```

Spusťte nebo restartujte doručení příchozích zpráv pro připojení. Volání je ignorováno, pokud je připojení již spuštěno.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException

Zastavit-Zastavit připojení

Rozhraní:

```
void  Stop();
```

Zastavte doručování příchozích zpráv pro připojení. Volání je ignorováno, pokud je připojení již zastaveno.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IConnectionFactory

Aplikace používá továrnu připojení k vytvoření připojení.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnectionFactory
```

Seznam XMS definovaných vlastností objektu ConnectionFactory viz "Vlastnosti ConnectionFactory" na stránce 2026.

metody

CreateConnection -Vytvořit továrnu připojení (s použitím výchozí identity uživatele)

Rozhraní:

```
IConnection CreateConnection();
```

Vytvořte továrnu připojení s výchozími vlastnostmi.

Pokud se připojujete k produktu IBM MQ a XMSC_USERID není nastaveno, použije správce front standardně userID přihlášeného uživatele. Pokud požadujete další ověření jednotlivých uživatelů na úrovni připojení, můžete napsat uživatelskou proceduru ověření klienta, která je nakonfigurována v produktu IBM MQ.

Parametry:
Není

Výjimky:

- Výjimka XMSException

CreateConnection -Vytvořit připojení (pomocí zadané identity uživatele)

Rozhraní:

```
IConnection CreateConnection(String userId, String password);
```

Vytvořte připojení pomocí zadané identity uživatele.

Pokud se připojujete k produktu IBM MQ a XMSC_USERID není nastaveno, použije správce front standardně userID přihlášeného uživatele. Pokud požadujete další ověření jednotlivých uživatelů na úrovni připojení, můžete napsat uživatelskou proceduru ověření klienta, která je nakonfigurována v produktu IBM MQ.

Připojení je vytvořeno v zastaveném režimu. Dokud aplikace nevyvolá volání **`Connection.start()`**, nebudou doručeny žádné zprávy.

Parametry:

userID (vstup)
Objekt typu String zapouzdřující identifikátor uživatele, který má být použit k ověření aplikace. Zadáte-li hodnotu null, dojde k pokusu o vytvoření připojení bez ověření.

heslo (vstup)
Objekt typu String zapouzdřující heslo, které má být použito k ověření aplikace. Zadáte-li hodnotu null, dojde k pokusu o vytvoření připojení bez ověření.

Vrací:
Objekt připojení.

Výjimky:

- Výjimka XMSException
- XMS_X_SECURITY_EXCEPTION

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IConnectionMetaData

Objekt ConnectionMetaData poskytuje informace o připojení.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnectionMetaData
```

Seznam XMS definovaných vlastností datového objektu ConnectionMetaviz “Vlastnosti dat ConnectionMeta” na stránce 2031.

.NET vlastnosti

JMSXPropertyNames -Získat vlastnosti definované zprávy JMS.

Rozhraní:

```
System.Collections.IEnumerator JMSXPropertyNames
    {
      get;
    }
```

Vrátit výčet názvů JMS definovaných vlastností zprávy podporovaných připojením.

JMS definované vlastnosti zprávy nejsou podporovány připojením v reálném čase ke zprostředkovateli.

Výjimky:

- Výjimka XMSException

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

ICíl

Místo určení je místo, kam aplikace odesílá zprávy, nebo je to zdroj, ze kterého aplikace přijímá zprávy, nebo obojí.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IDestination
```

Seznam XMS definovaných vlastností cílového objektu viz “Vlastnosti místa určení” na stránce 2031.

.NET vlastnosti

Název-Získat název cíle

Rozhraní:

```
String Name
{
  get;
}
```

Získejte název místa určení. Název je řetězec zapouzdřující buď název fronty, nebo název tématu.

Výjimky:

- Výjimka XMSException

TypeId -Získat typ cíle.

Rozhraní:

```
DestinationType TypeId
{
  get;
}
```

Získejte typ cíle. Typ cíle je jedna z následujících hodnot:

```
DestinationType.Queue
DestinationType.Topic
```

Výjimky:

- Výjimka XMSException

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

ExceptionListener

Aplikace používá modul listener pro výjimky, aby byla asynchronně upozorněna na problém s připojením.

Hierarchie dědičnosti:

Není

Pokud aplikace používá připojení pouze k asynchronnímu příjmu zpráv a pro žádný jiný účel, pak jediný způsob, jak se může aplikace dozvědět o problému s připojením, je použití modulu listener pro výjimky. V jiných situacích může modul listener pro výjimky poskytnout bezprostřednější způsob, jak se dozvědět o problému s připojením, než aby čekal na další synchronní volání XMS.

Delegát

ExceptionListener -Modul listener pro výjimky

Rozhraní:

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

Informujte aplikaci o problému s připojením.

Metody, které implementují tohoto delegáta, mohou být registrovány s připojením.

Další informace o použití modulů listener pro výjimky naleznete v tématu Použití modulů listener pro zprávy a výjimky v produktu .NET.

Parametry:

výjimka (vstup)
Ukazatel na výjimku vytvořenou pomocí XMS.

Vrací:
Void

Výjimka IllegalState

XMS vyvolá tuto výjimku, pokud aplikace volá metodu v chybném nebo nevhodném čase, nebo pokud XMS není v odpovídajícím stavu pro požadovanou operaci.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.Exception
            |
            +----IBM.XMS.IllegalStateException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

InitialContext

Aplikace používá objekt InitialContext k vytvoření objektů z definic objektů, které jsou načteny z úložiště spravovaných objektů.

Hierarchie dědičnosti:
Není

.NET vlastnosti

Prostředí-Získejte prostředí

Rozhraní:

```
Hashtable Environment
{
     get;
}
```

Získejte prostředí.

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

Konstruktory

InitialContext -Vytvořit počáteční kontext.

Rozhraní:

```
InitialContext(Hashtable env);
```

Vytvořte objekt InitialContext .

Parametry:

Informace nezbytné pro vytvoření připojení k úložišti spravovaných objektů jsou poskytnuty konstruktoru v hašovací tabulce prostředí.

Výjimky:

- Výjimka XMSException

metody

AddToprostředí-Přidat novou vlastnost do prostředí

Rozhraní:

```
Object AddToEnvironment(String propName, Object propVal);
```

Přidejte novou vlastnost do prostředí.

Parametry:

propName (vstup)
Objekt typu String zapouzdřující název vlastnosti, která má být přidána.

propVal (vstup)
Hodnota vlastnosti, která se má přidat.

Vrací:
Stará hodnota vlastnosti.

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

Zavřít-Zavřít tento kontext

Rozhraní:

```
void Close()
```

Zavřít tento kontext.

Parametry:
Není

Vrací:
Není

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

Vyhledat-vyhledat objekt v počátečním kontextu

Rozhraní:

```
Object Lookup(String name);
```

Vytvořte objekt z definice objektu, která je načtena z úložiště spravovaných objektů.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název spravovaného objektu, který má být načten. Název může být buď jednoduchý název, nebo komplexní název. Další podrobnosti viz Načtení spravovaných objektů.

Vrací:
Buď IConnectionFactory , nebo IDestination, v závislosti na typu načítaného objektu. Pokud má funkce přístup k adresáři, ale nemůže najít požadovaný objekt, vrátí se hodnota null.

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

RemoveFromprostředí-Odebrat vlastnost z prostředí

Rozhraní:

```
Object RemoveFromEnvironment(String propName);
```

Odebrat vlastnost z prostředí.

Parametry:

propName (vstup)
Objekt typu String zapouzdřující název vlastnosti, která má být odebrána.

Vrací:
Objekt, který byl odebrán.

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

InvalidClientIDException

XMS vyvolá tuto výjimku, pokud se aplikace pokusí nastavit identifikátor klienta pro připojení, ale identifikátor klienta není platný nebo se již používá.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidClientIDException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

Výjimka InvalidDestination

XMS vyvolá tuto výjimku, pokud aplikace uvede místo určení, které není platné.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidDestinationException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

Výjimka InvalidSelector

XMS vyvolá tuto výjimku, pokud aplikace poskytne výraz selektoru zpráv, jehož syntaxe je neplatná.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidSelectorException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

IMapMessage

Zpráva mapy je zpráva, jejíž tělo se skládá ze sady dvojic název-hodnota, kde každá hodnota má přidružený datový typ.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IMapMessage
```

Když aplikace získá hodnotu dvojice název-hodnota, může být hodnota převedena produktem XMS na jiný datový typ. Další informace o této formě implicitního převodu naleznete v informacích o zprávách mapy v části Tělo zprávy XMS.

.NET vlastnosti

MapNames -Získání názvů mapy

Rozhraní:

```
System.Collections.IEnumerator MapNames
{
  get;
}
```

Získejte výčet názvů v těle zprávy mapy.

Výjimky:

- Výjimka XMSException

metody

GetBoolean -Získat logickou hodnotu.

Rozhraní:

```
Boolean GetBoolean(String name);
```

Získejte logickou hodnotu identifikovanou názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje logickou hodnotu.

Vrací:
Logická hodnota načtená z těla zprávy mapy.

Výjimky:

- Výjimka XMSException

GetByte -bajt získání

Rozhraní:

```
Byte    GetByte(String name);
    Int16   GetSignedByte(String name);
```

Získejte bajt identifikovaný názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje bajt.

Vrací:
Bajt načtený z těla zprávy mapy. Na bajtu se neprovádí žádná konverze dat.

Výjimky:

- Výjimka XMSException

GetBytes -Počet bajtů získání

Rozhraní:

```
Byte[]  GetBytes(String name);
```

Získejte pole bajtů identifikované názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje pole bajtů.

Vrací:
Počet bajtů v poli.

Výjimky:

- Výjimka XMSException

GetChar -Získat znak

Rozhraní:

```
Char    GetChar(String name);
```

Získat znak identifikovaný názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje znak.

Vrací:
Znak načtený z těla zprávy mapy.

Výjimky:

- Výjimka XMSException

GetDouble -Získat číslo s pohyblivou řádovou čárkou s dvojitou přesností.

Rozhraní:

```
Double  GetDouble(String name);
```

Získat číslo s pohyblivou řádovou čárkou s dvojitou přesností identifikované názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje číslo s pohyblivou řádovou čárkou s dvojitou přesností.

Vrací:
Číslo s pohyblivou řádovou čárkou s dvojitou přesností načtené z těla zprávy mapy.

Výjimky:

- Výjimka XMSException

GetFloat -Získat číslo s pohyblivou řádovou čárkou.

Rozhraní:

```
Single  GetFloat(String name);
```

Získejte číslo s pohyblivou řádovou čárkou identifikované názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje číslo s pohyblivou řádovou čárkou.

Vrací:
Číslo s pohyblivou řádovou čárkou načtené z těla zprávy mapy.

Výjimky:

- Výjimka XMSException

GetInt -Získat celé číslo

Rozhraní:

```
Int32   GetInt(String name);
```

Získejte celé číslo identifikované názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje celé číslo.

Vrací:
Celé číslo načtené z těla zprávy mapy.

Výjimky:

- Výjimka XMSException

GetLong -Získat dlouhé celé číslo

Rozhraní:

```
Int64   GetLong(String name);
```

Získejte dlouhé celé číslo identifikované názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje dlouhé celé číslo.

Vrací:
Dlouhé celé číslo načtené z těla zprávy mapy.

Výjimky:

- Výjimka XMSException

GetObject -Získat objekt

Rozhraní:

```
Object  GetObject(String name);
```

Získejte odkaz na hodnotu dvojice název-hodnota z těla zprávy mapy. Dvojice název-hodnota je identifikována názvem.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název dvojice název-hodnota.

Vrací:
Hodnota, která je jedním z následujících typů objektů:

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

Výjimky:
Výjimka XMSException

GetShort -Získat krátké celé číslo

Rozhraní:

```
Int16   GetShort(String name);
```

Získejte krátké celé číslo identifikované názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje krátké celé číslo.

Vrací:
Krátké celé číslo načtené z těla zprávy mapy.

Výjimky:

- Výjimka XMSException

GetString -Získat řetězec

Rozhraní:

```
String  GetString(String name);
```

Získejte řetězec identifikovaný názvem z těla zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název, který identifikuje řetězec v těle zprávy mapy.

Vrací:
Objekt typu String zapouzdřující řetězec načtený z těla zprávy mapy. Je-li požadován převod dat, tato hodnota je řetězec po převodu.

Výjimky:

- Výjimka XMSException

ItemExists -Kontrola dvojice název-hodnota existuje

Rozhraní:

```
Boolean ItemExists(String name);
```

Zkontrolujte, zda tělo zprávy mapy obsahuje dvojici název-hodnota s uvedeným názvem.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název dvojice název-hodnota.

Vrací:

- `True`, pokud tělo zprávy mapy obsahuje dvojici název-hodnota s uvedeným názvem.
- `False`, pokud tělo zprávy mapy neobsahuje dvojici název-hodnota s uvedeným názvem.

Výjimky:

- Výjimka XMSException

SetBoolean -Nastavit logickou hodnotu

Rozhraní:

```
void  SetBoolean(String name, Boolean value);
```

Nastavte logickou hodnotu v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci logické hodnoty v těle zprávy mapy.

hodnota (vstup)
Logická hodnota, která se má nastavit.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetByte -Nastavit bajt

Rozhraní:

```
void  SetByte(String name, Byte value);
void  SetSignedByte(String name, Int16 value);
```

Nastavte bajt v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci bajtu v těle zprávy mapy.

hodnota (vstup)
Bajt, který má být nastaven.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetBytes -Nastavit bajty

Rozhraní:

```
void  SetBytes(String name, Byte[] value);
```

Nastavte pole bajtů v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci pole bajtů v těle zprávy mapy.

hodnota (vstup)
Pole bajtů, které mají být nastaveny.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetChar -Nastavit znak

Rozhraní:

```
void  SetChar(String name, Char value);
```

Nastavte 2bajtový znak v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci znaku v těle zprávy mapy.

hodnota (vstup)
Znak, který má být nastaven.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetDouble -Nastavit číslo s pohyblivou řádovou čárkou a dvojitou přesností

Rozhraní:

```
void  SetDouble(String name, Double value);
```

Nastavte číslo s pohyblivou řádovou čárkou s dvojitou přesností v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci čísla s pohyblivou řádovou čárkou s dvojitou přesností v těle zprávy mapy.

hodnota (vstup)
Číslo s pohyblivou řádovou čárkou s dvojitou přesností, které má být nastaveno.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetFloat -Nastavit číslo s pohyblivou řádovou čárkou

Rozhraní:

```
void  SetFloat(String name, Single value);
```

Nastavte číslo s pohyblivou řádovou čárkou v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci čísla s pohyblivou řádovou čárkou v těle zprávy mapy.

hodnota (vstup)
Číslo s pohyblivou řádovou čárkou, které má být nastaveno.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetInt -nastavit celé číslo

Rozhraní:

```
void  SetInt(String name, Int32 value);
```

Nastavte celé číslo v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci celého čísla v těle zprávy mapy.

hodnota (vstup)
Celé číslo, které má být nastaveno.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetLong -nastavit dlouhé celé číslo

Rozhraní:

```
void  SetLong(String name, Int64 value);
```

Nastavte dlouhé celé číslo v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci dlouhého celého čísla v těle zprávy mapy.

hodnota (vstup)
Dlouhé celé číslo, které má být nastaveno.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetObject -Nastavit objekt

Rozhraní:

```
void  SetObject(String name, Object value);
```

Nastavte hodnotu, která musí být primitivním typem XMS , v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci hodnoty v těle zprávy mapy.

hodnota (vstup)
Pole bajtů obsahující hodnotu, která má být nastavena.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetShort -Nastavit krátké celé číslo

Rozhraní:

```
void  SetShort(String name, Int16 value);
```

Nastavte krátké celé číslo v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci krátkého celého čísla v těle zprávy mapy.

hodnota (vstup)
Krátké celé číslo, které má být nastaveno.

Vrací:
Void

Výjimky:

- Výjimka XMSException

SetString -Nastavit řetězec

Rozhraní:

```
void  SetString(String name, String value);
```

Nastavte řetězec v těle zprávy mapy.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název pro identifikaci řetězce v těle zprávy mapy.

hodnota (vstup)
Objekt typu String zapouzdřující řetězec, který má být nastaven.

Vrací:
Void

Výjimky:

- Výjimka XMSException

Zděděné vlastnosti a metody

Níže uvedené vlastnosti jsou zděděny z rozhraní IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType,Properties

Níže uvedené metody jsou zděděny z rozhraní IMessage:

clearBody, clearProperties, PropertyExists

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

Zpráva IMessage

Objekt zprávy představuje zprávu, kterou aplikace odesílá nebo přijímá. IMessage je supertřída pro třídy zpráv, jako např. IMapMessage.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
```

Seznam polí záhlaví zprávy JMS v objektu zprávy viz Pole záhlaví zprávy XMS. Seznam JMS definovaných vlastností objektu zprávy naleznete v tématu Vlastnosti zprávy definované JMS. Seznam IBM definovaných vlastností objektu zprávy viz IBM-defined properties of a message. Seznam vlastností JMS_IBM_MQMD* pro objekt Message viz “Vlastnosti JMS_IBM_MQMD*” na stránce 2035

Zprávy jsou odstraněny programem pro uvolnění paměti. Když je zpráva odstraněna, uvolní prostředky, které používala.

.NET vlastnosti

GetJMSCorrelationID-získání a nastavení JMSCorrelationID

Rozhraní:

```
String JMSCorrelationID
{
  get;
  set;
}
```

Získat a nastavit identifikátor korelace zprávy jako objekt typu String.

Výjimky:

- Výjimka XMSException

JMSDeliveryMode -Získat a nastavit JMSDeliveryMode

Rozhraní:

```
DeliveryMode JMSDeliveryMode
{
  get;
  set;
}
```

Získat a nastavit režim doručení zprávy.

Způsob doručení zprávy má jednu z následujících hodnot:

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

Pro nově vytvořenou zprávu, která nebyla odeslána, je režim doručení `DeliveryMode.Trvalé`, s výjimkou připojení v reálném čase ke zprostředkovateli, pro kterého je režim doručení `DeliveryMode.NonPersistent`. Pro přijatou zprávu metoda vrací režim doručení nastavený voláním IMessageProducer.send () při odesílání zprávy, pokud přijímající aplikace nezmění režim doručení nastavením JMSDeliveryMode.

Výjimky:

- Výjimka XMSException

JMSDestination-Získat a nastavit JMSDestination

Rozhraní:

```
IDestination JMSDestination
{
  get;
  set;
}
```

Získat a nastavit cíl zprávy.

Místo určení je nastaveno voláním IMessageProducer.send () při odesílání zprávy. Hodnota JMSDestination je ignorována. Místo určení JMSDestination však můžete použít ke změně místa určení přijaté zprávy.

Pro nově vytvořenou zprávu, která nebyla odeslána, metoda vrátí objekt Destination s hodnotou null, pokud odesílající aplikace nenastaví cíl nastavením JMSDestination. Pro přijatou zprávu metoda vrací objekt Destination pro místo určení, které bylo nastaveno voláním IMessageProducer.send () při odeslání zprávy, pokud přijímající aplikace nezmění místo určení nastavením JMSDestination.

Výjimky:

- Výjimka XMSException

JMSExpiration-Získat a nastavit JMSExpiration

Rozhraní:

```
Int64 JMSExpiration
{
  get;
  set;
}
```

Získejte a nastavte čas vypršení platnosti zprávy.

Čas vypršení platnosti je nastaven voláním IMessageProducer.send () při odesílání zprávy. Jeho hodnota se vypočítá přičtením doby platnosti, jak je určeno odesílající aplikací, k času, kdy je zpráva odeslána. Čas vypršení platnosti je vyjádřen v milisekundách od 00:00:00 GMT 1. ledna 1970.

Pro nově vytvořenou zprávu, která nebyla odeslána, je čas vypršení platnosti 0, pokud odesílající aplikace nenastaví jiný čas vypršení platnosti nastavením JMSExpiration. Pro přijatou zprávu metoda vrací čas vypršení platnosti, který byl nastaven voláním IMessageProducer.send () při odesílání zprávy, pokud přijímající aplikace nezmění čas vypršení platnosti nastavením JMSExpiration.

Je-li doba platnosti 0, volání IMessageProducer.send () nastaví dobu vypršení platnosti na 0, čímž označí, že zpráva nevyprší.

Produkt XMS vyřadí zprávy s vypršenou platností a nedoručí je do aplikací.

Výjimky:

- Výjimka XMSException

JMSMessageID -Získat a nastavit JMSMessageID

Rozhraní:

```
String JMSMessageID
{
  get;
  set;
}
```

Získejte a nastavte identifikátor zprávy jako objekt řetězce zapouzdřující identifikátor zprávy.

Identifikátor zprávy je nastaven voláním IMessageProducer.send () při odesílání zprávy. Pro přijatou zprávu metoda vrací identifikátor zprávy, který byl nastaven voláním IMessageProducer.send () při odesílání zprávy, pokud přijímající aplikace nezmění identifikátor zprávy nastavením JMSMessageID.

Pokud zpráva nemá žádný identifikátor zprávy, metoda vrátí hodnotu null.

Výjimky:

- Výjimka XMSException

JMSPriority-Získat a nastavit JMSPriority

Rozhraní:

```
Int32 JMSPriority
{
  get;
  set;
}
```

Získat a nastavit prioritu zprávy.

Prioritu nastavuje volání IMessageProducer.send () při odesílání zprávy. Hodnota je celé číslo v rozsahu `0`, nejnižší priorita, až 9, nejvyšší priorita.

Pro nově vytvořenou zprávu, která nebyla odeslána, je priorita 4 , pokud odesílající aplikace nenastaví jinou prioritu nastavením JMSPriority. Pro přijatou zprávu metoda vrací prioritu, která byla nastavena voláním IMessageProducer.send () při odesílání zprávy, pokud přijímající aplikace nezmění prioritu nastavením JMSPriority.

Výjimky:

- Výjimka XMSException

JMSRedelivered-Získat a nastavit JMSRedelivered

Rozhraní:

```
Boolean JMSRedelivered
{
  get;
  set;
}
```

Získejte informace o tom, zda je zpráva znovu doručována, a informace o tom, zda je zpráva znovu doručována. Označení je nastaveno voláním IMessageConsumer.receive () při přijetí zprávy.

Tato vlastnost má následující hodnoty:

- `True`, pokud se zpráva znovu doručuje.
- `False`, pokud se zpráva znovu nedoručuje.

Pro připojení v reálném čase ke zprostředkovateli je hodnota vždy `False`.

Označení opakovaného doručení nastavené JMSRedelivered před odesláním zprávy je ignorováno voláním IMessageProducer.send () při odeslání zprávy a je ignorováno a nahrazeno voláním IMessageConsumer.receive () při přijetí zprávy. Můžete však použít JMSRedelivered ke změně indikace pro přijatou zprávu.

Výjimky:

- Výjimka XMSException

JMSReplyTo -Získat a nastavit JMSReplyTo

Rozhraní:

```
IDestination JMSReplyTo
{
  get;
  set;
}
```

Získejte a nastavte místo určení, kam se má odeslat odpověď na zprávu.

Hodnota této vlastnosti je cílový objekt pro místo určení, kam se má odeslat odpověď na zprávu. Objekt místa určení s hodnotou Null znamená, že není očekávána žádná odpověď.

Výjimky:

- Výjimka XMSException

JMSTimestamp-Získat a nastavit JMSTimestamp

Rozhraní:

```
Int64 JMSTimestamp
{
  get;
  set;
}
```

Získat a nastavit čas, kdy byla zpráva odeslána.

Časové razítko je nastaveno voláním IMessageProducer.send () při odeslání zprávy a je vyjádřeno v milisekundách od 1. ledna 1970 od 00:00:00 GMT.

Pro nově vytvořenou zprávu, která nebyla odeslána, je časové razítko 0, pokud odesílající aplikace nenastaví jiné časové razítko nastavením JMSTimestamp. Pro přijatou zprávu metoda vrací časové razítko nastavené voláním IMessageProducer.send () při odeslání zprávy, pokud přijímající aplikace nezmění časové razítko nastavením JMSTimestamp.

Výjimky:

- Výjimka XMSException

Notes:

1. Není-li časové razítko definováno, metoda vrátí hodnotu 0, ale nevyvolá žádnou výjimku.

JMSType-Získat a nastavit JMSType

Rozhraní:

```
String JMSType
{
  get;
  set;
}
```

Získejte a nastavte typ zprávy.

Hodnota JMSType je řetězec zapouzdřující typ zprávy. Je-li požadován převod dat, tato hodnota je typ po převodu.

Výjimky:

- Výjimka XMSException

PropertyNames -Získání vlastností

Rozhraní:

```
System.Collections.IEnumerator PropertyNames
{
  get;
}
```

Získejte výčet vlastností názvů zprávy.

Výjimky:

- Výjimka XMSException

metody

Potvrdit-Potvrdit

Rozhraní:

```
void  Acknowledge();
```

Potvrzte tuto zprávu a všechny dříve nepotvrzené zprávy přijaté relací.

Aplikace může volat tuto metodu, pokud je režim potvrzení relace AcknowledgeMode.ClientAcknowledge. Volání metody jsou ignorována, pokud má relace jiný režim potvrzení nebo pokud je transakční.

Zprávy, které byly přijaty, ale nebyly potvrzeny, mohou být znovu doručeny.

Další informace o potvrzování zpráv viz ../develop/xms_cmesack.dita#xms_cmesack.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState

ClearBody -Vymazat tělo

Rozhraní:

```
void  ClearBody();
```

Vymazat tělo zprávy. Pole záhlaví a vlastnosti zprávy nejsou vymazány.

Pokud aplikace vymaže tělo zprávy, tělo zůstane ve stejném stavu jako prázdné tělo v nově vytvořené zprávě. Stav prázdného těla v nově vytvořené zprávě závisí na typu těla zprávy. Další informace viz Tělo zprávy XMS.

Aplikace může kdykoli vymazat tělo zprávy bez ohledu na to, v jakém stavu se tělo nachází. Pokud je tělo zprávy jen pro čtení, jediný způsob, jak může aplikace do těla zapisovat, je, aby aplikace nejprve tělo vymazala.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException

ClearProperties -Vymazat vlastnosti

Rozhraní:

```
void  ClearProperties();
```

Vymazat vlastnosti zprávy. Pole záhlaví a tělo zprávy nejsou vymazány.

Pokud aplikace vymaže vlastnosti zprávy, budou tyto vlastnosti čitelné a schopné zápisu.

Aplikace může kdykoli vymazat vlastnosti zprávy bez ohledu na to, v jakém stavu se vlastnosti nacházejí. Pokud jsou vlastnosti zprávy jen pro čtení, jediný způsob, jak se mohou tyto vlastnosti stát zapisovatelnými, je, aby aplikace nejprve vymazala vlastnosti.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException

PropertyExists -Zkontrolujte, zda vlastnost existuje.

Rozhraní:

```
Boolean PropertyExists(String propertyName);
```

Zkontrolujte, zda má zpráva vlastnost s uvedeným názvem.

Parametry:

propertyName (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:

- `True`, pokud má zpráva vlastnost s uvedeným názvem.
- `False`, pokud zpráva nemá vlastnost s uvedeným názvem.

Výjimky:

- Výjimka XMSException

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IMessageConsumer

Aplikace používá spotřebitele zpráv k přijímání zpráv odeslaných do místa určení.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessageConsumer
```

Seznam XMS definovaných vlastností objektu MessageConsumer naleznete v části “Vlastnosti objektu MessageConsumer” na stránce 2038.

.NET vlastnosti

MessageListener -Získat a nastavit modul listener pro zprávy.

Rozhraní:

```
MessageListener MessageListener
{
  get;
  set;
}
```

Získejte modul listener pro zprávy, který je registrován u spotřebitele zpráv, a registrujte modul listener pro zprávy u spotřebitele zpráv.

Není-li u spotřebitele zpráv registrován žádný modul listener pro zprávy, má parametr MessageListener hodnotu Null. Pokud je modul listener pro zprávy již registrován u spotřebitele zpráv, můžete registraci zrušit zadáním hodnoty null.

Další informace o použití modulů listener pro zprávy naleznete v tématu Použití modulů listener pro zprávy a výjimky v prostředí .NET.

Výjimky:

- Výjimka XMSException

MessageSelector -Získat selektor zpráv

Rozhraní:

```
String MessageSelector
{
  get;
}
```

Získejte selektor zpráv pro spotřebitele zpráv. Návratová hodnota je objekt typu String zapouzdřující výraz selektoru zpráv. Je-li požadován převod dat, tato hodnota je výraz selektoru zpráv po převodu. Pokud spotřebitel zpráv nemá selektor zpráv, hodnota MessageSelector je objekt typu String s hodnotou null.

Výjimky:

- Výjimka XMSException

metody

Zavřít-zavřít spotřebitele zpráv

Rozhraní:

```
void  Close();
```

Zavřete spotřebitele zpráv.

Pokud se aplikace pokusí zavřít spotřebitele zpráv, který je již zavřený, volání se ignoruje.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException

Přijmout-Přijmout

Rozhraní:

```
IMessage Receive();
```

Přijměte další zprávu pro spotřebitele zpráv. Volání čeká na zprávu po neomezenou dobu nebo dokud není spotřebitel zpráv zavřen.

Parametry:
Není

Vrací:
Ukazatel na objekt zprávy. Pokud je spotřebitel zpráv zavřen, zatímco volání čeká na zprávu, metoda vrátí ukazatel na objekt zprávy s hodnotou null.

Výjimky:

- Výjimka XMSException

Příjem-příjem (s intervalem čekání)

Rozhraní:

```
IMessage Receive(Int64 delay);
```

Přijměte další zprávu pro spotřebitele zpráv. Volání čeká na zprávu pouze určenou dobu, nebo dokud není spotřebitel zpráv zavřen.

Parametry:

prodleva (vstup)
Doba v milisekundách, po kterou volání čeká na zprávu. Zadáte-li interval čekání 0, bude volání čekat na zprávu nekonečně dlouho.

Vrací:
Ukazatel na objekt zprávy. Pokud během intervalu čekání nedorazí žádná zpráva nebo pokud je spotřebitel zpráv zavřen, zatímco volání čeká na zprávu, metoda vrátí ukazatel na objekt zprávy s hodnotou null, ale nevyvolá žádnou výjimku.

Výjimky:

- Výjimka XMSException

ReceiveNoWait-Příjem bez čekání

Rozhraní:

```
IMessage ReceiveNoWait();
```

Přijmout další zprávu pro spotřebitele zpráv, pokud je k dispozici okamžitě.

Parametry:
Není

Vrací:
Ukazatel na objekt zprávy. Pokud není okamžitě k dispozici žádná zpráva, metoda vrátí ukazatel na objekt zprávy s hodnotou null.

Výjimky:

- Výjimka XMSException

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

MessageEOFException

XMS vyvolá tuto výjimku, pokud XMS zjistí konec proudu bajtových zpráv, když aplikace čte tělo bajtové zprávy.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
            |
            +----IBM.XMS.MessageEOFException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

Výjimka MessageFormat

XMS vyvolá tuto výjimku, pokud XMS zjistí zprávu s neplatným formátem.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
            |
            +----IBM.XMS.MessageFormatException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

IMessageListener (delegát)

Aplikace používá modul listener pro zprávy k asynchronnímu příjmu zpráv.

Hierarchie dědičnosti:
Není

Delegát

MessageListener -Modul listener pro zprávy.

Rozhraní:

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

Asynchronně doručte zprávu spotřebiteli zpráv.

Metody, které implementují tohoto delegáta, mohou být registrovány s připojením.

Další informace o použití modulů listener pro zprávy naleznete v tématu Použití modulů listener pro zprávy a výjimky v produktu .NET.

Parametry:

mesg (vstupní)
Objekt zprávy.

Vrací:
Void

MessageNotReadableException

XMS vyvolá tuto výjimku, pokud se aplikace pokusí přečíst tělo zprávy, které je pouze pro zápis.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageNotReadableException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

MessageNotWritableException

XMS vyvolá tuto výjimku, pokud se aplikace pokusí zapsat do těla zprávy, která je jen pro čtení.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageNotWritableException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

IMessageProducer

Aplikace používá producenta zpráv k odesílání zpráv do místa určení.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
```

```
   |
   +----IBM.XMS.IMessageProducer
```

Seznam XMS definovaných vlastností objektu MessageProducer naleznete v tématu “Vlastnosti prvku MessageProducer” na stránce 2038.

.NET vlastnosti

DeliveryMode -Získat a nastavit výchozí režim doručení.

Rozhraní:

```
DeliveryMode DeliveryMode
{
  get;
  set;
}
```

Získat a nastavit výchozí režim doručení pro zprávy odeslané producentem zpráv.

Výchozí režim doručení má jednu z následujících hodnot:

`DeliveryMode.Persistent`
`DeliveryMode.NonPersistent`

Pro připojení v reálném čase ke zprostředkovateli musí být hodnota `DeliveryMode.NonPersistent`.

Výchozí hodnota je `DeliveryMode.Persistent`, s výjimkou připojení v reálném čase ke zprostředkovateli, pro kterého je výchozí hodnota `DeliveryMode.NonPersistent`.

Výjimky:

- Výjimka XMSException

Cíl-Získat cíl

Rozhraní:

```
IDestination Destination
{
  get;
}
```

Získejte místo určení pro producenta zpráv.

Parametry:
Není

Vrací:
Cílový objekt. Pokud producent zpráv nemá místo určení, metoda vrátí objekt místa určení s hodnotou Null.

Výjimky:

- Výjimka XMSException

DisableMsgID-příznak získání a nastavení zakázání ID zprávy

Rozhraní:

```
Boolean DisableMessageID
{
  get;
  set;
}
```

Získejte informace o tom, zda přijímající aplikace vyžaduje, aby byly identifikátory zpráv zahrnuty do zpráv odesílaných producentem zpráv, a zda přijímající aplikace vyžaduje, aby byly identifikátory zpráv zahrnuty do zpráv odesílaných producentem zpráv.

Při připojení ke správci front nebo při připojení v reálném čase ke zprostředkovateli je tento příznak ignorován. Při připojení ke sběrnici pro integraci služeb je příznak dodržen.

ID DisabledMsgmá následující hodnoty:

- `True`, pokud přijímající aplikace nevyžaduje zahrnutí identifikátorů zpráv do zpráv odeslaných producentem zpráv.
- `False`, pokud přijímající aplikace vyžaduje zahrnutí identifikátorů zpráv do zpráv odeslaných producentem zpráv.

Výjimky:

- Výjimka XMSException

DisableMsgTS-Příznak časového razítka pro získání a nastavení zákazu

Rozhraní:

```
Boolean DisableMessageTimestamp
{
  get;
  set;
}
```

Získejte informace o tom, zda přijímající aplikace vyžaduje zahrnutí časových razítek do zpráv odesílaných producentem zpráv a zda přijímající aplikace vyžaduje zahrnutí časových razítek do zpráv odesílaných producentem zpráv.

Při připojení v reálném čase ke zprostředkovateli je tento příznak ignorován. Při připojení ke správci front nebo při připojení ke sběrnici pro integraci služeb je příznak dodržen.

DisableMsgTS má následující hodnoty:

- `True`, pokud přijímající aplikace nevyžaduje zahrnutí časových razítek do zpráv odeslaných producentem zpráv.
- `False`, pokud přijímající aplikace vyžaduje zahrnutí časových razítek do zpráv odeslaných producentem zpráv.

Vrací:

Výjimky:

- Výjimka XMSException

Priorita-Získat a nastavit výchozí prioritu

Rozhraní:

```
Int32 Priority
{
  get;
  set;
}
```

Získejte a nastavte výchozí prioritu pro zprávy odeslané producentem zpráv.

Hodnota výchozí priority zprávy je celé číslo v rozsahu `0`, nejnižší priorita je 9, nejvyšší priorita.

U připojení v reálném čase ke zprostředkovateli je priorita zprávy ignorována.

Výjimky:

- Výjimka XMSException

TimeTo-Aktivní-Získat a nastavit výchozí dobu platnosti

Rozhraní:

```
Int64 TimeToLive
{
  get;
  set;
}
```

Získejte a nastavte výchozí dobu, po kterou zpráva existuje, než vyprší její platnost.

Čas se měří od okamžiku, kdy producent zpráv odešle zprávu, a jedná se o výchozí dobu platnosti v milisekundách. Hodnota 0 znamená, že zpráva nikdy nevyprší.

Pro připojení v reálném čase ke zprostředkovateli je tato hodnota vždy 0.

Výjimky:

- Výjimka XMSException

metody

Zavřít-zavřít producent zpráv

Rozhraní:

```
void  Close();
```

Zavřete producent zpráv.

Pokud se aplikace pokusí zavřít producent zpráv, který je již zavřený, volání se ignoruje.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException

Odeslat-Odeslat

Rozhraní:

```
void Send(IMessage msg) ;
```

Odeslat zprávu do místa určení, které bylo určeno při vytvoření producenta zpráv. Odešlete zprávu pomocí výchozího režimu doručení, priority a doby platnosti producenta zpráv.

Parametry:

msg (vstup)
Objekt zprávy.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka MessageFormat
- Výjimka InvalidDestination

Odeslat-Odeslat (určení režimu doručení, priority a doby platnosti)

Rozhraní:

```
void Send(IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive);
```

Odeslat zprávu do místa určení, které bylo určeno při vytvoření producenta zpráv. Odešlete zprávu s použitím určeného režimu doručení, priority a doby platnosti.

Parametry:

msg (vstup)
Objekt zprávy.

deliveryMode (vstup)
Režim doručení pro zprávu, který musí mít jednu z následujících hodnot:

`DeliveryMode.Persistent`
`DeliveryMode.NonPersistent`

Pro připojení v reálném čase ke zprostředkovateli musí být hodnota `DeliveryMode.NonPersistent`.

priorita (vstup)
Priorita zprávy. Hodnotou může být celé číslo v rozsahu `0`pro nejnižší prioritu až `9`pro nejvyšší prioritu. U připojení v reálném čase ke zprostředkovateli je hodnota ignorována.

timeToAktivní (vstup)
Doba platnosti zprávy v milisekundách. Hodnota 0 znamená, že zpráva nikdy nevyprší. Pro připojení v reálném čase ke zprostředkovateli musí být hodnota 0.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka MessageFormat
- Výjimka InvalidDestination
- Výjimka IllegalState

Odeslat-Odeslat (do určeného místa určení)

Rozhraní:

```
void Send(IDestination dest, IMessage msg) ;
```

Pokud používáte producenta zpráv, pro kterého nebyl při vytvoření producenta zpráv určen žádný cíl, odešlete zprávu do určeného místa určení. Odešlete zprávu pomocí výchozího režimu doručení, priority a doby platnosti producenta zpráv.

Při vytváření producenta zpráv obvykle určujete místo určení, ale pokud tak neučiníte, musíte určit místo určení při každém odeslání zprávy.

Parametry:

cíl (vstup)
Cílový objekt.

msg (vstup)
Objekt zprávy.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka MessageFormat
- Výjimka InvalidDestination

Odeslat-Odeslat (do určeného místa určení s určením režimu doručení, priority a doby platnosti)

Rozhraní:

```
void Send(IDestination dest,
          IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive) ;
```

Pokud používáte producenta zpráv, pro kterého nebyl při vytvoření producenta zpráv určen žádný cíl, odešlete zprávu do určeného místa určení. Odešlete zprávu s použitím určeného režimu doručení, priority a doby platnosti.

Při vytváření producenta zpráv obvykle určujete místo určení, ale pokud tak neučiníte, musíte určit místo určení při každém odeslání zprávy.

Parametry:

cíl (vstup)
Cílový objekt.

msg (vstup)
Objekt zprávy.

deliveryMode (vstup)
Režim doručení pro zprávu, který musí mít jednu z následujících hodnot:

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

Pro připojení v reálném čase ke zprostředkovateli musí být hodnota `DeliveryMode.NonPersistent`.

priorita (vstup)
Priorita zprávy. Hodnotou může být celé číslo v rozsahu `0`pro nejnižší prioritu až `9`pro nejvyšší prioritu. U připojení v reálném čase ke zprostředkovateli je hodnota ignorována.

timeToAktivní (vstup)
Doba platnosti zprávy v milisekundách. Hodnota 0 znamená, že zpráva nikdy nevyprší. Pro připojení v reálném čase ke zprostředkovateli musí být hodnota 0.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka MessageFormat
- Výjimka InvalidDestination
- Výjimka IllegalState

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IObjectMessage

Zpráva objektu je zpráva, jejíž tělo se skládá ze serializovaného objektu Java nebo .NET .

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IObjectMessage
```

.NET vlastnosti

Objekt-Získat a nastavit objekt jako bajty

Rozhraní:

```
System.Object Object
    {
      get;
      set;
    }

    Byte[] GetObject();
```

Získat a nastavit objekt, který tvoří tělo zprávy objektu.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException
- MessageNotWritableException

Zděděné vlastnosti a metody

Níže uvedené vlastnosti jsou zděděny z rozhraní IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType,Properties

Níže uvedené metody jsou zděděny z rozhraní IMessage:

clearBody, clearProperties, PropertyExists

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IPropertyContext

IPropertyContext je abstraktní supertřída, která obsahuje metody, které získají a nastaví vlastnosti. Tyto metody jsou zděděny jinými třídami.

Hierarchie dědičnosti:
Není

metody

GetBooleanVlastnost-Získat logickou vlastnost

Rozhraní:

```
Boolean GetBooleanProperty(String property_name);
```

Získejte hodnotu logické vlastnosti s uvedeným názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

GetByteVlastnost-Získat bajtovou vlastnost

Rozhraní:

```
Byte    GetByteProperty(String property_name) ;
Int16   GetSignedByteProperty(String property_name) ;
```

Získejte hodnotu vlastnosti bajtu identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetBytes-Vlastnost Get Byte Array

Rozhraní:

```
Byte[]  GetBytesProperty(String property_name) ;
```

Získejte hodnotu vlastnosti bajtového pole identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Počet bajtů v poli.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

GetCharVlastnost-Získat znakovou vlastnost

Rozhraní:

```
Char    GetCharProperty(String property_name) ;
```

Získejte hodnotu vlastnosti dvoubajtového znaku identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetDouble-Získat vlastnost s pohyblivou řádovou čárkou s přesností na dvě desetinná místa

Rozhraní:

```
Double  GetDoubleProperty(String property_name) ;
```

Získejte hodnotu vlastnosti s pohyblivou řádovou čárkou s dvojitou přesností identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

GetFloatVlastnost-Získat vlastnost s pohyblivou řádovou čárkou

Rozhraní:

```
Single  GetFloatProperty(String property_name) ;
```

Získejte hodnotu vlastnosti s pohyblivou řádovou čárkou identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

GetIntVlastnost- GetIntVlastnost

Rozhraní:

```
Int32   GetIntProperty(String property_name) ;
```

Získejte hodnotu celočíselné vlastnosti identifikované názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

GetLongVlastnost-Vlastnost Get Long Integer

Rozhraní:

```
Int64   GetLongProperty(String property_name) ;
```

Získejte hodnotu vlastnosti dlouhého celého čísla identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetObject-Získat vlastnost objektu

Rozhraní:

```
Object  GetObjectProperty( String property_name) ;
```

Získejte hodnotu a datový typ vlastnosti identifikované názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti, která je jedním z následujících typů objektů:

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

GetShortVlastnost-Vlastnost Get Short Integer

Rozhraní:

```
Int16   GetShortProperty(String property_name) ;
```

Získejte hodnotu vlastnosti krátkého celého čísla identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

GetStringVlastnost- GetStringVlastnost

Rozhraní:

```
String  GetStringProperty(String property_name) ;
```

Získejte hodnotu vlastnosti řetězce identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

Vrací:
Objekt typu String zapouzdřující řetězec, který je hodnotou vlastnosti. Je-li požadován převod dat, tato hodnota je řetězec po převodu.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

SetBooleanVlastnost-Nastavit logickou vlastnost

Rozhraní:

```
void SetBooleanProperty( String property_name,  Boolean value) ;
```

Nastavte hodnotu logické vlastnosti identifikované názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetByteVlastnost-Vlastnost Set Byte

Rozhraní:

```
void SetByteProperty( String property_name,  Byte value) ;
      void SetSignedByteProperty( String property_name,  Int16 value) ;
```

Nastavte hodnotu vlastnosti bajtu identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetBytes-Vlastnost Set Byte Array

Rozhraní:

```
void SetBytesProperty( String property_name, Byte[] value ) ;
```

Nastavte hodnotu vlastnosti bajtového pole identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti, která je polem bajtů.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetCharVlastnost-Nastavit znakovou vlastnost

Rozhraní:

```
void SetCharProperty( String property_name,  Char value) ;
```

Nastavte hodnotu 2bajtové znakové vlastnosti určené názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetDoubleVlastnost-Nastavit vlastnost s dvojitou přesností na pohyblivou řádovou čárkou

Rozhraní:

```
void SetDoubleProperty( String property_name,  Double value) ;
```

Nastavte hodnotu vlastnosti s pohyblivou řádovou čárkou s dvojitou přesností identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetFloatVlastnost-Nastavit vlastnost s pohyblivou řádovou čárkou

Rozhraní:

```
void SetFloatProperty( String property_name,  Single value) ;
```

Nastavte hodnotu vlastnosti s pohyblivou řádovou čárkou identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetIntVlastnost-Nastavit celočíselnou vlastnost

Rozhraní:

```
void SetIntProperty( String property_name,  Int32 value) ;
```

Nastavte hodnotu celočíselné vlastnosti určené názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetLongVlastnost-Nastavit vlastnost Long Integer

Rozhraní:

```
void SetLongProperty( String property_name,  Int64 value) ;
```

Nastavte hodnotu vlastnosti dlouhého celého čísla identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetObject-Nastavit vlastnost objektu

Rozhraní:

```
void SetObjectProperty( String property_name,  Object value) ;
```

Nastavte hodnotu a datový typ vlastnosti identifikované názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

objectType (vstup)
Hodnota vlastnosti, která musí být jedním z následujících typů objektů:

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

hodnota (vstup)
Hodnota vlastnosti jako pole bajtů.

délka (vstup)
Počet bajtů v poli.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetShortVlastnost-Nastavit vlastnost krátkého celého čísla

Rozhraní:

```
void SetShortProperty( String property_name,  Int16 value) ;
```

Nastavte hodnotu vlastnosti krátkého celého čísla identifikovanou názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetStringVlastnost-Nastavit vlastnost řetězce

Rozhraní:

```
void SetStringProperty( String property_name,  String value);
```

Nastavte hodnotu vlastnosti řetězce identifikované názvem.

Parametry:

název_vlastnosti (vstup)
Objekt typu String zapouzdřující název vlastnosti.

hodnota (vstup)
Objekt typu String zapouzdřující řetězec, který je hodnotou vlastnosti.

Vrací:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

IQueueBrowser

Aplikace používá prohlížeč front k procházení zpráv ve frontě bez jejich odebrání.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
System.Collections.IEnumerable
   |
   +----IBM.XMS.IQueueBrowser
```

.NET vlastnosti

MessageSelector -Získat selektor zpráv

Rozhraní:

```
String MessageSelector
{
  get;
}
```

Získejte selektor zpráv pro prohlížeč front.

Selektor zpráv je objekt typu String, který zapouzdřuje výraz selektoru zpráv. Je-li požadován převod dat, tato hodnota je výraz selektoru zpráv po převodu. Pokud prohlížeč fronty nemá selektor zpráv, metoda vrátí objekt typu String s hodnotou null.

Výjimky:

- Výjimka XMSException

Fronta-fronta získání

Rozhraní:

```
IDestination Queue
{
  get;
}
```

Získat frontu přidruženou k prohlížeči front jako cílový objekt představující frontu.

Výjimky:

- Výjimka XMSException

metody

Zavřít-zavřít prohlížeč front

Rozhraní:

```
void  Close();
```

Zavřete prohlížeč front.

Pokud se aplikace pokusí zavřít prohlížeč front, který je již zavřený, volání se ignoruje.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException

GetEnumerator -Získat zprávy

Rozhraní:

```
IEnumerator GetEnumerator();
```

Získejte seznam zpráv ve frontě.

Tato metoda vrací výčtový modul, který zapouzdřuje seznam objektů zpráv. Pořadí objektů zpráv je stejné jako pořadí, ve kterém budou zprávy načteny z fronty. Aplikace pak může pomocí výčtového nástroje postupně procházet každou zprávu.

Výčtový modul je dynamicky aktualizován, jak jsou zprávy vkládány do fronty a odebírány z fronty. Pokaždé, když aplikace volá IEnumerator.MoveNext() pro procházení další zprávy ve frontě, zpráva odráží aktuální obsah fronty.

Pokud aplikace volá tuto metodu pro prohlížeč front více než jednou, každé volání vrátí nový výčtový modul. Aplikace proto může použít více než jeden výčtový modul k procházení zpráv ve frontě a k udržování více pozic ve frontě.

Parametry:
Není

Vrací:
Objekt iterátoru.

Výjimky:

- Výjimka XMSException

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

Žadatel

Aplikace používá žadatele k odeslání zprávy požadavku a poté čeká na odpověď a obdrží ji.

Hierarchie dědičnosti:
Není

Konstruktory

Žadatel-Vytvořit žadatele

Rozhraní:

```
Requestor(ISession sess, IDestination dest);
```

Vytvořte žadatele.

Parametry:

sess (vstup)
Objekt relace. Relace nesmí být transakční a musí mít jeden z následujících režimů potvrzení:

`AcknowledgeMode.AutoAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

cíl (vstup)
Cílový objekt představující místo určení, kam může aplikace odesílat zprávy požadavků.

Kontext podprocesu:
Relace přidružená k žadateli

Výjimky:

- Výjimka XMSException

metody

Zavřít-zavřít žadatele

Rozhraní:

```
void  Close();
```

Zavřete žadatele.

Pokud se aplikace pokusí zavřít žadatele, který je již zavřený, volání se ignoruje.

Poznámka: Když aplikace zavře žadatele, přidružená relace se také nezavře. V tomto ohledu se produkt XMS chová odlišně ve srovnání s platformou JMS.

Parametry:
Není

Vrací:
Void

Kontext podprocesu:
Libovolný

Výjimky:

- Výjimka XMSException

Požadavek-odezva na požadavek

Rozhraní:

```
IMessage Request(IMessage requestMessage);
```

Odešlete zprávu požadavku a poté počkejte na odpověď od aplikace, která přijme zprávu požadavku, a přijměte ji.

Volání této metody blokuje, dokud není přijata odpověď nebo dokud relace neskončí, podle toho, co nastane dříve.

Parametry:

requestMessage (vstup)
Objekt zprávy zapouzdřující zprávu požadavku.

Vrací:
Ukazatel na objekt zprávy zapouzdřující zprávu odpovědi.

Kontext podprocesu:
Relace přidružená k žadateli

Výjimky:

- Výjimka XMSException

Výjimka ResourceAllocation

XMS vyvolá tuto výjimku, pokud XMS nemůže přidělit prostředky požadované metodou.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.ResourceAllocationException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

SecurityException

Produkt XMS vyvolá tuto výjimku, pokud je identifikátor uživatele a heslo poskytnuté pro ověření aplikace odmítnuto. XMS také vyvolá tuto výjimku, pokud kontrola oprávnění selže a zabrání dokončení metody.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.SecurityException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

IRelace

Relace je kontext s jedním podprocesem pro odesílání a příjem zpráv.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.ISession
```

Seznam XMS definovaných vlastností objektu relace viz “Vlastnosti relace” na stránce 2038.

.NET vlastnosti

AcknowledgeMode -získat režim potvrzení

Rozhraní:

```
AcknowledgeMode AcknowledgeMode
    {
      get;
    }
```

Získejte režim potvrzení pro relaci.

Režim potvrzení je určen při vytvoření relace.

Za předpokladu, že relace není transakční, režim potvrzení má jednu z následujících hodnot:

`AcknowledgeMode.AutoAcknowledge`
`AcknowledgeMode.ClientAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

Další informace o režimech potvrzení naleznete v tématu Potvrzení zprávy.

Relace, která je transakční, nemá žádný režim potvrzení. Je-li relace transakční, metoda vrátí hodnotu `AcknowledgeMode.SessionTransacted` .

Výjimky:

- Výjimka XMSException

Transakce-určit, zda se má jednat o transakci

Rozhraní:

```
Boolean Transacted
    {
      get;
    }
```

Určete, zda je relace transakční.

Uvedené transakce jsou:

- True, pokud je relace transakční.
- False, pokud relace není transakční.

V případě připojení v reálném čase ke zprostředkovateli metoda vždy vrátí hodnotu False.

Výjimky:

- Výjimka XMSException

metody

Zavřít-zavřít relaci

Rozhraní:

```
void Close();
```

Zavřít relaci. Je-li relace transakční, jsou všechny probíhající transakce odvolány.

Pokud se aplikace pokusí zavřít relaci, která je již zavřená, volání se ignoruje.

Parametry:
Není

Vrací:
Void

Kontext podprocesu:
Libovolný

Výjimky:

- Výjimka XMSException

Potvrdit-potvrdit

Rozhraní:

```
void Commit();
```

Potvrdit všechny zprávy zpracované v aktuální transakci.

Relace musí být transakční relací.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState
- TransactionRolledBackException

CreateBrowser -Vytvořit prohlížeč front

Rozhraní:

```
IQueueBrowser CreateBrowser(IDestination queue) ;
```

Vytvořte prohlížeč front pro uvedenou frontu.

Parametry:

fronta (vstup)
Cílový objekt představující frontu.

Vrací:
Objekt QueueBrowser .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination

CreateBrowser -Vytvořit prohlížeč front (se selektorem zpráv)

Rozhraní:

```
IQueueBrowser CreateBrowser(IDestination queue, String selector) ;
```

Vytvořte prohlížeč front pro uvedenou frontu pomocí selektoru zpráv.

Parametry:

fronta (vstup)
Cílový objekt představující frontu.

selektor (vstup)
Objekt typu String zapouzdřující výraz selektoru zpráv. Do prohlížeče fronty jsou doručeny pouze zprávy s vlastnostmi, které odpovídají výrazu selektoru zpráv.

Objekt typu String s hodnotou null znamená, že pro prohlížeč front neexistuje žádný selektor zpráv.

Vrací:
Objekt QueueBrowser .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka InvalidSelector

CreateBytesZpráva-Create Bytes Message

Rozhraní:

```
IBytesMessage CreateBytesMessage();
```

Vytvořit bajtovou zprávu.

Parametry:
Není

Vrací:
Objekt BytesMessage .

Výjimky:

- Výjimka XMSException
- IllegalStateVýjimka (Relace je zavřená)

CreateConsumer -Vytvořit spotřebitele

Rozhraní:

```
IMessageConsumer CreateConsumer(IDestination dest) ;
```

Vytvořte spotřebitele zpráv pro určené místo určení.

Parametry:

cíl (vstup)
Cílový objekt.

Vrací:
Objekt MessageConsumer .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination

CreateConsumer -Vytvořit spotřebitele (se selektorem zpráv)

Rozhraní:

```
IMessageConsumer CreateConsumer(IDestination dest,
                                String selector) ;
```

Vytvořte spotřebitele zpráv pro určené místo určení pomocí selektoru zpráv.

Parametry:

cíl (vstup)
Cílový objekt.

selektor (vstup)
Objekt typu String zapouzdřující výraz selektoru zpráv. Spotřebiteli zpráv jsou doručeny pouze zprávy s vlastnostmi, které odpovídají výrazu selektoru zpráv.

Objekt typu String s hodnotou Null znamená, že pro spotřebitele zpráv neexistuje žádný selektor zpráv.

Vrací:
Objekt MessageConsumer .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka InvalidSelector

CreateConsumer -Vytvořit spotřebitele (se selektorem zpráv a příznakem lokální zprávy)

Rozhraní:

```
IMessageConsumer CreateConsumer(IDestination dest,
                                String selector,
                                Boolean noLocal) ;
```

Vytvořte spotřebitele zpráv pro určené místo určení pomocí selektoru zpráv a v případě, že je cílem téma, určete, zda spotřebitel zpráv přijímá zprávy publikované vlastním připojením.

Parametry:

cíl (vstup)
Cílový objekt.

selektor (vstup)
Objekt typu String zapouzdřující výraz selektoru zpráv. Spotřebiteli zpráv jsou doručeny pouze zprávy s vlastnostmi, které odpovídají výrazu selektoru zpráv.

Objekt typu String s hodnotou Null znamená, že pro spotřebitele zpráv neexistuje žádný selektor zpráv.

noLocal (vstup)
Hodnota True znamená, že spotřebitel zpráv nepřijímá zprávy publikované vlastním připojením. Hodnota False znamená, že spotřebitel zpráv přijímá zprávy publikované vlastním připojením. Výchozí hodnota je false.

Vrací:
Objekt MessageConsumer .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka InvalidSelector

CreateDurableOdběratel-Vytvořit trvalého odběratele

Rozhraní:

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,
                                         String subscription) ;
```

Vytvořte trvalého odběratele pro určené téma.

Tato metoda není platná pro připojení v reálném čase ke zprostředkovateli.

Další informace o trvalých odběratelích naleznete v tématu Trvalí odběratelé.

Parametry:

cíl (vstup)
Cílový objekt představující téma. Téma nesmí být dočasné téma.

odběr (vstup)
Objekt typu String zapouzdřující název, který identifikuje trvalý odběr. Název musí být jedinečný v rámci identifikátoru klienta pro připojení.

Vrací:
Objekt MessageConsumer představující trvalého odběratele.

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination

CreateDurableOdběratel-Vytvořit trvalého odběratele (se selektorem zpráv a příznakem lokální zprávy)

Rozhraní:

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,
                                         String subscription,
                                         String selector,
                                         Boolean noLocal) ;
```

Vytvořte trvalého odběratele pro určené téma pomocí selektoru zpráv a určete, zda trvalý odběratel přijímá zprávy publikované vlastním připojením.

Tato metoda není platná pro připojení v reálném čase ke zprostředkovateli.

Další informace o trvalých odběratelích naleznete v tématu Trvalí odběratelé.

Parametry:

cíl (vstup)
Cílový objekt představující téma. Téma nesmí být dočasné téma.

odběr (vstup)
Objekt typu String zapouzdřující název, který identifikuje trvalý odběr. Název musí být jedinečný v rámci identifikátoru klienta pro připojení.

selektor (vstup)
Objekt typu String zapouzdřující výraz selektoru zpráv. Trvalému odběrateli jsou doručeny pouze zprávy s vlastnostmi, které odpovídají výrazu selektoru zpráv.

Objekt typu String s hodnotou null znamená, že pro trvalého odběratele neexistuje žádný selektor zpráv.

noLocal (vstup)
Hodnota True znamená, že trvalý odběratel neobdrží zprávy publikované vlastním připojením. Hodnota False znamená, že trvalý odběratel přijímá zprávy publikované vlastním připojením. Výchozí hodnota je false.

Vrací:
Objekt MessageConsumer představující trvalého odběratele.

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka InvalidSelector

CreateMapZpráva-Vytvořit zprávu mapy

Rozhraní:

```
IMapMessage CreateMapMessage();
```

Vytvořte zprávu mapy.

Parametry:
Není

Vrací:
Objekt MapMessage .

Výjimky:

- Výjimka XMSException
- IllegalStateVýjimka (Relace je zavřená)

CreateMessage -Vytvořit zprávu

Rozhraní:

```
IMessage CreateMessage();
```

Vytvořte zprávu, která nemá žádné tělo.

Parametry:
Není

Vrací:
Objekt zprávy.

Výjimky:

- Výjimka XMSException
- IllegalStateVýjimka (Relace je zavřená)

CreateObjectZpráva-Vytvořit zprávu objektu

Rozhraní:

```
IObjectMessage CreateObjectMessage();
```

Vytvořit zprávu objektu.

Parametry:
Není

Vrací:
Objekt ObjectMessage .

Výjimky:

- Výjimka XMSException
- IllegalStateVýjimka (Relace je zavřená)

CreateProducer -Vytvořit Producent

Rozhraní:

```
IMessageProducer CreateProducer(IDestination dest) ;
```

Vytvořte producent zpráv pro odesílání zpráv do určeného místa určení.

Parametry:

cíl (vstup)
Cílový objekt.

Zadáte-li cílový objekt s hodnotou Null, bude producent zpráv vytvořen bez místa určení. V tomto případě musí aplikace určit místo určení při každém použití producenta zpráv k odeslání zprávy.

Vrací:
Objekt MessageProducer .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination

CreateQueue -Vytvořit frontu

Rozhraní:

```
IDestination CreateQueue(String queue) ;
```

Vytvořte cílový objekt, který bude představovat frontu na serveru systému zpráv.

Tato metoda nevytváří frontu na serveru systému zpráv. Před voláním této metody aplikací je třeba vytvořit frontu.

Parametry:

fronta (vstup)
Objekt typu String zapouzdřující název fronty nebo zapouzdřující identifikátor URI (Uniform Resource Identifier), který identifikuje frontu.

Vrací:
Cílový objekt představující frontu.

Výjimky:

- Výjimka XMSException

CreateStreamZpráva-Vytvořit zprávu proudu

Rozhraní:

```
IStreamMessage CreateStreamMessage();
```

Vytvořte zprávu proudu.

Parametry:
Není

Vrací:
Objekt StreamMessage .

Výjimky:

- Výjimka XMSException
- XMS_ILLEGAL_STATE_EXCEPTION

CreateTemporaryFronta-Vytvořit dočasnou frontu

Rozhraní:

```
IDestination CreateTemporaryQueue() ;
```

Vytvořte dočasnou frontu.

Rozsah dočasné fronty je připojení. Dočasnou frontu mohou používat pouze relace vytvořené připojením.

Dočasná fronta zůstane, dokud nebude explicitně odstraněna, nebo dokud připojení neskončí, podle toho, co nastane dříve.

Další informace o dočasných frontách naleznete v tématu Dočasné cíle.

Parametry:
Není

Vrací:
Cílový objekt představující dočasnou frontu.

Výjimky:

- Výjimka XMSException

CreateTemporary-téma-Vytvoření dočasného tématu

Rozhraní:

```
IDestination CreateTemporaryTopic() ;
```

Vytvořte dočasné téma.

Rozsahem dočasného tématu je připojení. Dočasné téma mohou používat pouze relace vytvořené připojením.

Dočasné téma zůstává, dokud není explicitně odstraněno, nebo dokud připojení neskončí, podle toho, co nastane dříve.

Další informace o dočasných tématech viz Dočasné cíle.

Parametry:
Není

Vrací:
Cílový objekt představující dočasné téma.

Výjimky:

- Výjimka XMSException

CreateTextZpráva-Vytvořit textovou zprávu

Rozhraní:

```
ITextMessage CreateTextMessage();
```

Vytvořit textovou zprávu s prázdným tělem.

Parametry:
Není

Vrací:
Objekt TextMessage .

Výjimky:

- Výjimka XMSException

CreateTextZpráva-Vytvořit textovou zprávu (inicializováno)

Rozhraní:

```
ITextMessage CreateTextMessage(String initialValue);
```

Vytvořte textovou zprávu, jejíž tělo je inicializováno s uvedeným textem.

Parametry:

initialValue (vstupní)
Objekt typu String zapouzdřující text pro inicializaci těla textové zprávy.

Není

Vrací:
Objekt TextMessage .

Výjimky:

- Výjimka XMSException

CreateTopic -Vytvořit téma

Rozhraní:

```
IDestination CreateTopic(String topic) ;
```

Vytvořte cílový objekt, který bude představovat téma.

Parametry:

téma (vstup)
Objekt typu String zapouzdřující název tématu nebo zapouzdřující identifikátor URI (Uniform Resource Identifier), který dané téma identifikuje.

Vrací:
Cílový objekt představující téma.

Výjimky:

- Výjimka XMSException

Obnovit-obnovit

Rozhraní:

```
void Recover();
```

Obnovit relaci. Doručení zprávy je zastaveno a poté restartováno s nejstarší nepotvrzenou zprávou.

Relace nesmí být transakční relací.

Další informace o obnově relace naleznete v tématu Potvrzení zprávy.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState

Odvolání-odvolání

Rozhraní:

```
void Rollback();
```

Odvolat všechny zprávy zpracované v aktuální transakci.

Relace musí být transakční relací.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState

Odhlásit-Odhlásit

Rozhraní:

```
void Unsubscribe(String subscription);
```

Odstranit trvalý odběr. Server systému zpráv odstraní záznam trvalého odběru, který udržuje, a neodešle žádné další zprávy trvalému odběrateli.

Aplikace nemůže odstranit trvalý odběr za následujících okolností:

- Zatímco existuje aktivní spotřebitel zpráv pro trvalý odběr
- Zatímco je spotřebovaná zpráva součástí nevyřízené transakce
- Zatímco spotřebovaná zpráva nebyla potvrzena

Tato metoda není platná pro připojení v reálném čase ke zprostředkovateli.

Parametry:

odběr (vstup)
Objekt typu String zapouzdřující název, který identifikuje trvalý odběr.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka IllegalState

Zděděné vlastnosti a metody

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

IStreamMessage

Zpráva proudu je zpráva, jejíž tělo se skládá z proudu hodnot, kde každá hodnota má přidružený datový typ. Obsah těla se zapisuje a čte postupně.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IStreamMessage
```

Když aplikace přečte hodnotu z proudu zpráv, může ji produkt XMS převést na jiný datový typ. Další informace o tomto způsobu implicitního převodu viz Tělo XMS zprávy.

metody

ReadBoolean -Čtení logické hodnoty

Rozhraní:

```
Boolean ReadBoolean();
```

Čtení logické hodnoty z proudu zpráv.

Parametry:
Není

Vrací:
Logická hodnota, která se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadByte -Bajt čtení

Rozhraní:

```
Int16   ReadSignedByte();
    Byte    ReadByte();
```

Přečtěte si podepsané 8bitové celé číslo z proudu zpráv.

Parametry:
Není

Vrací:
Bajt, který se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadBytes -Načtené bajty

Rozhraní:

```
Int32   ReadBytes(Byte[] array);
```

Čtení pole bajtů z proudu zpráv.

Parametry:

pole (vstup)
Vyrovnávací paměť obsahující pole přečtených bajtů a délku vyrovnávací paměti v bajtech.

Pokud je počet bajtů v poli menší nebo roven délce vyrovnávací paměti, celé pole se načte do vyrovnávací paměti. Je-li počet bajtů v poli větší než délka vyrovnávací paměti, vyrovnávací paměť se vyplní částí pole a interní kurzor označí pozici dalšího bajtu, který se má číst. Následné volání funkce readBytes() načte bajty z pole počínaje aktuální pozicí kurzoru.

Zadáte-li na vstupu ukazatel s hodnotou null, volání přeskočí pole bajtů, aniž by je přečetlo.

Vrací:
Počet bajtů přečtených do vyrovnávací paměti. Pokud je vyrovnávací paměť částečně vyplněna, hodnota je menší než délka vyrovnávací paměti, což znamená, že v poli zbývají žádné další

bajty pro čtení. Pokud z pole před voláním nezbývají žádné bajty k přečtení, hodnota je XMSC_END_OF_BYTEARRAY.

Zadáte-li na vstupu ukazatel s hodnotou Null, metoda nevrátí žádnou hodnotu.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadChar -Přečtené znaky

Rozhraní:

```
Char    ReadChar();
```

Přečte 2bajtový znak z proudu zpráv.

Parametry:
Není

Vrací:
Znak, který se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadDouble -Přečtěte si číslo s pohyblivou řádovou čárkou s dvojitou přesností.

Rozhraní:

```
Double  ReadDouble();
```

Přečte 8bajtové číslo s pohyblivou řádovou čárkou s dvojitou přesností z proudu zpráv.

Parametry:
Není

Vrací:
Přečtené číslo s pohyblivou řádovou čárkou s dvojitou přesností.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadFloat -Číslo pohyblivé řádové čárky čtení

Rozhraní:

```
Single  ReadFloat();
```

Přečtěte si 4bajtové číslo s pohyblivou řádovou čárkou z proudu zpráv.

Parametry:
Není

Vrací:
Číslo s pohyblivou řádovou čárkou, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadInt -Číst celé číslo

Rozhraní:

```
Int32   ReadInt();
```

Přečtěte si 32bitové celé číslo se znaménkem z proudu zpráv.

Parametry:
Není

Vrací:
Celé číslo, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadLong -Číst dlouhé celé číslo

Rozhraní:

```
Int64   ReadLong();
```

Přečtěte si 64bitové celé číslo se znaménkem z proudu zpráv.

Parametry:
Není

Vrací:
Dlouhé celé číslo, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadObject -Čtení objektu

Rozhraní:

```
Object  ReadObject();
```

Přečtěte si hodnotu z proudu zpráv a vraťte její datový typ.

Parametry:
Není

Vrací:
Hodnota, která je jedním z následujících typů objektů:

```
Boolean
Byte
Byte[]
```

```
Char
Double
Single
Int32
Int64
Int16
String
```

Výjimky:
Výjimka XMSException

ReadShort -Číst krátké celé číslo

Rozhraní:

```
Int16   ReadShort();
```

Přečtěte si podepsané 16bitové celé číslo z proudu zpráv.

Parametry:
Není

Vrací:
Krátké celé číslo, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadString -Čtení řetězce

Rozhraní:

```
String  ReadString();
```

Čtení řetězce z proudu zpráv. V případě potřeby produkt XMS převede znaky v řetězci na lokální kódovou stránku.

Parametry:
Není

Vrací:
Objekt typu String zapouzdřující načtený řetězec. Je-li vyžadován převod dat, jedná se o řetězec po převodu.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

Resetovat-Resetovat

Rozhraní:

```
void  Reset();
```

Vložte tělo zprávy do režimu jen pro čtení a umístěte kurzor na začátek proudu zpráv.

Parametry:
Není

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

WriteBoolean -Hodnota logického zápisu

Rozhraní:

```
void  WriteBoolean(Boolean value);
```

Zapsat logickou hodnotu do proudu zpráv.

Parametry:

hodnota (vstup)
Logická hodnota, která se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteByte -Bajt zápisu

Rozhraní:

```
void  WriteByte(Byte value);
void  WriteSignedByte(Int16 value);
```

Zapsat bajt do proudu zpráv.

Parametry:

hodnota (vstup)
Bajt, který se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteBytes -Zapsané bajty

Rozhraní:

```
void  WriteBytes(Byte[] value);
```

Zapsat pole bajtů do proudu zpráv.

Parametry:

hodnota (vstup)
Pole bajtů, které se mají zapsat.

délka (vstup)
Počet bajtů v poli.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteChar -Znak zápisu

Rozhraní:

```
void  WriteChar(Char value);
```

Zapsat znak do proudu zpráv jako 2 bajty, nejprve bajt vysokého pořadí.

Parametry:

hodnota (vstup)
Znak, který se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteDouble -číslo s pohyblivou řádovou čárkou a dvojitou přesností pro zápis.

Rozhraní:

```
void  WriteDouble(Double value);
```

Převést číslo s pohyblivou řádovou čárkou s dvojitou přesností na dlouhé celé číslo a zapsat dlouhé celé číslo do proudu zpráv jako 8 bajtů, nejprve bajt s vysokým pořadovým číslem.

Parametry:

hodnota (vstup)
Číslo s pohyblivou řádovou čárkou s dvojitou přesností, které má být zapsáno.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteFloat -Číslo plovoucí řádové čárky pro zápis

Rozhraní:

```
void  WriteFloat(Single value);
```

Převést číslo s pohyblivou řádovou čárkou na celé číslo a zapsat celé číslo do proudu zpráv jako 4 bajty, nejprve bajt s vysokým pořadovým číslem.

Parametry:

hodnota (vstup)
Číslo s pohyblivou řádovou čárkou, které se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteInt -celé číslo pro zápis

Rozhraní:

```
void  WriteInt(Int32 value);
```

Zapsat celé číslo do proudu zpráv jako 4 bajty, nejprve bajt s vyšším pořadovým číslem.

Parametry:

hodnota (vstup)
Celé číslo, které se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteLong -Zapsat dlouhé celé číslo

Rozhraní:

```
void  WriteLong(Int64 value);
```

Zapsat dlouhé celé číslo do proudu zpráv jako 8 bajtů, nejvyšší pořadí bajtů jako první.

Parametry:

hodnota (vstup)
Dlouhé celé číslo, které se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteObject -Zapsat objekt

Rozhraní:

```
void  WriteObject(Object value);
```

Zapište hodnotu s uvedeným datovým typem do proudu zpráv.

Parametry:

objectType (vstup)
Hodnota, která musí být jedním z následujících typů objektů:

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

hodnota (vstup)
Pole bajtů obsahující hodnotu, která má být zapsána.

délka (vstup)
Počet bajtů v poli.

Vrací:
Void

Výjimky:

- Výjimka XMSException

WriteShort -Zapsat krátké celé číslo

Rozhraní:

```
void  WriteShort(Int16 value);
```

Zapsat krátké celé číslo do proudu zpráv jako 2 bajty, nejprve jako bajt vysokého pořadí.

Parametry:

hodnota (vstup)
Krátké celé číslo, které se má zapsat.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteString -Zápis řetězce

Rozhraní:

```
void  WriteString(String value);
```

Zapsat řetězec do proudu zpráv.

Parametry:

hodnota (vstup)
Objekt typu String zapouzdřující řetězec, který má být zapsán.

Vrací:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Zděděné vlastnosti a metody

Níže uvedené vlastnosti jsou zděděny z rozhraní IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType,Properties

Níže uvedené metody jsou zděděny z rozhraní IMessage:

clearBody, clearProperties, PropertyExists

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

ITextMessage

Textová zpráva je zpráva, jejíž tělo obsahuje řetězec.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.ITextMessage
```

.NET vlastnosti

Text-Získat a nastavit text

Rozhraní:

```
String Text
{
  get;
  set;
}
```

Získejte a nastavte řetězec, který tvoří tělo textové zprávy.

V případě potřeby produkt XMS převede znaky v řetězci na lokální kódovou stránku.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageNotWritableException
- MessageEOFException

Zděděné vlastnosti a metody

Níže uvedené vlastnosti jsou zděděny z rozhraní IMessage:

JMSCorrelationID, JMSDeliveryMode, JMSDestination, JMSExpiration, JMSMessageID, JMSPriority, JMSRedelivered, JMSReplyTo, JMSTimestamp, JMSType,Properties

Níže uvedené metody jsou zděděny z rozhraní IMessage:

clearBody, clearProperties, PropertyExists

Níže uvedené metody jsou zděděny z rozhraní IPropertyContext:

GetBooleanProperty, GetByteProperty, GetBytesProperty, GetCharProperty, GetDoubleProperty, GetFloatProperty, GetIntProperty, GetLongProperty, GetObjectProperty, GetShortProperty, GetStringProperty, SetBooleanProperty, SetByteProperty, SetBytesProperty, SetCharProperty, SetDoubleProperty, SetFloatProperty, SetIntProperty, SetLongProperty, SetObjectProperty, SetShortProperty, SetStringProperty

TransactionInProgressException

XMS vyvolá tuto výjimku, pokud aplikace požaduje operaci, která není platná, protože probíhá transakce.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
            |
            +----IBM.XMS.TransactionInProgressException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

TransactionRolledBackException

XMS vyvolá tuto výjimku, pokud aplikace volá funkci Session.commit() k potvrzení aktuální transakce, ale transakce je poté odvolána.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
            |
            +----IBM.XMS.TransactionRolledBackException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedVýjimka

Výjimka XMSException

Pokud produkt XMS zjistí chybu při zpracování volání metody .NET , XMS vygeneruje výjimku. Výjimkou je objekt, který zapouzdřuje informace o chybě.

Hierarchie dědičnosti:

```
System.Exception
   |
   +----IBM.XMS.XMSException
```

Existují různé typy výjimky XMS a objekt XMSException je pouze jedním typem výjimky. Třída XMSException je však supertřídou ostatních tříd výjimek XMS . XMS vyvolá objekt XMSException v situacích, kdy žádný z ostatních typů výjimek není vhodný.

.NET vlastnosti

ErrorCode -Kód chyby získání

Rozhraní:

```
public String ErrorCode
{
  get {return errorCode_;}
}
```

Získejte kód chyby.

Výjimky:

- Výjimka XMSException

LinkedException -Získat propojenou výjimku

Rozhraní:

```
public Exception LinkedException
{
  get { return linkedException_;}
  set { linkedException_ = value;}
}
```

Získejte další výjimku v řetězci výjimek.

Metoda vrací hodnotu null, pokud v řetězci nejsou žádné další výjimky.

Výjimky:

- Výjimka XMSException

XMSFactoryFactory

Pokud aplikace nepoužívá spravované objekty, použijte tuto třídu k vytvoření továren připojení, front a témat.

Hierarchie dědičnosti:
Není

.NET vlastnosti

Metadata-načtení metadat

Rozhraní:

```
IConnectionMetaData MetaData
```

Získejte metadata odpovídající typu připojení objektu XMSFactoryFactory .

Výjimky:
Není

metody

CreateConnectionTovárna-Vytvořit továrnu připojení

Rozhraní:

```
IConnectionFactory CreateConnectionFactory();
```

Vytvořte objekt ConnectionFactory deklarovaného typu.

Parametry:
Není

Vrací:
Objekt ConnectionFactory .

Výjimky:

- Výjimka XMSException

CreateQueue -Vytvořit frontu

Rozhraní:

```
IDestination CreateQueue(String name);
```

Vytvořte cílový objekt, který bude představovat frontu na serveru systému zpráv.

Tato metoda nevytváří frontu na serveru systému zpráv. Před voláním této metody aplikací je třeba vytvořit frontu.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název fronty nebo zapouzdřující identifikátor URI (Uniform Resource Identifier), který identifikuje frontu.

Vrací:
Cílový objekt představující frontu.

Výjimky:

- Výjimka XMSException

CreateTopic -Vytvořit téma

Rozhraní:

```
IDestination CreateTopic(String name);
```

Vytvořte cílový objekt, který bude představovat téma.

Parametry:

název (vstup)
Objekt typu String zapouzdřující název tématu nebo zapouzdřující identifikátor URI (Uniform Resource Identifier), který dané téma identifikuje.

Vrací:
Cílový objekt představující téma.

Výjimky:

- Výjimka XMSException

GetInstance -Získat instanci XMSFactoryFactory

Rozhraní:

```
static XMSFactoryFactory GetInstance(int connectionType);
```

Vytvořte instanci XMSFactoryFactory. Aplikace XMS používá objekt XMSFactoryFactory k získání odkazu na objekt ConnectionFactory , který odpovídá požadovanému typu protokolu. Tento objekt ConnectionFactory pak může vytvářet připojení pouze pro tento typ protokolu.

Parametry:

connectionType (vstup)
Typ připojení, pro který objekt ConnectionFactory vytváří připojení:

- XMSC.CT_WPM
- XMSC.CT_RTT
- XMSC.CT_WMQ

Vrací:
Objekt XMSFactoryFactory vyhrazený pro deklarovaný typ připojení.

Výjimky:

- Výjimka NotSupported

Vlastnosti objektů XMS

Tato sekce dokumentuje vlastnosti objektu definované pomocí XMS.

Tato sekce obsahuje informace o následujících typech objektů:

- “Vlastnosti připojení” na stránce 2026
- “Vlastnosti ConnectionFactory” na stránce 2026
- “Vlastnosti dat ConnectionMeta” na stránce 2031
- “Vlastnosti místa určení” na stránce 2031
- “Vlastnosti InitialContext” na stránce 2032
- “Vlastnosti zprávy” na stránce 2033
- “Vlastnosti objektu MessageConsumer” na stránce 2038
- “Vlastnosti prvku MessageProducer” na stránce 2038
- “Vlastnosti relace” na stránce 2038

Popis každého typu objektu uvádí vlastnosti objektu určeného typu a poskytuje krátký popis každé vlastnosti.

Tento oddíl také poskytuje definici každé vlastnosti (viz “Definice vlastností” na stránce 2038).

Pokud aplikace definuje své vlastní vlastnosti objektů popsaných v této sekci, nezpůsobí chybu, ale může způsobit nepředvídatelné výsledky.

Poznámka: Názvy a hodnoty vlastností v této sekci jsou zobrazeny ve formátu XMSC.*NAME*, což je formulář používaný pro C a C + +. V produktu .NETvšak může být tvar názvu vlastnosti buď XMSC.*NAME* , nebo XMSC_*NAME*, v závislosti na tom, jak jej používáte:

- Pokud zadáváte vlastnost, název vlastnosti musí být ve tvaru XMSC.*NAME* , jak ukazuje následující příklad:

```
cf.SetStringProperty(XMSC.WMQ_CHANNEL, "DOTNET.SVRCONN");
```

- Pokud zadáváte řetězec, název vlastnosti musí být ve formátu XMSC_*NAME* , jak ukazuje následující příklad:

```
cf.SetStringProperty("XMSC_WMQ_CHANNEL", "DOTNET.SVRCONN");
```

  V produktu .NETjsou názvy a hodnoty vlastností poskytovány jako konstanty ve třídě XMSC. Tyto konstanty identifikují řetězce a budou použity jakoukoli aplikací XMS .NET . Pokud používáte tyto předdefinované konstanty, názvy a hodnoty vlastností jsou ve formátu XMSC.*NAME*, takže byste například použili XMSC.USERID, nikoli XMSC_USERID.

Datové typy jsou také ve formátu používaném pro C/C + +. Odpovídající hodnoty pro .NET najdete v části Datové typy pro .NET.

Vlastnosti připojení

Přehled vlastností objektu připojení s odkazy na podrobnější referenční informace.

Tabulka 872. Vlastnosti připojení

Název majetku	Popis
“XMSC_WMQ_RESOLVED_QUEUE_MANAGER” na stránce 2072	Tato vlastnost se používá k získání názvu správce front, ke kterému je připojena.
“XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID” na stránce 2072	Tato vlastnost je po připojení naplněna ID správce front.
XMSC_WPM_CONNECTION_PROTOCOL	Komunikační protokol použitý pro připojení k jádru systému zpráv. Tato vlastnost je jen pro čtení.
XMSC_WPM_HOST_NAME	Název hostitele nebo adresa IP systému, který obsahuje jádro systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.
XMSC_WPM_ME_NAME	Název jádra systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.
XMSC_WPM_PORT	Číslo portu naslouchaného v jádru systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.

Objekt připojení má také vlastnosti jen pro čtení, které jsou odvozeny z vlastností továrny připojení, která byla použita k vytvoření připojení. Tyto vlastnosti jsou odvozeny nejen z vlastností továrny připojení, které byly nastaveny v době vytvoření připojení, ale také z výchozích hodnot vlastností, které nebyly nastaveny. Vlastnosti zahrnují pouze ty, které jsou relevantní pro typ serveru systému zpráv, ke kterému je aplikace připojena. Názvy vlastností jsou stejné jako názvy vlastností továrny připojení.

Vlastnosti ConnectionFactory

Přehled vlastností objektu ConnectionFactory s odkazy na podrobnější referenční informace.

Tabulka 873. Vlastnosti ConnectionFactory

Název majetku	Popis
“XMSC_ASYNC_EXCEPTIONS” na stránce 2048	Tato vlastnost určuje, zda rozhraní XMS informuje modul ExceptionListener pouze v případě, že dojde k přerušení připojení nebo když dojde k asynchronnímu výskytu jakékoli výjimky pro volání rozhraní XMS API. Tato vlastnost platí pro všechna připojení vytvořená z této továrny připojení, která mají registrován modul ExceptionListener.

Tabulka 873. Vlastnosti ConnectionFactory (pokračování)	
Název majetku	**Popis**
V 9.3.0 “XMSC_WMQ_BALANCING_APPLICATION_TYPE” na stránce 2056	Typ volby vyvážení
V 9.3.0 “XMSC_WMQ_BALANCING_OPTIONS” na stránce 2057	Volby vyvážení nastavené vydávající aplikací
V 9.3.0 “XMSC_WMQ_BALANCING_TIMEOUT” na stránce 2057	Vypršel časový limit, po kterém může nové vyvážení přerušit aktivitu aplikace.
XMSC_CLIENT_ID	Identifikátor klienta pro účely připojení.
XMSC_CONNECTION_TYPE	Typ serveru systému zpráv, ke kterému se aplikace připojuje.
XMSC_PASSWORD	Heslo, které lze použít k ověření aplikace při pokusu o připojení k serveru systému zpráv.
“XMSC_RTT_BROKER_PING_INTERVAL” na stránce 2053	Časový interval v milisekundách, po který XMS .NET kontroluje připojení k serveru systému zpráv v reálném čase ke zjištění jakékoli aktivity.
XMSC_RTT_CONNECTION_PROTOCOL	Komunikační protokol použitý pro připojení v reálném čase ke zprostředkovateli.
XMSC_RTT_HOST_NAME	Název nebo adresa IP hostitele systému, na kterém je spuštěn zprostředkovatel.
XMSC_RTT_LOCAL_ADDRESS	Název hostitele nebo adresa IP lokálního síťového rozhraní, které má být použito pro připojení v reálném čase ke zprostředkovateli.
XMSC_RTT_MULTICAST	Nastavení výběrového vysílání pro továrnu připojení nebo cíl.
XMSC_RTT_PORT	Číslo portu, na kterém zprostředkovatel naslouchá příchozím požadavkům.
XMSC_USERID	Identifikátor uživatele, který lze použít k ověření aplikace při pokusu o připojení k serveru systému zpráv.
XMSC_WMQ_BROKER_CONTROLQ	Název řídicí fronty používané zprostředkovatelem.
XMSC_WMQ_BROKER_PUBQ	Název fronty monitorované zprostředkovatelem, kde aplikace odesílají zprávy, které publikují.
XMSC_WMQ_BROKER_QMGR	Název správce front, ke kterému je zprostředkovatel připojen.
XMSC_WMQ_BROKER_SUBQ	Název fronty odběratele pro spotřebitele netrvalých zpráv.
XMSC_WMQ_BROKER_VERSION	Typ zprostředkovatele použitý aplikací pro připojení nebo pro cíl.
“XMSC_WMQ_CCDTURL” na stránce 2059	Adresa URL (Uniform Resource Locator) identifikující název a umístění souboru, který obsahuje tabulku definic kanálů klienta, a také určuje, jakým způsobem lze k souboru přistupovat.

Tabulka 873. Vlastnosti ConnectionFactory (pokračování)	
Název majetku	**Popis**
XMSC_WMQ_CHANNEL	Název kanálu, který se má použít pro připojení.
“XMSC_WMQ_CLIENT_RECONNECT_OPTIONS” na stránce 2060	Tato vlastnost určuje volby opětovného připojení klienta pro nová připojení vytvořená touto továrnou.
“XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT” na stránce 2061	Tato vlastnost určuje dobu v sekundách, po kterou se připojení klienta pokusí znovu připojit.
XMSC_WMQ_CONNECTION_MODE	Režim, pomocí kterého se aplikace připojuje ke správci front.
“XMSC_WMQ_CONNECTION_NAME_LIST” na stránce 2061	Tato vlastnost určuje hostitele, ke kterým se klient pokusí znovu připojit po přerušení připojení.
XMSC_WMQ_FAIL_IF_QUIESCE	Zda se volání konkrétních metod nezdaří, pokud je správce front, k němuž je aplikace připojen, ve stavu uvedení do klidového stavu.
XMSC_WMQ_HOST_NAME	Název nebo adresa IP hostitele systému, na kterém je spuštěn správce front.
XMSC_WMQ_LOCAL_ADDRESS	Pro připojení ke správci front tato vlastnost určuje rozhraní lokální sítě, lokálního portu nebo rozsahu lokálních portů, nebo obojí.
XMSC_WMQ_MESSAGE_SELECTION	Určuje, zda výběr zpráv provádí klient XMS nebo zprostředkovatel.
XMSC_WMQ_MSG_BATCH_SIZE	Maximální počet zpráv, které mají být načteny z fronty v jedné dávce, v případě asynchronního doručování zpráv.
XMSC_WMQ_POLLING_INTERVAL	Pokud každý modul listener pro zprávy v rámci relace nemá ve frontě žádnou odpovídající zprávu, jedná se o maximální interval v milisekundách, který uplyne předtím, než se každý modul listener pro zprávy znovu pokusí získat zprávu z příslušné fronty.
“XMSC_WMQ_PROVIDER_VERSION” na stránce 2070	Verze, vydání, úroveň modifikace a opravný balík správce front, ke kterému se tato aplikace hodlá připojovat.
XMSC_WMQ_PORT	Číslo portu, na kterém správce front naslouchá příchozím požadavkům.
XMSC_WMQ_PUB_ACK_INTERVAL	Počet zpráv publikovaných vydavatelem, než klient XMS vyžádá potvrzení od zprostředkovatele.
“XMSC_WMQ_PUT_ASYNC_ALLOWED” na stránce 2065	Tato vlastnost určuje, zda producenti zpráv mohou používat k odesílání zpráv do tohoto místa určení asynchronní operace put.
XMSC_WMQ_QMGR_CCSID	Identifikátor (CCSID) kódované znakové sady nebo kódové stránky, ve které jsou pole znakových dat definovaná v rozhraní MQI (Message Queue Interface) vyměňována mezi klientem XMS a klientem IBM MQ .
XMSC_WMQ_QUEUE_MANAGER	Název správce front, s nímž má být navázáno připojení.
XMSC_WMQ_RECEIVE_EXIT	Identifikuje uživatelskou proceduru pro přijetí zprávy kanálu, která má být spuštěna.

Tabulka 873. Vlastnosti ConnectionFactory (pokračování)	
Název majetku	**Popis**
XMSC_WMQ_RECEIVE_EXIT_INIT	Uživatelská data, která jsou předána uživatelské proceduře pro přijetí zprávy kanálu při volání.
XMSC_WMQ_SECURITY_EXIT	Identifikuje uživatelskou proceduru pro zabezpečení zprávy kanálu.
XMSC_WMQ_SECURITY_EXIT_INIT	Uživatelská data, která jsou předána uživatelské proceduře pro zabezpečení zprávy kanálu při volání.
"XMSC_WMQ_SEND_CHECK_COUNT" na stránce 2074	Počet povolených odeslaných volání mezi kontrolou asynchronních chyb vložení během jedné netransakční relace XMS.
XMSC_WMQ_SEND_EXIT	Identifikuje uživatelskou proceduru pro odeslání zprávy kanálu.
XMSC_WMQ_SEND_EXIT_INIT	Uživatelská data, která jsou předána uživatelským procedurám pro odeslání zprávy kanálu při volání.
"XMSC_WMQ_SHARE_CONV_ALLOWED" na stránce 2074	Zda může připojení klienta sdílet svůj soket s jinými připojeními nejvyšší úrovně XMS ze stejného procesu ke stejnému správci front, pokud se shodují definice kanálů. Tato vlastnost slouží k povolení úplné izolace produktu Connections v samostatných soketech, pokud je to vyžadováno pro vývoj aplikací, údržbu nebo provozní důvody.
XMSC_WMQ_SSL_CERT_STORES	Umístění serverů obsahující seznamy odvolaných certifikátů (CRL), které mají být použity v připojení SSL ke správci front.
XMSC_WMQ_SSL_CIPHER_SPEC	Název specifikace CipherSpec, která má být použita v zabezpečeném připojení ke správci front.
XMSC_WMQ_SSL_CIPHER_SUITE	Název sady CipherSuite, která má být použita v rámci připojení TLS ke správci front. Protokol použitý při vyjednávání zabezpečeného připojení závisí na určené sadě CipherSuite.
XMSC_WMQ_SSL_CRYPTO_HW	Podrobnosti konfigurace šifrovacího hardwaru připojeného k systému klienta.
XMSC_WMQ_SSL_FIPS_REQUIRED	Hodnota této vlastnosti určuje, zda aplikace může nebo nemůže používat šifrovací sady, které nevyhovují standardu FIPS. Je-li tato vlastnost nastavena na hodnotu true, pro připojení klientského serveru se používají pouze algoritmy FIPS.
XMSC_WMQ_SSL_KEY_REPOSITORY	Umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty.
XMSC_WMQ_SSL_KEY_RESETCOUNT	Hodnota KeyResetCount představuje celkový počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace SSL, než je znovu vyjednán tajný klíč.
XMSC_WMQ_SSL_PEER_NAME	Název typu peer, které se použije při připojení SSL ke správci front.

Tabulka 873. Vlastnosti ConnectionFactory (pokračování)	
Název majetku	**Popis**
XMSC_WMQ_SYNCPOINT_ALL_GETS	Určuje, zda musí být všechny zprávy načteny z front v rámci řízení synchronizačního bodu.
“XMSC_WMQ_TARGET_CLIENT” na stránce 2081	
XMSC_WMQ_TEMP_Q_PREFIX	Předpona použitá k vytvoření názvu dynamické fronty IBM MQ , která se vytvoří, když aplikace vytvoří XMS dočasnou frontu.
XMSC_WMQ_TEMP_TOPIC_PREFIX	Při vytváření dočasných témat produkt XMS vygeneruje řetězec tématu ve tvaru "`TEMP/TEMPTOPICPREFIX/unique_id`", nebo pokud tato vlastnost obsahuje výchozí hodnotu, vygeneruje se tento řetězec "`TEMP/unique_id`". Určení neprázdné hodnoty umožní definování specifických modelových front za účelem vytvoření spravovaných front pro odběratele dočasných témat vytvořených v rámci tohoto připojení.
XMSC_WMQ_TEMPORARY_MODEL	Název modelové fronty IBM MQ , ze které se vytvoří dynamická fronta, když aplikace vytvoří XMS dočasnou frontu.
XMSC_WPM_BUS_NAME	U továrny připojení se jedná o název sběrnice SIBus, ke které se aplikace připojuje, nebo pro cíl, název sběrnice SIBus, ve které cíl existuje.
XMSC_WPM_CONNECTION_PROXIMITY	Nastavení blízkosti připojení pro připojení.
XMSC_WPM_DUR_SUB_HOME	Název jádra systému zpráv, v nichž jsou spravovány všechny trvalé odběry pro připojení nebo cíl.
XMSC_WPM_LOCAL_ADDRESS	Pro připojení ke sběrnici SIBus tato vlastnost určuje rozhraní lokální sítě, lokálního portu nebo rozsahu lokálních portů, nebo obojí.
XMSC_WPM_NON_PERSISTENT_MAP	Úroveň spolehlivosti přechodných zpráv, které se odesílají pomocí připojení.
XMSC_WPM_PERSISTENT_MAP	Úroveň spolehlivosti trvalých zpráv, které se odesílají pomocí připojení.
XMSC_WPM_PROVIDER_KONCOVÉ body	Sekvence jednoho nebo více adres koncového bodu zaváděcích serverů.
XMSC_WPM_TARGET_GROUP	Název cílové skupiny jader systému zpráv.
XMSC_WPM_TARGET_VÝZNAM	Významnost cílové skupiny jader systému zpráv.
XMSC_WPM_TARGET_TRANSPORT_CHAIN	Název transportu příchozích požadavků, který musí aplikace používat pro připojení k jádru systému zpráv.
XMSC_WPM_TARGET_TYPE	Typ cílové skupiny jader systému zpráv.
XMSC_WPM_TEMP_Q_PREFIX	Předpona použitá k vytvoření názvu dočasné fronty, která se vytvoří ve sběrnici pro integraci služeb, když aplikace vytvoří XMS dočasnou frontu.
XMSC_WPM_TEMP_TOPIC_PREFIX	Předpona používaná k vytvoření názvu dočasného tématu vytvořeného aplikací.

Vlastnosti dat ConnectionMeta

Přehled vlastností datového objektu ConnectionMetas odkazy na podrobnější referenční informace.

Tabulka 874. Vlastnosti dat ConnectionMeta	
Název majetku	**Popis**
XMSC_JMS_MAJOR_VERSION	Číslo hlavní verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.
XMSC_JMS_MINOR_VERSION	Číslo vedlejší verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.
XMSC_JMS_VERSION	Identifikátor verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.
XMSC_MAJOR_VERSION	Číslo verze klienta XMS . Tato vlastnost je jen pro čtení.
XMSC_MINOR_VERSION	Číslo vydání klienta XMS . Tato vlastnost je jen pro čtení.
XMSC_PROVIDER_NAME	Poskytovatel klienta XMS . Tato vlastnost je jen pro čtení.
XMSC_VERSION	Identifikátor verze rozhraní cliXMSent. Tato vlastnost je jen pro čtení.

Vlastnosti místa určení

Přehled vlastností cílového objektu s odkazy na podrobnější referenční informace.

Tabulka 875. Vlastnosti místa určení	
Název majetku	**Popis**
XMSC_DELIVERY_MODE	Režim doručení zpráv odeslaných do cíle.
XMSC_PRIORITY	Priorita zpráv odeslaných do cíle.
XMSC_RTT_MULTICAST	Nastavení výběrového vysílání pro továrnu připojení nebo cíl.
XMSC_TIME_TO_LIVE	Doba životnosti zpráv odeslaných do cíle.
XMSC_WMQ_BROKER_VERSION	Typ zprostředkovatele použitý aplikací pro připojení nebo pro cíl.
XMSC_WMQ_CCSID	Identifikátor (CCSID) kódované znakové sady nebo kódové stránky, ve které jsou řetězce znakových dat v těle zprávy, když klient XMS předá zprávu do místa určení.
XMSC_WMQ_DUR_SUBQ	Název fronty odběratele pro trvalého odběratele, který přijímá zprávy z cíle. **Poznámka:** Tuto vlastnost lze použít s verzí 2.0 produktu IBM Message Service Client for .NET , ale nemá žádný vliv na aplikaci připojenou ke správci front IBM WebSphere MQ 7.0 , pokud není vlastnost XMSC_WMQ_PROVIDER_VERSION továrny připojení nastavena na číslo verze nižší než 7.
XMSC_WMQ_ENCODING	Způsob reprezentace číselných dat v těle zprávy v případě, že klient produktu XMS předá zprávu do místa určení.
XMSC_WMQ_FAIL_IF_QUIESCE	Zda se volání konkrétních metod nezdaří, pokud je správce front, k němuž je aplikace připojen, ve stavu uvedení do klidového stavu.

Tabulka 875. Vlastnosti místa určení (pokračování)	
Název majetku	**Popis**
“XMSC_WMQ_MESSAGE_BODY” na stránce 2063	Tato vlastnost určuje, zda aplikace XMS zpracovává MQRFH2 zprávy IBM MQ jako součást informačního obsahu zprávy (tj. jako součást těla zprávy).
“XMSC_WMQ_MQMD_MESSAGE_CONTEXT” na stránce 2064	Určuje, jaká úroveň kontextu zprávy má být nastavena aplikací XMS . Aby tato vlastnost mohla nabýt účinnosti, musí aplikace běžet s příslušným oprávněním kontextu.
“XMSC_WMQ_MQMD_READ_ENABLED” na stránce 2065	Tato vlastnost určuje, zda může aplikace XMS extrahovat hodnoty polí MQMD.
“XMSC_WMQ_MQMD_WRITE_ENABLED” na stránce 2065	Tato vlastnost určuje, zda může aplikace XMS nastavit hodnoty polí MQMD.
“XMSC_WMQ_READ_AHEAD_ALLOWED” na stránce 2066	Tato vlastnost určuje, zda mohou příjemci zpráv a zprostředkovatelé front používat dopředné čtení k načtení netrvalých netransakčních zpráv z tohoto cíle do interní vyrovnávací paměti před jejich přijetím.
“XMSC_WMQ_READ_AHEAD_CLOSE_POLICY” na stránce 2066	Tato vlastnost určuje, zda budou zprávy doručovány do asynchronního modulu listener zpráv, co se stane se zprávami v interní vyrovnávací paměti pro čtení při zavření spotřebitele zpráv.
“XMSC_WMQ_RECEIVE_CCSID” na stránce 2071	Cílová vlastnost, která nastavuje cílový CCSID pro převod zpráv správce front. Hodnota je ignorována, pokud není parametr XMSC_WMQ_RECEIVE_CONVERSION nastaven na hodnotu WMQ_RECEIVE_CONVERSION_QMGR.
“XMSC_WMQ_RECEIVE_CONVERSION” na stránce 2071	Cílová vlastnost, která určuje, zda má správce front provést převod dat.
XMSC_WMQ_TARGET_CLIENT	Určuje, zda zprávy odeslané do cíle obsahují záhlaví MQRFH2.
XMSC_WMQ_TEMP_TOPIC_PREFIX	Při vytváření dočasných témat produkt XMS vygeneruje řetězec tématu ve tvaru "`TEMP/TEMPTOPICPREFIX/unique_id`", nebo pokud tato vlastnost obsahuje výchozí hodnotu, vygeneruje se tento řetězec "`TEMP/unique_id`". Určení neprázdné hodnoty umožní definování specifických modelových front za účelem vytvoření spravovaných front pro odběratele dočasných témat vytvořených v rámci tohoto připojení.
XMSC_WPM_BUS_NAME	U továrny připojení se jedná o název sběrnice SIBus, ke které se aplikace připojuje, nebo pro cíl, název sběrnice SIBus, ve které cíl existuje.
XMSC_WPM_TOPIC_SPACE	Název prostoru témat, který obsahuje dané téma.

Vlastnosti InitialContext

Přehled vlastností objektu InitialContext s odkazy na podrobnější referenční informace.

Tabulka 876. Vlastnosti InitialContext	
Název majetku	**Popis**
XMSC_IC_PROVIDER_URL	Použije se k vyhledání adresáře pojmenování rozhraní JNDI tak, aby služba pojmenování COS nevyžadovala být na stejném serveru jako webová služba.
XMSC_IC_SECURITY_AUTHENTICATION	Na základě Java Kontextového rozhraní SECURITY_AUTHENTICATION. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.
XMSC_IC_SECURITY_CREDENTIALS	Na základě Java Kontextového rozhraní SECURITY_CREDENTIALS. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.
XMSC_IC_SECURITY_PRINCIPAL	Na základě Java Kontextového rozhraní SECURITY_PRINCIPAL. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.
XMSC_IC_SECURITY_PROTOCOL	Na základě Java kontextového rozhraní SECURITY_PROTOCOL Tato vlastnost je použitelná pouze pro kontext pojmenování COS.
XMSC_IC_URL	Pro kontexty LDAP a FileSystem, adresa úložiště obsahující administrované objekty. Pro kontexty pojmenování COS, adresa webové služby, která vyhledává objekty v adresáři.

Vlastnosti zprávy

Přehled vlastností objektu Zpráva s odkazy na podrobnější referenční informace.

Tabulka 877. Vlastnosti zprávy	
Název majetku	**Popis**
JMS_IBM_CHARACTER_SET	Identifikátor (CCSID) kódované znakové sady nebo kódové stránky, ve které jsou řetězce znakových dat v těle zprávy, když klient XMS předá zprávu do svého zamýšleného místa určení. V produktu XMS má tato vlastnost číselnou hodnotu a mapuje se na CCSID. Tato vlastnost je však založena na vlastnosti JMS, takže má řetězcovou hodnotu typu a mapuje se na znakovou sadu Java , která představuje tento numerický identifikátor CCSID.
JMS_IBM_Encoding	Způsob reprezentace číselných dat v těle zprávy v případě, že klient produktu XMS předá zprávu do zamýšleného místa určení.
JMS_IBM_EXCEPTIONMESSAGE	Text, který popisuje, proč byla zpráva odeslána do cíle výjimek. Tato vlastnost je jen pro čtení.
JMS_IBM_EXCEPTIONPROBLEMDESTINATION	Název cíle, ve kterém byla zpráva před odesláním zprávy do cíle výjimek.
JMS_IBM_EXCEPTIONREASON	Kód příčiny označující příčinu, proč byla zpráva odeslána do cíle výjimek.
JMS_IBM_EXCEPTIONTIMESTAMP	Čas, kdy byla zpráva odeslána do cíle výjimek.
JMS_IBM_FEEDBACK	Kód, který označuje chování zprávy sestavy.
JMS_IBM_FORMAT	Chování dat aplikace ve zprávě.

Tabulka 877. Vlastnosti zprávy (pokračování)	
Název majetku	**Popis**
JMS_IBM_LAST_MSG_IN_GROUP	Označuje, zda je zpráva poslední zprávou ve skupině zpráv.
JMS_IBM_MSGTYPE	Typ zprávy.
JMS_IBM_PUTAPPLTYPE	Typ aplikace, která odeslala zprávu.
JMS_IBM_PUTDATE	Datum, kdy byla zpráva odeslána.
JMS_IBM_PUTTIME	Čas, kdy byla zpráva odeslána.
JMS_IBM_REPORT_COA	Požadavek na zprávy sestavy 'potvrdit při příjmu' určující, kolik dat aplikace z původní zprávy musí být zahrnuto do zprávy sestavy.
JMS_IBM_REPORT_COD	Požadavek na zprávy sestavy 'potvrdit při doručení' určující, kolik dat aplikace z původní zprávy musí být zahrnuto do zprávy sestavy.
JMS_IBM_REPORT_DISCARD_MSG	Požadavek, aby byla zpráva vyřazena, pokud ji nelze doručit do zamýšleného cíle.
VÝJIMKA-JMS_IBM_REPORT_EXCEPTION	Požadavek na zprávy sestavy výjimky určující, kolik dat aplikace z původní zprávy musí být zahrnuto do zprávy sestavy.
JMS_IBM_REPORT_EXPIRATION	Požadavek na zprávy sestavy vypršení platnosti určující, kolik dat aplikace z původní zprávy musí být zahrnuto do zprávy sestavy.
JMS_IBM_REPORT_NAN	Požadavek na zprávy sestavy negativního oznámení na akci.
JMS_IBM_REPORT_PAN	Požadavek na zprávy sestavy pozitivního oznámení na akci.
JMS_IBM_REPORT_PASS_CORREL_ID	Požadavek, aby identifikátor korelace jakékoli sestavy nebo zprávy odpovědi byl stejný jako identifikátor korelace původní zprávy.
JMS_IBM_REPORT_PASS_MSG_ID	Požadavek, aby identifikátor zprávy jakékoli sestavy nebo zprávy odpovědi byl stejný jako identifikátor zprávy původní zprávy.
JMS_IBM_RETAIN	Nastavení této vlastnosti označuje, správce front bude zprávu považovat za Zachované publikování.
JMS_IBM_SYSTEM_MESSAGEID	Identifikátor, který jedinečně identifikuje zprávu v rámci sběrnice SIBus. Tato vlastnost je jen pro čtení.
JMSX_APPID	Název aplikace, která odeslala zprávu.
JMSX_DELIVERY_COUNT	Počet pokusů o doručení zprávy.
JMSX_GROUPID	Identifikátor skupiny zpráv, do které zpráva patří.
JMSX_GROUPSEQ	Pořadové číslo zprávy v rámci skupiny zpráv.
JMSX_USERID	Identifikátor uživatele přidružený k aplikaci, která odeslala zprávu.

Vlastnosti JMS_IBM_MQMD*

Produkt IBM Message Service Client for .NET umožňuje klientským aplikacím číst/zapisovat pole MQMD pomocí rozhraní API. Také umožňuje přístup k datům zprávy produktu MQ . Standardně je přístup k MQMD zakázán a aplikace jej musí explicitně povolit pomocí vlastností místa určení XMSC_WMQ_MQMD_WRITE_ENABLED a XMSC_WMQ_MQMD_READ_ENABLED. Tyto dvě vlastnosti jsou na sobě nezávislé.

Všechna pole MQMD s výjimkou polí StrucId a Version jsou vystavena jako další vlastnosti objektu zprávy a mají předponu JMS_IBM_MQMD.

Vlastnosti JMS_IBM_MQMD* mají vyšší přednost před jinými vlastnostmi, jako je JMS_IBM*, popsanými v předchozí tabulce.

Odesílání zpráv

Jsou reprezentována všechna pole MQMD s výjimkou polí StrucId a Version. Tyto vlastnosti odkazují pouze na pole MQMD. V případě, že se vlastnost vyskytuje jak v záhlaví MQMD, tak v záhlaví MQRFH2 , není verze v MQRFH2 nastavena nebo extrahována. Lze nastavit libovolnou z těchto vlastností s výjimkou JMS_IBM_MQMD_BackoutCount. Jakákoli hodnota nastavená pro JMS_IBM_MQMD_BackoutCount je ignorována.

Pokud má vlastnost maximální délku a zadáte hodnotu, která je příliš dlouhá, hodnota se ořízne.

Pro určité vlastnosti musíte také nastavit vlastnost XMSC_WMQ_MQMD_MESSAGE_CONTEXT na cílovém objektu. Aby tato vlastnost mohla nabýt účinnosti, musí aplikace běžet s příslušným oprávněním kontextu. Pokud nenastavíte parametr XMSC_WMQ_MQMD_MESSAGE_CONTEXT na odpovídající hodnotu, bude hodnota vlastnosti ignorována. Pokud nastavíte parametr XMSC_WMQ_MQMD_MESSAGE_CONTEXT na příslušnou hodnotu, ale nemáte dostatečná kontextová oprávnění pro správce front, dojde k výjimce. Vlastnosti vyžadující specifické hodnoty XMSC_WMQ_MQMD_MESSAGE_CONTEXT jsou následující.

Následující vlastnosti vyžadují nastavení parametru XMSC_WMQ_MQMD_MESSAGE_CONTEXT na hodnotu XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT nebo XMSC_WMQ_MDCTX_SET_ALL_CONTEXT:

- JMS_IBM_MQMD_UserIdentifier
- JMS_IBM_MQMD_AccountingToken
- Data JMS_IBM_MQMD_ApplIdentity

Následující vlastnosti vyžadují nastavení parametru XMSC_WMQ_MQMD_MESSAGE_CONTEXT na hodnotu XMSC_WMQ_MDCTX_SET_ALL_CONTEXT:

- Typ JMS_IBM_MQMD_PutAppl
- Název JMS_IBM_MQMD_PutAppl
- JMS_IBM_MQMD_PutDate
- JMS_IBM_MQMD_PutTime
- Data JMS_IBM_MQMD_ApplOrigin

Příjem zpráv

Všechny tyto vlastnosti jsou k dispozici v přijaté zprávě, pokud je vlastnost XMSC_WMQ_MQMD_READ_ENABLED nastavena na hodnotu true, bez ohledu na skutečné vlastnosti, které jsou nastaveny produkující aplikací. Aplikace nemůže upravit vlastnosti přijaté zprávy, pokud nejsou nejprve vymazány všechny vlastnosti v souladu se specifikací JMS. Přijatá zpráva může být postoupena bez úpravy vlastností.

Poznámka: Pokud vaše aplikace obdrží zprávu z místa určení s vlastností XMSC_WMQ_MQMD_READ_ENABLED nastavenou na hodnotu true a předá ji do místa určení s hodnotou XMSC_WMQ_MQMD_WRITE_ENABLED nastavenou na hodnotu true, bude výsledkem zkopírování všech hodnot polí MQMD přijaté zprávy do předané zprávy. Tabulka vlastností

Tabulka 878. Vlastnosti objektu zprávy reprezentující pole MQMD

Vlastnost	Popis	Typ
JMS_IBM_MQMD_REPORT	Volby pro zprávy sestavy	System.Int32
JMS_IBM_MQMD_MSGTYPE	Typ zprávy	System.Int32
JMS_IBM_MQMD_EXPIRY	životnost zprávy	System.Int32
JMS_IBM_MQMD_FEEDBACK	Zpětná vazba nebo kód příčiny	System.Int32
JMS_IBM_MQMD_ENCODING	Číselné kódování dat zprávy	System.Int32
JMS_IBM_MQMD_CODEDCHARSETID	Identifikátor znakové sady dat zprávy	System.Int32
JMS_IBM_MQMD_FORMAT	Název formátu dat zprávy	System.String
JMS_IBM_MQMD_PRIORITY **Poznámka:** Pokud přiřadíte hodnotu JMS_IBM_MQMD_PRIORITY, která není v rozsahu 0-9, tato hodnota porušuje specifikaci JMS.	Priorita zprávy	System.Int32
JMS_IBM_MQMD_PERSISTENCE	Trvalost zpráv	System.Int32
JMS_IBM_MQMD_MSGID **Poznámka:** Specifikace JMS uvádí, že ID zprávy musí být nastaveno poskytovatelem JMS a že musí být buď jedinečné, nebo null. Pokud přiřadíte hodnotu k JMS_IBM_MQMD_MSGID, tato hodnota se zkopíruje do JMSMessageID. Proto není nastaven poskytovatelem JMS a nemusí být jedinečný: tato hodnota porušuje specifikaci JMS.	Identifikátor zprávy	Bajtové pole **Poznámka:** Použití vlastností bajtového pole ve zprávě porušuje specifikaci JMS.
JMS_IBM_MQMD_CORRELID **Poznámka:** Pokud přiřadíte hodnotu JMS_IBM_MQMD_CORRELID, která začíná řetězcem 'ID:', tato hodnota porušuje specifikaci JMS.	Identifikátor korelace	Bajtové pole **Poznámka:** Použití vlastností bajtového pole ve zprávě porušuje specifikaci JMS.
JMS_IBM_MQMD_BACKOUTCOUNT	Počítadlo odvolaných	System.Int32
JMS_IBM_MQMD_REPLYTOQ	Název fronty odpovědí	System.String
JMS_IBM_MQMD_REPLYTOQMGR	Název správce front odpovědí	System.String
JMS_IBM_MQMD_USERIDENTIFIER	Identifikátor uživatele	System.String
JMS_IBM_MQMD_ACCOUNTINGTOKEN	Token evidence	Bajtové pole **Poznámka:** Použití vlastností bajtového pole ve zprávě porušuje specifikaci JMS.
JMS_IBM_MQMD_APPLIDENTITYDATA	Údaje aplikace týkající se identity	System.String
JMS_IBM_MQMD_PUTAPPLTYPE	Typ aplikace, která vložila zprávu	System.Int32

Tabulka 878. Vlastnosti objektu zprávy reprezentující pole MQMD (pokračování)		
Vlastnost	**Popis**	**Typ**
JMS_IBM_MQMD_PUTAPPLNAME	Název aplikace, která vložila zprávu	System.String
JMS_IBM_MQMD_PUTDATE	Datum, kdy byla zpráva vložena	System.String
JMS_IBM_MQMD_PUTTIME	Čas, kdy byla zpráva vložena	System.String
JMS_IBM_MQMD_APPLORIGINDATA	Údaje o žádosti týkající se původu	System.String
JMS_IBM_MQMD_GROUPID	Identifikátor skupiny	Bajtové pole **Poznámka:** Použití vlastností bajtového pole ve zprávě porušuje specifikaci JMS.
JMS_IBM_MQMD_MSGSEQNUMBER	Pořadové číslo lokální zprávy ve skupině	System.Int32
JMS_IBM_MQMD_OFFSET	Posunutí dat ve fyzické zprávě od začátku logické zprávy	System.Int32
JMS_IBM_MQMD_MSGFLAGS	Příznaky zprávy	System.Int32
JMS_IBM_MQMD_ORIGINALLENGTH	Délka původní zprávy	System.Int32

Další podrobnosti viz MQMD .

Příklady

Tento příklad má za následek vložení zprávy do fronty nebo tématu s produktem MQMD.UserIdentifier nastaven na "JoeBloggs".

```
// Create a ConnectionFactory, connection, session, producer, message
// ...

// Create a destination
// ...

// Enable MQMD write
dest.setBooleanProperty(XMSC_WMQ_MQMD_WRITE_ENABLED,
      XMSC_WMQ_MQMD_WRITE_ENABLED_YES);

// Optionally, set a message context if applicable for this MD field
dest.setIntProperty(XMSC_WMQ_MQMD_MESSAGE_CONTEXT,
  XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT);

// On the message, set property to provide custom UserId
msg.setStringProperty(JMS_IBM_MQMD_USERIDENTIFIER, "JoeBloggs");

// Send the message
// ...
```

Před nastavením parametru JMS_IBM_MQMD_USERIDENTIFIER je nutné nastavit parametr XMSC_WMQ_MQMD_MESSAGE_CONTEXT. Další informace o použití příkazu XMSC_WMQ_MQMD_MESSAGE_CONTEXT naleznete ve vlastnostech objektu zprávy.

Podobně můžete extrahovat obsah polí MQMD nastavením parametru XMSC_WMQ_MQMD_READ_ENABLED na hodnotu true před přijetím zprávy a poté pomocí metod get zprávy, například vlastnosti getString. Všechny přijaté vlastnosti jsou jen pro čtení.

Tento příklad způsobí, že pole hodnoty bude obsahovat hodnotu MQMD.ApplIdentityData zprávy získané z fronty nebo tématu.

```
// Create a ConnectionFactory, connection, session, consumer
// ...

// Create a destination
// ...

// Enable MQMD read
dest.setBooleanProperty(XMSC_WMQ_MQMD_READ_ENABLED, XMSC_WMQ_MQMD_READ_ENABLED_YES);

// Receive a message
// ...

// Get required MQMD field value using a property
System.String value = rcvMsg.getStringProperty(JMS_IBM_MQMD_APPLIDENTITYDATA);
```

Vlastnosti objektu MessageConsumer

Přehled vlastností objektu MessageConsumer s odkazy na podrobnější referenční informace.

Tabulka 879. Vlastnosti objektu MessageConsumer

Název majetku	Popis
XMSC_IS_SUBSCRIPTION_MULTICAST	Označuje, zda jsou zprávy doručovány spotřebiteli zpráv pomocí WebSphere MQ Multicast Transport. Tato vlastnost je jen pro čtení.
XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST	Označuje, zda jsou zprávy doručovány spotřebiteli zpráv pomocí produktu WebSphere MQ Multicast Transport se spolehlivou kvalitou služby. Tato vlastnost je jen pro čtení.

Viz .Další podrobnosti naleznete ve vlastnostech NET objektu IMessageConsumer .

Vlastnosti prvku MessageProducer

Přehled vlastností objektu MessageProducer s odkazy na podrobnější referenční informace.

Viz .Další podrobnosti viz vlastnosti NET IMessageProducer .

Vlastnosti relace

Přehled vlastností objektu relace s odkazy na podrobnější referenční informace.

Viz .Další podrobnosti o vlastnostech NET relace ISession .

Definice vlastností

Tato sekce poskytuje definici každé vlastnosti objektu.

Každá definice vlastnosti obsahuje následující informace:

- Datový typ vlastnosti.
- Typy objektů, které mají vlastnost
- Pro vlastnost Destination se jedná o název, který lze použít v identifikátoru URI (Uniform Resource Identifier).
- Podrobnější popis nemovitosti
- Platné hodnoty vlastnosti
- Výchozí hodnota vlastnosti.

Vlastnosti, jejichž názvy začínají jednou z následujících předpon, jsou relevantní pouze pro uvedený typ připojení:

XMSC_RTT
Vlastnosti jsou relevantní pouze pro připojení v reálném čase ke zprostředkovateli. Názvy vlastností jsou definovány jako pojmenované konstanty v hlavičkovém souboru `xmsc_rtt.h`.

XMSC_WMQ
Vlastnosti jsou relevantní pouze v případě, že se aplikace připojuje ke správci front IBM MQ . Názvy vlastností jsou definovány jako pojmenované konstanty v hlavičkovém souboru `xmsc_wmq.h`.

XMSC_WPM
Vlastnosti jsou relevantní pouze v případě, že se aplikace připojuje ke sběrnici pro integraci služeb WebSphere . Názvy vlastností jsou definovány jako pojmenované konstanty v hlavičkovém souboru `xmsc_wpm.h`.

Není-li v jejich definicích uvedeno jinak, zbývající vlastnosti jsou relevantní pro všechny typy připojení. Názvy vlastností jsou definovány jako pojmenované konstanty v hlavičkovém souboru `xmsc.h`. Vlastnosti, jejichž názvy začínají předponou JMSX, jsou JMS definované vlastnosti zprávy a vlastnosti, jejichž názvy začínají předponou JMS_IBM, jsou IBM definované vlastnosti zprávy. Další informace o vlastnostech zpráv naleznete v tématu Vlastnosti XMS zprávy.

Není-li ve své definici uvedeno jinak, je každá vlastnost relevantní jak v doméně odběru typu point-to-point, tak v doméně odběru publikování.

Aplikace může získat a nastavit hodnotu libovolné vlastnosti, pokud není vlastnost označena jako jen pro čtení.

JMS_IBM_CHARACTER_SET

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Identifikátor (CCSID) kódované znakové sady nebo kódové stránky, ve které jsou řetězce znakových dat v těle zprávy, když klient XMS předá zprávu do svého zamýšleného místa určení. V produktu XMS má tato vlastnost číselnou hodnotu a mapuje se na CCSID. Tato vlastnost je však založena na vlastnosti JMS, takže má řetězcovou hodnotu typu a mapuje se na znakovou sadu Java , která představuje tento numerický identifikátor CCSID. Tato vlastnost potlačuje jakýkoli CCSID zadaný pro místo určení vlastností XMSC_WMQ_CCSID .

Standardně není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_Encoding

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Způsob reprezentace číselných dat v těle zprávy v případě, že klient produktu XMS předá zprávu do zamýšleného místa určení. Tato vlastnost potlačuje jakékoli kódování určené pro místo určení vlastností XMSC_WMQ_ENCODING . Tato vlastnost určuje reprezentaci binárních celých čísel, pakovaných desetinných celých čísel a čísel s pohyblivou řádovou čárkou.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **Encoding** deskriptoru zprávy.

Aplikace může k nastavení vlastnosti použít následující pojmenované konstanty:

pojmenovaná konstanta	Význam
MQENC_INTEGER_NORMAL	Normální kódování celých čísel
MQENC_INTEGER_REVERSED	Obrácené kódování celých čísel

pojmenovaná konstanta	Význam
MQENC_DECIMAL_NORMAL	Normální pakované kódování desetinných čísel
MQENC_DECIMAL_REVERSED	Obrácené pakované kódování desetinných čísel
MQENC_FLOAT_IEEE_NORMAL	Normální kódování čísel s pohyblivou řádovou čárkou IEEE
MQENC_FLOAT_IEEE_REVERSED	Obrácené kódování čísel s pohyblivou řádovou čárkou IEEE
MQENC_FLOAT_S390	Kódování pohyblivé řádové čárky architektury z/OS
MQENC_NATIVE	Nativní kódování počítače

Chcete-li vytvořit hodnotu pro vlastnost, může aplikace přidat tři z těchto konstant takto:

- Konstanta, jejíž název začíná hodnotou MQENC_INTEGER, která určuje reprezentaci binárních celých čísel.
- Konstanta, jejíž název začíná na MQENC_DECIMAL, pro určení reprezentace pakovaných desetinných celých čísel.
- Konstanta, jejíž název začíná na MQENC_FLOAT, pro určení reprezentace čísel s pohyblivou řádovou čárkou.

Alternativně může aplikace nastavit vlastnost na MQENC_NATIVE, jejíž hodnota je závislá na prostředí.

Standardně není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

Zpráva JMS_IBM_EXCEPTIONMESSAGE

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Text, který popisuje, proč byla zpráva odeslána do cíle výjimek. Tato vlastnost je jen pro čtení.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojí ke sběrnici pro integraci služeb a obdrží zprávu z cíle výjimek.

JMS_IBM_EXCEPTIONPROBLEMDESTINATION

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Název cíle, ve kterém byla zpráva před odesláním zprávy do cíle výjimek.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojí ke sběrnici pro integraci služeb a obdrží zprávu z cíle výjimek.

JMS_IBM_EXCEPTIONREASON

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Kód příčiny označující příčinu, proč byla zpráva odeslána do cíle výjimek.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojí ke sběrnici pro integraci služeb a obdrží zprávu z cíle výjimek.

ČASOVÉ razítko JMS_IBM_EXCEPTIONTIMESTAMP

Datový typ:
System.Int64

Majetek společnosti:
Zpráva

Čas, kdy byla zpráva odeslána do cíle výjimek.

Čas je vyjádřen v milisekundách od 00:00:00 GMT 1. ledna 1970.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojí ke sběrnici pro integraci služeb a obdrží zprávu z cíle výjimek.

ZPĚTNÁ vazba jms_ibm_feedback

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Kód, který označuje chování zprávy sestavy.

Platné hodnoty vlastnosti jsou kódy zpětné vazby a kódy příčiny, které lze zadat v poli **Feedback** deskriptoru zprávy.

Standardně není vlastnost nastavena.

JMS_IBM_FORMAT

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Chování dat aplikace ve zprávě.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **Format** deskriptoru zprávy.

Standardně není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_LAST_MSG_IN_GROUP

Datový typ:
System.Boolean

Majetek společnosti:
Zpráva

Označuje, zda je zpráva poslední zprávou ve skupině zpráv.

Nastavte vlastnost na hodnotu true, pokud je zpráva poslední zprávou ve skupině zpráv. Jinak nastavte vlastnost na hodnotu false nebo ji nenastavujte. Standardně není vlastnost nastavena.

Hodnota true odpovídá stavovému příznaku MQMF_LAST_MSG_IN_GROUP, který lze zadat v poli **MsgFlags** deskriptoru zprávy.

Tato vlastnost je v doméně publikování/odběru ignorována a není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_MSGTYPE

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Typ zprávy.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
MQMT_DATAGRAM	Zpráva je taková, která nevyžaduje odpověď.
MQMT_REQUEST	Zpráva je zpráva, která vyžaduje odpověď.
MQMT_REPLY	Zpráva je zpráva odpovědi.
MQMT_REPORT	Zpráva je zpráva sestavy.

Tyto hodnoty odpovídají typům zpráv, které lze zadat v poli **MsgType** deskriptoru zprávy.

Standardně není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_PUTAPPLTYPE

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Typ aplikace, která odeslala zprávu.

Platné hodnoty vlastnosti jsou typy aplikací, které lze zadat v poli **PutApplType** deskriptoru zprávy.

Standardně není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_PUTDATE

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Datum, kdy byla zpráva odeslána.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **PutDate** deskriptoru zprávy.

Standardně není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_PUTTIME

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Čas, kdy byla zpráva odeslána.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **PutTime** deskriptoru zprávy.

Standardně není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_REPORT_COA

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek na zprávy sestavy 'potvrdit při příjmu' určující, kolik dat aplikace z původní zprávy musí být zahrnuto do zprávy sestavy.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
MQRO_COA	Požadujte zprávu sestavy "potvrdit při příchodu" bez dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_COA_WITH_DATA	Vyžádat zprávy sestavy 'confirm on arrival' s prvními 100 bajty dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_COA_WITH_FULL_DATA	Vyžádat 'potvrzení při příchodu' zprávy sestavy se všemi daty aplikace z původní zprávy zahrnutými ve zprávě sestavy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zprávy. Další informace o těchto volbách viz Sestava (MQLONG).

Standardně není vlastnost nastavena.

JMS_IBM_REPORT_COD

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek na zprávy sestavy 'potvrdit při doručení' určující, kolik dat aplikace z původní zprávy musí být zahrnuto do zprávy sestavy.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
MQRO_COD	Požadovat 'potvrzení při doručení' zpráv sestavy, bez dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_COD_WITH_DATA	Vyžádat 'potvrzení při doručení' zpráv sestavy s prvními 100 bajty dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_COD_WITH_FULL_DATA	Požadovat zprávu "potvrdit při doručení" se všemi daty aplikace z původní zprávy zahrnutými ve zprávě sestavy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zprávy.

Standardně není vlastnost nastavena.

JMS_IBM_REPORT_DISCARD_MSG

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek, aby byla zpráva vyřazena, pokud ji nelze doručit do zamýšleného cíle.

Nastavte vlastnost na hodnotu MQRO_DISCARD_MSG, chcete-li požadovat, aby byla zpráva vyřazena, pokud ji nelze doručit do zamýšleného místa určení. Požadujete-li vložení zprávy do fronty nedoručených zpráv nebo odeslání do cíle výjimek, nenastavujte vlastnost. Standardně není vlastnost nastavena.

Hodnota MQRO_DISCARD_MSG odpovídá volbě sestavy, kterou lze zadat v poli **Report** deskriptoru zprávy.

VÝJIMKA-JMS_IBM_REPORT_EXCEPTION

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek na zprávy sestavy výjimky určující, kolik dat aplikace z původní zprávy musí být zahrnuto do zprávy sestavy.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
MQRO_EXCEPTION	Zprávy sestavy výjimek požadavků bez dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_EXCEPTION_WITH_DATA	Zprávy sestavy výjimek požadavků s prvními 100 bajty dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_EXCEPTION_WITH_FULL_DATA	Zprávy sestavy výjimek požadavků se všemi daty aplikace z původní zprávy zahrnutými ve zprávě sestavy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zprávy.

Standardně není vlastnost nastavena.

JMS_IBM_REPORT_EXPIRATION

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek na zprávy sestavy vypršení platnosti určující, kolik dat aplikace z původní zprávy musí být zahrnuto do zprávy sestavy.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
MQRO_EXPIRATION	Zprávy sestavy vypršení platnosti požadavku bez dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.

Platná hodnota	Význam
MQRO_EXPIRATION_WITH_DATA	Zprávy sestavy vypršení platnosti požadavku s prvními 100 bajty dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_EXPIRATION_WITH_FULL_DATA	Zprávy sestavy vypršení platnosti požadavku se všemi daty aplikace z původní zprávy zahrnutými ve zprávě sestavy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zprávy.

Standardně není vlastnost nastavena.

JMS_IBM_REPORT_NAN

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek na zprávy sestavy negativního oznámení na akci.

Nastavte vlastnost na hodnotu MQRO_NAN, chcete-li požadovat zprávy sestavy oznámení negativní akce. Pokud nevyžadujete negativní zprávy sestavy oznámení akce, nenastavujte vlastnost. Standardně není vlastnost nastavena.

Hodnota MQRO_NAN odpovídá volbě sestavy, kterou lze zadat v poli **Report** deskriptoru zprávy.

JMS_IBM_REPORT_PAN

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek na zprávy sestavy pozitivního oznámení na akci.

Nastavte vlastnost na hodnotu MQRO_PAN, chcete-li požadovat zprávy sestavy oznámení pozitivní akce. Pokud nevyžadujete kladné zprávy sestavy oznámení akce, nenastavujte vlastnost. Standardně není vlastnost nastavena.

Hodnota MQRO_PAN odpovídá volbě sestavy, kterou lze zadat v poli **Report** deskriptoru zprávy.

JMS_IBM_REPORT_PASS_CORREL_ID

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek, aby identifikátor korelace jakékoli sestavy nebo zprávy odpovědi byl stejný jako identifikátor korelace původní zprávy.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
MQRO_PASS_CORREL_ID	Požadavek, aby identifikátor korelace jakékoli sestavy nebo zprávy odpovědi byl stejný jako identifikátor korelace původní zprávy.

Platná hodnota	Význam
MQRO_COPY_MSG_ID_TO_CORREL_ID	Požadavek, aby identifikátor korelace jakékoli zprávy sestavy nebo odpovědi byl stejný jako identifikátor zprávy původní zprávy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zprávy.

Výchozí hodnota vlastnosti je MQRO_COPY_MSG_ID_TO_CORREL_ID.

JMS_IBM_REPORT_PASS_MSG_ID

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Požadavek, aby identifikátor zprávy jakékoli sestavy nebo zprávy odpovědi byl stejný jako identifikátor zprávy původní zprávy.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
MQRO_PASS_MSG_ID	Požadavek, aby identifikátor zprávy jakékoli sestavy nebo zprávy odpovědi byl stejný jako identifikátor zprávy původní zprávy.
MQRO_NEW_MSG_ID	Požádejte o vygenerování nového identifikátoru zprávy pro každou zprávu sestavy nebo zprávu odpovědi.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli Sestava deskriptoru zprávy.

Výchozí hodnota vlastnosti je MQRO_NEW_MSG_ID.

JMS_IBM_RETAIN

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Nastavení této vlastnosti označuje, správce front bude zprávu považovat za Zachované publikování. Když odběratel obdrží zprávy z témat, může okamžitě po přihlášení k odběru obdržet další zprávy nad rámec zpráv přijatých v předchozích vydáních. Tyto zprávy jsou volitelná zachovaná publikování pro odebíraná témata. Pro každé téma odpovídající odběru je v případě zachovaného publikování k dispozici publikování pro doručení odběrateli zpráv.

RETAIN_PUBLICATION je jedinou platnou hodnotou této vlastnosti. Standardně není tato vlastnost nastavena.

Poznámka: Tato vlastnost je relevantní pouze v doméně publikování/odběru.

ID_SYSTÉMOVÉ_ZPRÁVY JMS_IBM_SYSTEM_MESSAGEID

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Identifikátor, který jedinečně identifikuje zprávu v rámci sběrnice SIBus. Tato vlastnost je jen pro čtení.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke sběrnici pro integraci služeb.

JMSX_APPID

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Název aplikace, která odeslala zprávu.

Tato vlastnost je vlastností JMS definovanou s názvem JMS JMSXAppID. Další informace o vlastnosti viz *Java Specifikace služby systému zpráv, verze 1.1*.

Standardně není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

POČET_DORUČENÍ_ZPRÁV

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Počet pokusů o doručení zprávy.

Tato vlastnost je JMS definovaná vlastnost s názvem JMS JMSXDeliveryCount. Další informace o vlastnosti viz *Java Message Service Specifikace, verze 1.1*.

Standardně není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

JMSX_GROUPID

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Identifikátor skupiny zpráv, do které zpráva patří.

Tato vlastnost je definovaná vlastnost JMS s názvem JMS name JMSXGroupID. Další informace o vlastnosti viz *Java Message Service Specifikace, verze 1.1*.

Standardně není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

JMSX_GROUPSEQ

Datový typ:
System.Int32

Majetek společnosti:
Zpráva

Pořadové číslo zprávy v rámci skupiny zpráv.

Tato vlastnost je JMS definovaná vlastnost s názvem JMS JMSXGroupSeq. Další informace o vlastnosti viz *Java Message Service Specifikace, verze 1.1*.

Standardně není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

JMSX_USERID

Datový typ:
Řetězec

Majetek společnosti:
Zpráva

Identifikátor uživatele přidružený k aplikaci, která odeslala zprávu.

Tato vlastnost je definovaná vlastnost JMS s názvem JMS name JMSXUserID. Další informace o vlastnosti viz *Java Message Service Specifikace, verze 1.1*.

Standardně není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

XMSC_ASYNC_EXCEPTIONS

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: ASYNCEXCEPTION

Krátký název nástroje pro administraci JMS: AEX

Tato vlastnost určuje, zda rozhraní XMS informuje modul ExceptionListener pouze v případě, že dojde k přerušení připojení nebo když dojde k asynchronnímu výskytu jakékoli výjimky pro volání rozhraní XMS API. Tato vlastnost platí pro všechna připojení vytvořená z této továrny připojení, která mají registrován modul ExceptionListener.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_ASYNC_EXCEPTIONS_ALL
Všechny výjimky zjištěné asynchronně mimo rozsah synchronního volání rozhraní API a všechny výjimky s přerušením připojení jsou odeslány do modulu ExceptionListener.

XMSC_ASYNC_EXCEPTIONS_CONNECTIONBROKEN
Do modulu ExceptionListenerjsou odesílány pouze výjimky označující přerušené připojení. Jakékoli další výjimky, které se vyskytnou během asynchronního zpracování, nejsou hlášeny modulu ExceptionListener, a proto aplikace není o těchto výjimkách informována.

Standardně je tato vlastnost nastavena na hodnotu XMSC_ASYNC_EXCEPTIONS_ALL.

XMSC_CLIENT_ID

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CLIENTID

Krátký název nástroje pro administraci JMS: CID

Identifikátor klienta pro účely připojení.

Identifikátor klienta se používá pouze pro podporu trvalých odběrů v doméně publikování/odběru a v doméně typu point-to-point se ignoruje. Další informace o nastavení identifikátorů klienta naleznete v tématu ConnectionFactories a objekty připojení.

Tato vlastnost není relevantní pro připojení v reálném čase ke zprostředkovateli.

XMSC_CONNECTION_TYPE

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Typ serveru systému zpráv, ke kterému se aplikace připojuje.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_CT_RTT	Připojení ke zprostředkovateli v reálném čase.
XMSC_CT_WMQ	Připojení ke správci front IBM MQ .
XMSC_CT_WPM	Připojení k serveru WebSphere Application Server service integration bus.

Standardně není vlastnost nastavena.

XMSC_REŽIM_DORUČENÍ

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Název použitý v identifikátoru URI:

`Perzistence` (pro cíl IBM MQ)
`deliveryMode` (pro výchozí místo určení poskytovatele systému zpráv WebSphere)

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: PERSISTENCE

Krátký název nástroje pro administraci JMS: PER

Režim doručení zpráv odeslaných do cíle.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_DELIVERY_NOT_PERSISTENT	Zpráva odeslaná do místa určení je dočasná. Výchozí režim doručení producenta zpráv nebo jakýkoli jiný režim doručení zadaný ve volání Send je ignorován. Pokud je cílem fronta IBM MQ , hodnota atributu fronty *DefPersistence* se také ignoruje.
XMSC_DELIVERY_PERSISTENT	Zpráva odeslaná do místa určení je trvalá. Výchozí režim doručení producenta zpráv nebo jakýkoli jiný režim doručení zadaný ve volání Send je ignorován. Pokud je cílem fronta IBM MQ , hodnota atributu fronty *DefPersistence* se také ignoruje.
XMSC_DELIVERY_AS_APP	Pro zprávu odeslanou do místa určení je nastaven režim doručení ve volání Send. Pokud volání Send neuvádí žádný režim doručení, použije se místo toho výchozí režim doručení producenta zpráv. Pokud je cílem fronta IBM MQ , hodnota atributu fronty *DefPersistence* se ignoruje.

Platná hodnota	Význam
XMSC_DELIVERY_AS_DEST	Pokud je cílem fronta IBM MQ , má zpráva vložená do fronty režim doručení určený hodnotou atributu fronty *DefPersistence*. Výchozí režim doručení producenta zpráv nebo jakýkoli jiný režim doručení zadaný ve volání Send je ignorován. Pokud cíl není frontou IBM MQ , význam je stejný jako u XMSC_DELIVERY_AS_APP.

Výchozí hodnota je XMSC_DELIVERY_AS_APP.

XMSC_IC_PROVIDER_URL

Datový typ:
Řetězec

Majetek společnosti:
InitialContext

Použije se k vyhledání adresáře pojmenování rozhraní JNDI tak, aby služba pojmenování COS nevyžadovala být na stejném serveru jako webová služba.

XMSC_IC_SECURITY_AUTHENTICATION

Datový typ:
Řetězec

Majetek společnosti:
InitialContext

Na základě Java Kontextového rozhraní SECURITY_AUTHENTICATION. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.

XMSC_IC_SECURITY_CREDENTIALS

Datový typ:
Řetězec

Majetek společnosti:
InitialContext

Na základě Java Kontextového rozhraní SECURITY_CREDENTIALS. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.

XMSC_IC_SECURITY_PRINCIPAL

Datový typ:
Řetězec

Majetek společnosti:
InitialContext

Na základě Java Kontextového rozhraní SECURITY_PRINCIPAL. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.

XMSC_IC_SECURITY_PROTOCOL

Datový typ:
Řetězec

Majetek společnosti:
InitialContext

Na základě Java kontextového rozhraní SECURITY_PROTOCOL Tato vlastnost je použitelná pouze pro kontext pojmenování COS.

XMSC_IC_URL

Datový typ:
Řetězec

Majetek společnosti:
InitialContext

Pro kontexty LDAP a FileSystem, adresa úložiště obsahující administrované objekty.

Pro kontexty LDAP a FileSystem, adresa úložiště obsahující administrované objekty.

XMSC_IS_SUBSCRIPTION_MULTICAST

Datový typ:
System.Boolean

Majetek společnosti:
MessageConsumer

Označuje, zda jsou zprávy doručovány spotřebiteli zpráv pomocí WebSphere MQ Multicast Transport. Tato vlastnost je jen pro čtení.

Hodnota vlastnosti je true, pokud jsou zprávy doručovány spotřebiteli zpráv pomocí WebSphere MQ Multicast Transport. Jinak je hodnota false.

Tato vlastnost je relevantní pouze pro připojení v reálném čase ke zprostředkovateli.

VÝBĚROVÉ vysílání XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST

Datový typ:
System.Boolean

Majetek společnosti:
MessageConsumer

Označuje, zda jsou zprávy doručovány spotřebiteli zpráv pomocí produktu WebSphere MQ Multicast Transport se spolehlivou kvalitou služby. Tato vlastnost je jen pro čtení.

Hodnota vlastnosti je true, pokud jsou zprávy doručovány spotřebiteli zpráv pomocí produktu WebSphere MQ Multicast Transport se spolehlivou kvalitou služby. Jinak je hodnota false.

Tato vlastnost je relevantní pouze pro připojení v reálném čase ke zprostředkovateli.

XMSC_JMS_MAJOR_VERSION

Datový typ:
System.Int32

Majetek společnosti:
ConnectionMetaData

Číslo hlavní verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.

XMSC_JMS_MINOR_VERSION

Datový typ:
System.Int32

Majetek společnosti:
ConnectionMetaData

Číslo vedlejší verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.

XMSC_JMS_VERSION

Datový typ:
Řetězec

Majetek společnosti:
ConnectionMetaData

Identifikátor verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.

XMSC_MAJOR_VERSION

Datový typ:
System.Int32

Majetek společnosti:
ConnectionMetaData

Číslo verze klienta XMS . Tato vlastnost je jen pro čtení.

XMSC_MINOR_VERSION

Datový typ:
System.Int32

Majetek společnosti:
ConnectionMetaData

Číslo vydání klienta XMS . Tato vlastnost je jen pro čtení.

XMSC_PASSWORD

Datový typ:
Bajtové pole

Majetek společnosti:
ConnectionFactory

Heslo, které lze použít k ověření aplikace při pokusu o připojení k serveru systému zpráv. Heslo se používá s vlastností XMSC_USERID .

Standardně není vlastnost nastavena.

Multi Pokud se připojujete k produktu IBM MQ on Multiplatformsa nastavíte vlastnost XMSC_USERID továrny připojení, musí odpovídat hodnotě **userid** přihlášeného uživatele. Pokud tyto vlastnosti nenastavíte, správce front standardně použije **userid** přihlášeného uživatele. Pokud požadujete další ověření jednotlivých uživatelů na úrovni připojení, můžete napsat uživatelskou proceduru ověření klienta, která je nakonfigurována v produktu IBM MQ. Další informace o vytvoření uživatelské procedury ověření klienta naleznete v tématu Plánování ověřování pro klientskou aplikaci.

z/OS Chcete-li ověřit uživatele při připojování k produktu IBM MQ for z/OS , musíte použít uživatelskou proceduru pro zabezpečení zprávy.

XMSC_PRIORITY

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Název použitý v identifikátoru URI:
priorita

Priorita zpráv odeslaných do cíle.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
Celé číslo v rozsahu 0, nejnižší priorita, až 9, nejvyšší priorita	Zpráva odeslaná do místa určení má určenou prioritu. Výchozí priorita producenta zpráv nebo jakákoli priorita uvedená ve volání Send se ignoruje. Pokud je cílem fronta IBM MQ , hodnota atributu fronty **DefPriority** se také ignoruje.
XMSC_PRIORITY_AS_APP	Zpráva odeslaná do místa určení má prioritu uvedenou ve volání odeslání. Pokud volání Send neuvádí žádnou prioritu, použije se výchozí priorita producenta zpráv. Pokud je cílem fronta IBM MQ , hodnota atributu fronty **DefPriority** se ignoruje.
XMSC_PRIORITY_AS_DEST	Pokud je cílem fronta IBM MQ , má zpráva vložená do fronty prioritu určenou hodnotou atributu fronty **DefPriority**. Výchozí priorita producenta zpráv nebo jakákoli priorita uvedená ve volání Send se ignoruje. Pokud cíl není frontou IBM MQ , význam je stejný jako u parametru XMSC_PRIORITY_AS_APP.

Výchozí hodnota je XMSC_PRIORITY_AS_APP.

WebSphere MQ Real-Time Transport a WebSphere MQ Multicast Transport neprovádět žádnou akci na základě priority zprávy.

XMSC_PROVIDER_NAME

Datový typ:
Řetězec

Majetek společnosti:
ConnectionMetaData

Poskytovatel klienta XMS . Tato vlastnost je jen pro čtení.

XMSC_RTT_BROKER_PING_INTERVAL

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Časový interval v milisekundách, po který XMS .NET kontroluje připojení k serveru systému zpráv v reálném čase ke zjištění jakékoli aktivity. Pokud není zjištěna žádná aktivita, klient zahájí příkaz ping; připojení se zavře, pokud není zjištěna žádná odezva na příkaz ping.

Výchozí hodnota vlastnosti je 30000.

XMSC_RTT_CONNECTION_PROTOCOL

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Komunikační protokol použitý pro připojení v reálném čase ke zprostředkovateli.

Hodnota vlastnosti musí být XMSC_RTT_CP_TCP, což znamená připojení v reálném čase ke zprostředkovateli přes TCP/IP. Výchozí hodnota je XMSC_RTT_CP_TCP.

XMSC_RTT_HOST_NAME

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název nebo adresa IP hostitele systému, na kterém je spuštěn zprostředkovatel.

Tato vlastnost se používá s vlastností XMSC_RTT_PORT k identifikaci zprostředkovatele.

Standardně není vlastnost nastavena.

XMSC_RTT_LOCAL_ADDRESS

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název hostitele nebo adresa IP lokálního síťového rozhraní, které má být použito pro připojení v reálném čase ke zprostředkovateli.

Tato vlastnost je užitečná pouze v případě, že systém, na kterém je aplikace spuštěna, má dvě nebo více síťových rozhraní a musíte být schopni určit, které rozhraní musí být použito pro připojení v reálném čase. Pokud má systém pouze jedno síťové rozhraní, lze použít pouze toto rozhraní. Pokud má systém dvě nebo více síťových rozhraní a vlastnost není nastavena, rozhraní se vybere náhodně.

Standardně není vlastnost nastavena.

XMSC_RTT_MULTICAST

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory a cíl

Název použitý v identifikátoru URI:
mulicast-vysílání

Nastavení výběrového vysílání pro továrnu připojení nebo cíl. Tuto vlastnost může mít pouze místo určení, které je tématem.

Aplikace používá tuto vlastnost k povolení výběrového vysílání ve spojení s připojením v reálném čase ke zprostředkovateli a, je-li výběrové vysílání povoleno, k určení přesného způsobu, jakým je výběrové vysílání používáno k doručování zpráv ze zprostředkovatele spotřebiteli zpráv. Vlastnost nemá žádný vliv na to, jak producent zpráv odesílá zprávy zprostředkovateli.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_RTT_MULTICAST_DISABLED	Zprávy nejsou doručeny spotřebiteli zpráv pomocí WebSphere MQ Multicast Transport. Tato hodnota je výchozí hodnotou pro objekt ConnectionFactory .
XMSC_RTT_MULTICAST_ASCF	Zprávy jsou doručovány spotřebiteli zpráv v souladu s nastavením výběrového vysílání pro továrnu připojení přidruženou ke spotřebiteli zpráv. Nastavení výběrového vysílání pro továrnu připojení je zaznamenáno v době vytvoření připojení. Tato hodnota je platná pouze pro cílový objekt a je výchozí hodnotou pro cílový objekt.

Platná hodnota	Význam
XMSC_RTT_MULTICAST_ENABLED	Pokud je téma konfigurováno pro výběrové vysílání ve zprostředkovateli, jsou zprávy doručeny spotřebiteli zpráv pomocí WebSphere MQ Multicast Transport. Spolehlivá kvalita služby se používá, pokud je téma nakonfigurováno pro spolehlivé výběrové vysílání.
XMSC_RTT_MULTICAST_RELIABLE	Pokud je téma konfigurováno pro spolehlivé výběrové vysílání ve zprostředkovateli, zprávy jsou doručeny spotřebiteli zpráv pomocí produktu WebSphere MQ Multicast Transport se spolehlivou kvalitou služby. Pokud téma není konfigurováno pro spolehlivé výběrové vysílání, nemůžete pro toto téma vytvořit spotřebitele zpráv.
XMSC_RTT_MULTICAST_NOT_RELIABLE	Pokud je téma konfigurováno pro výběrové vysílání ve zprostředkovateli, jsou zprávy doručeny spotřebiteli zpráv pomocí WebSphere MQ Multicast Transport. Spolehlivá kvalita služby se nepoužívá, i když je téma konfigurováno pro spolehlivé výběrové vysílání.

XMSC_RTT_PORT

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Číslo portu, na kterém zprostředkovatel naslouchá příchozím požadavkům. Na zprostředkovateli musíte nakonfigurovat uzel zpracování zpráv Real-timeInput nebo Real-timeOptimizedFlow, aby naslouchal na tomto portu.

Tato vlastnost se používá s vlastností XMSC_RTT_HOST_NAME k identifikaci zprostředkovatele.

Výchozí hodnota vlastnosti je XMSC_RTT_DEFAULT_PORT nebo 1506.

XMSC_TIME_TO_LIVE

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Název použitý v identifikátoru URI:

expirace (pro cíl IBM MQ)
timeToLive (pro výchozí místo určení poskytovatele systému zpráv WebSphere)

Doba životnosti zpráv odeslaných do cíle.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
0	Platnost zprávy odeslané do místa určení nikdy nevyprší.

Platná hodnota	Význam
Kladné celé číslo	Zpráva odeslaná do místa určení má určenou dobu platnosti v milisekundách. Výchozí doba platnosti producenta zpráv nebo jakákoli doba platnosti uvedená ve volání Send je ignorována.
XMSC_TIME_TO_LIVE_AS_APP	Zpráva odeslaná do místa určení má dobu platnosti určenou v rámci volání odeslání. Pokud volání Send neurčuje dobu platnosti, použije se výchozí doba platnosti producenta zpráv.

Výchozí hodnota je XMSC_TIME_TO_LIVE_AS_APP.

XMSC_USERID

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Identifikátor uživatele, který lze použít k ověření aplikace při pokusu o připojení k serveru systému zpráv. Identifikátor uživatele se používá s vlastností XMSC_PASSWORD .

Standardně není vlastnost nastavena.

Multi Pokud se připojujete k produktu IBM MQ for Multiplatformsa nastavíte vlastnost XMSC_USERID továrny připojení, musí odpovídat **userid** přihlášeného uživatele. Pokud tyto vlastnosti nenastavíte, správce front standardně použije **userid** přihlášeného uživatele. Pokud požadujete další ověření jednotlivých uživatelů na úrovni připojení, můžete napsat uživatelskou proceduru ověření klienta, která je nakonfigurována v produktu IBM MQ. Další informace o vytvoření uživatelské procedury ověření klienta naleznete v tématu Plánování ověřování pro klientskou aplikaci.

z/OS Chcete-li ověřit uživatele při připojování k produktu IBM MQ for z/OS , musíte použít uživatelskou proceduru pro zabezpečení zprávy.

XMSC_VERSION

Datový typ:
Řetězec

Majetek společnosti:
ConnectionMetaData

Identifikátor verze rozhraní cliXMSent. Tato vlastnost je jen pro čtení.

V 9.3.0 *XMSC_WMQ_BALANCING_APPLICATION_TYPE*

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_BALANCING_APPLICATION_TYPE_SIMPLE	Jednoduché vyvážení; kromě pravidel popsaných v tématu Opětovné vyvážení aplikací v uniformních klastrechse nepoužívají žádná specifická pravidla. Toto je výchozí hodnota.

Platná hodnota	Význam
XMSC_WMQ_BALANCING_APPLICATION_TYPE_REQUEST_REPLY	Vyvažování požadavek-odezva; po každém volání MQPUT je pro zprávu odpovědi očekáváno odpovídající volání MQGET. Vyvažování je zpožděno, dokud není taková zpráva přijata, nebo dokud není překročena hodnota EXPIRACE zprávy požadavku.

Kromě toho lze tuto vlastnost nastavit v souboru `client.ini` . Pořadí předvoleb je:

1. Vlastnosti nastavené v aplikaci
2. Shoda s názvem Sekce aplikace v souboru `mqclient.ini` .
3. Sekce předvoleb aplikace v souboru `mqclient.ini` .

V 9.3.0 *XMSC_WMQ_BALANCING_OPTIONS*

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Odpovídající hodnota
XMSC_WMQ_BALANCING_OPTIONS_NONE	Nejsou nastaveny žádné volby. Toto je výchozí hodnota
XMSC_WMQ_BALANCING_OPTIONS_IGNORE_TRANSACTIONS	Nastavení této volby umožňuje vyvážit aplikace i v případě, že se nacházejí uprostřed transakce.

Kromě toho lze tuto vlastnost nastavit v souboru `client.ini` . Pořadí předvoleb je:

1. Vlastnosti nastavené v aplikaci
2. Shoda s názvem Sekce aplikace v souboru `mqclient.ini` .
3. Sekce předvoleb aplikace v souboru `mqclient.ini` .

V 9.3.0 *XMSC_WMQ_BALANCING_TIMEOUT*

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_BALANCING_TIMEOUT_IMMEDIATE	Dojde k okamžitému vypršení časového limitu
XMSC_WMQ_BALANCING_TIMEOUT_AS_DEFAULT	Nastavená výchozí hodnota časového limitu. Toto je výchozí hodnota
XMSC_WMQ_BALANCING_TIMEOUT_NEVER	Žádný časový limit se nevyskytne

Poznámka: Musíte zadat pouze jednu hodnotu z definovaných hodnot, nebo hodnotu 0-999999999 sekund.

Kromě toho lze tuto vlastnost nastavit v souboru `client.ini` . Pořadí předvoleb je:

1. Vlastnosti nastavené v aplikaci
2. Shoda s názvem Sekce aplikace v souboru `mqclient.ini` .
3. Sekce předvoleb aplikace v souboru `mqclient.ini` .

XMSC_WMQ_BROKER_CONTROLQ

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název řídicí fronty používané zprostředkovatelem.

Výchozí hodnota vlastnosti je SYSTEM.BROKER.CONTROL.QUEUE.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_BROKER_PUBQ

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název fronty monitorované zprostředkovatelem, kde aplikace odesílají zprávy, které publikují.

Výchozí hodnota vlastnosti je SYSTEM.BROKER.DEFAULT.STREAM.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_BROKER_QMGR

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název správce front, ke kterému je zprostředkovatel připojen.

Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_BROKER_SUBQ

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název fronty odběratele pro spotřebitele netrvalých zpráv.

Název fronty odběratele musí začínat následujícími znaky:

SYSTEM.JMS.ND.

Chcete-li, aby všichni přechodní spotřebitelé zpráv sdíleli frontu odběratele, zadejte úplný název sdílené fronty. Fronta se zadaným názvem musí existovat dříve, než může aplikace vytvořit spotřebitele přechodných zpráv.

Chcete-li, aby každý spotřebitel přechodných zpráv načítal zprávy z vlastní fronty výlučných odběratelů, zadejte název fronty, která končí hvězdičkou (*). Když pak aplikace vytvoří spotřebitele netrvalých zpráv, klient XMS vytvoří dynamickou frontu pro výhradní použití spotřebitelem zpráv. Klient XMS používá

hodnotu vlastnosti k nastavení obsahu pole **DynamicQName** v deskriptoru objektu, který se používá k vytvoření dynamické fronty.

Výchozí hodnota vlastnosti je SYSTEM.JMS.ND.SUBSCRIBER.QUEUE, což znamená, že produkt XMS standardně používá přístup ke sdílené frontě.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_BROKER_VERSION

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory a cíl

Název použitý v identifikátoru URI:
brokerVersion

Typ zprostředkovatele použitý aplikací pro připojení nebo pro cíl. Tuto vlastnost může mít pouze místo určení, které je tématem.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_BROKER_V1	Aplikace používá zprostředkovatele publikování/odběru IBM MQ . Aplikace může tuto hodnotu také použít, pokud provádíte migraci z IBM MQ publikování/odběr na WebSphere Message Broker , ale nezměnila aplikaci.
XMSC_WMQ_BROKER_V2	Aplikace používá zprostředkovatele IBM Integration Bus.
XMSC_WMQ_BROKER_UNURČENÝ	Po migraci zprostředkovatele nastavte tuto vlastnost tak, aby se záhlaví RFH2 již nepoužívala. Po migraci již tato vlastnost není relevantní.

Výchozí hodnota pro továrnu připojení je XMSC_WMQ_BROKER_UNURČENÝ, ale standardně není vlastnost pro místo určení nastavena. Nastavení vlastnosti pro místo určení potlačí jakoukoli hodnotu určenou vlastností továrny připojení.

XMSC_WMQ_CCDTURL

Datový typ:
System.String

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CCDTURL

Krátký název nástroje pro administraci JMS: CCDT

Adresa URL (Uniform Resource Locator) identifikující název a umístění souboru, který obsahuje tabulku definic kanálů klienta, a také určuje, jakým způsobem lze k souboru přistupovat.

Standardně není tato vlastnost nastavena.

XMSC_WMQ_CCSID

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Název použitý v identifikátoru URI:
CCSID

Identifikátor (CCSID) kódované znakové sady nebo kódové stránky, ve které jsou řetězce znakových dat v těle zprávy, když klient XMS předá zprávu do místa určení. Je-li nastaveno pro jednotlivou zprávu, vlastnost JMS_IBM_CHARACTER_SET přepíše CCSID zadané pro místo určení touto vlastností.

Výchozí hodnota vlastnosti je 1208.

Tato vlastnost je relevantní pouze pro zprávy odeslané do místa určení, nikoli pro zprávy přijaté z místa určení.

XMSC_WMQ_CHANNEL (kanál)

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CHANNEL

Krátký název nástroje pro administraci JMS: CHAN

Název kanálu, který se má použít pro připojení.

Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu klienta.

XMSC_WMQ_CLIENT_RECONNECT_OPTIONS

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CLIENTRECONNECTOPTIONS

Krátký název nástroje pro administraci JMS: CROPT

Tato vlastnost určuje volby opětovného připojení klienta pro nová připojení vytvořená touto továrnou. Nachází se v prostředí XMSC a je jedním z následujících:

- WMQ_CLIENT_RECONNECT_AS_DEF (výchozí). Použijte hodnotu uvedenou v souboru `mqclient.ini` . Nastavte hodnotu pomocí vlastnosti **DefRecon** v sekci Kanály. Lze ji nastavit na jednu z následujících možností:
  1. Ano. Chová se jako volba WMQ_CLIENT_RECONNECT
  2. -Ne. Výchozí. Neuvádí žádné volby opětovného připojení
  3. Správce front. Chová se jako volba WMQ_CLIENT_RECONNECT_Q_MGR
  4. Zakázáno. Chová se jako volba WMQ_CLIENT_RECONNECT_DISABLED.
- WMQ_CLIENT_RECONNECT. Znovu se připojte k libovolnému správci front zadanému v seznamu názvů připojení.
- WMQ_CLIENT_RECONNECT_Q_MGR. Znovu se připojí ke stejnému správci front, ke kterému je původně připojen. Vrátí hodnotu `MQRC_RECONNECT_QMID_MISMATCH` , pokud má správce front, ke kterému se pokouší připojit (určený v seznamu názvů připojení), jiný identifikátor QMID než původně připojený správce front.
- WMQ_CLIENT_RECONNECT_DISABLED. Opětovné připojení je zakázáno.

XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CLIENTRECONNECTTIMEOUT

Krátký název nástroje pro administraci JMS: CRT

Vlastnost XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT je platná pouze pro spravovaného klienta XMS .NET .

Tato vlastnost určuje dobu v sekundách, po kterou se připojení klienta pokusí znovu připojit.

Po pokusu o opětovné připojení po tuto dobu dojde k selhání klienta s chybou MQRC_RECONNECT_FAILED. Výchozí nastavení této vlastnosti je XMSC.WMQ_CLIENT_RECONNECT_TIMEOUT_DEFAULT.

Výchozí hodnota této vlastnosti je 1800.

XMSC_WMQ_CONNECTION_MODE

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Režim, pomocí kterého se aplikace připojuje ke správci front.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_CM_BINDINGS	Připojení ke správci front v režimu vazeb pro optimální výkon. Tato hodnota je výchozí hodnota pro C/C + +.
XMSC_WMQ_CM_CLIENT	Připojení ke správci front v režimu klienta pro zajištění plně spravovaného zásobníku. Tato hodnota je výchozí hodnota pro .NET.
XMSC_WMQ_CM_CLIENT_UNMANAGED (pouze pro .NET)	Připojení ke správci front, které vynucuje zásobník nespravovaného klienta.

XMSC_WMQ_CONNECTION_NAME_LIST

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CONNECTIONNAMELIST

Krátký název nástroje pro administraci JMS: CNLIST

Tato vlastnost určuje hostitele, ke kterým se klient pokusí znovu připojit po přerušení připojení.

Seznam názvů připojení je čárkami oddělený seznam dvojic hostitel/IP port. Výchozí nastavení této vlastnosti je WMQ_CONNECTION_NAME_LIST_DEFAULT.

Například`127.0.0.1 (1414),host2.example.com(1400)`

Výchozí nastavení této vlastnosti je `localhost (1414)`.

XMSC_WMQ_DUR_SUBQ

Datový typ:
Řetězec

Majetek společnosti:
Místo určení

Název fronty odběratele pro trvalého odběratele, který přijímá zprávy z cíle. Tuto vlastnost může mít pouze místo určení, které je tématem.

Název fronty odběratele musí začínat následujícími znaky:

SYSTEM.JMS.D.

Chcete-li, aby všichni trvalí odběratelé sdíleli frontu odběratele, zadejte úplný název sdílené fronty. Aby mohla aplikace vytvořit trvalého odběratele, musí existovat fronta se zadaným názvem.

Chcete-li, aby každý trvalý odběratel načítal zprávy ze své vlastní výlučné fronty odběratelů, zadejte název fronty, která končí hvězdičkou (*). Když pak aplikace vytvoří trvalého odběratele, klient XMS vytvoří dynamickou frontu pro výhradní použití trvalým odběratelem. Klient XMS používá hodnotu vlastnosti k nastavení obsahu pole **DynamicQName** v deskriptoru objektu, který se používá k vytvoření dynamické fronty.

Výchozí hodnota vlastnosti je SYSTEM.JMS.D.SUBSCRIBER.QUEUE, což znamená, že produkt XMS standardně používá přístup ke sdílené frontě.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_ENCODING

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Způsob reprezentace číselných dat v těle zprávy v případě, že klient produktu XMS předá zprávu do místa určení. Je-li nastaveno pro jednotlivou zprávu, vlastnost JMS_IBM_ENCODING potlačí kódování zadané pro místo určení touto vlastností. Tato vlastnost určuje reprezentaci binárních celých čísel, pakovaných desetinných celých čísel a čísel s pohyblivou řádovou čárkou.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **Encoding** deskriptoru zprávy.

Aplikace může k nastavení vlastnosti použít následující pojmenované konstanty:

pojmenovaná konstanta	Význam
MQENC_INTEGER_NORMAL	Normální kódování celých čísel
MQENC_INTEGER_REVERSED	Obrácené kódování celých čísel
MQENC_DECIMAL_NORMAL	Normální pakované kódování desetinných čísel
MQENC_DECIMAL_REVERSED	Obrácené pakované kódování desetinných čísel
MQENC_FLOAT_IEEE_NORMAL	Normální kódování čísel s pohyblivou řádovou čárkou IEEE
MQENC_FLOAT_IEEE_REVERSED	Obrácené kódování čísel s pohyblivou řádovou čárkou IEEE
MQENC_FLOAT_S390	z/OS kódování pohyblivé řádové čárky architektury
MQENC_NATIVE	Nativní kódování počítače

Chcete-li vytvořit hodnotu pro vlastnost, může aplikace přidat tři z těchto konstant takto:

- Konstanta, jejíž název začíná hodnotou MQENC_INTEGER, která určuje reprezentaci binárních celých čísel.
- Konstanta, jejíž název začíná na MQENC_DECIMAL, pro určení reprezentace pakovaných desetinných celých čísel.
- Konstanta, jejíž název začíná na MQENC_FLOAT, pro určení reprezentace čísel s pohyblivou řádovou čárkou.

Alternativně může aplikace nastavit vlastnost na MQENC_NATIVE, jejíž hodnota je závislá na prostředí.

Výchozí hodnota vlastnosti je MQENC_NATIVE.

Tato vlastnost je relevantní pouze pro zprávy odeslané do místa určení, nikoli pro zprávy přijaté z místa určení.

XMSC_WMQ_FAIL_IF_QUIESCE

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory a cíl

Název použitý v identifikátoru URI:
FAILIFQUIESCE

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: FAILIFQUIESCE

Krátký název nástroje pro administraci JMS: FIQ

Zda se volání konkrétních metod nezdaří, pokud je správce front, k němuž je aplikace připojen, ve stavu uvedení do klidového stavu.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_FIQ_YES	Volání určitých metod se nezdaří, pokud je správce front ve stavu uvedení do klidového stavu. Když aplikace zjistí, že je správce front uveden do klidového stavu, může dokončit svou okamžitou úlohu a zavřít připojení, což správci front umožní jeho zastavení.
XMSC_WMQ_FIQ_NO	Neselhala žádná volání metody, protože správce front je ve stavu uvedení do klidového stavu. Zadáte-li tuto hodnotu, aplikace nemůže zjistit, zda je správce front uveden do klidového stavu. Aplikace může pokračovat v provádění operací se správcem front, a zabránit tak zastavení správce front.

Výchozí hodnota pro továrnu připojení je XMSC_WMQ_FIQ_YES, ale standardně není vlastnost nastavena pro místo určení. Nastavení vlastnosti pro místo určení potlačí jakoukoli hodnotu určenou vlastností továrny připojení.

XMSC_WMQ_MESSAGE_BODY

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Tato vlastnost určuje, zda aplikace XMS zpracovává MQRFH2 zprávy IBM MQ jako součást informačního obsahu zprávy (tj. jako součást těla zprávy).

Poznámka: Při odesílání zpráv do místa určení vlastnost XMSC_WMQ_MESSAGE_BODY nahrazuje existující vlastnost cíle XMS XMSC_WMQ_TARGET_CLIENT.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_MESSAGE_BODY_JMS
Přijmout: Typ a tělo příchozí zprávy XMS jsou určeny obsahem MQRFH2 (je-li přítomen) nebo MQMD (není-li v přijaté IBM MQ zprávě uveden žádný MQRFH2).

Odeslat: Tělo odchozí zprávy XMS obsahuje předpřipravené a automaticky generované záhlaví MQRFH2 na základě vlastností a polí záhlaví zprávy XMS .

XMSC_WMQ_MESSAGE_BODY_MQ
Přijmout: Typ příchozí XMS zprávy je vždy ByteMessage, bez ohledu na obsah přijaté IBM MQ zprávy nebo pole formátu přijatého MQMD. Tělo zprávy XMS je nezměněná data zprávy vrácená základním voláním rozhraní API poskytovatele systému zpráv. Znaková sada a kódování dat v těle zprávy je určeno pomocí polí CodedCharSetId a Encoding v deskriptoru MQMD. Formát dat v těle zprávy je určen polem Formát deskriptoru MQMD.

Odeslat: Tělo odchozí zprávy XMS obsahuje informační obsah aplikace tak, jak je; a do těla se nepřidá žádné automaticky generované záhlaví IBM MQ .

XMSC_WMQ_MESSAGE_BODY_UNURČENÝ
Přijmout: Klient XMS určí vhodnou hodnotu pro tuto vlastnost. V cestě příjmu je tato hodnota hodnotou vlastnosti WMQ_MESSAGE_BODY_JMS.

Odeslat: Klient XMS určí vhodnou hodnotu pro tuto vlastnost. V cestě odeslání je tato hodnota hodnotou vlastnosti XMSC_WMQ_TARGET_CLIENT.

Standardně je tato vlastnost nastavena na hodnotu XMSC_WMQ_MESSAGE_BODY_UNURČENÝ.

XMSC_WMQ_MQMD_MESSAGE_CONTEXT

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Určuje, jaká úroveň kontextu zprávy má být nastavena aplikací XMS . Aby tato vlastnost mohla nabýt účinnosti, musí aplikace běžet s příslušným oprávněním kontextu.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_MDCTX_DEFAULT
Pro odchozí zprávy volání rozhraní API MQOPEN a struktura MQPMO neuvádí žádné explicitní volby kontextu zprávy.

XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT
Volání rozhraní API MQOPEN určuje volbu kontextu zprávy MQOO_SET_IDENTITY_CONTEXT a struktura MQPMO určuje volbu MQPMO_SET_IDENTITY_CONTEXT.

XMSC_WMQ_MDCTX_SET_ALL_CONTEXT
Volání MQOPEN API určuje volbu kontextu zprávy MQOO_SET_ALL_CONTEXT a struktura MQPMO určuje volbu MQPMO_SET_ALL_CONTEXT.

Standardně je tato vlastnost nastavena na hodnotu XMSC_WMQ_MDCTX_DEFAULT.

Poznámka: Tato vlastnost není relevantní, když se aplikace připojí k produktu WebSphere Application Server service integration bus.

Následující vlastnosti vyžadují, aby byla vlastnost XMSC_WMQ_MQMD_MESSAGE_CONTEXT nastavena na hodnotu vlastnosti XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT nebo hodnotu vlastnosti XMSC_WMQ_MDCTX_SET_ALL_CONTEXT při odesílání zprávy, aby byl efekt žádoucí:

- JMS_IBM_MQMD_USERIDENTIFIER
- JMS_IBM_MQMD_ACCOUNTINGTOKEN
- JMS_IBM_MQMD_APPLIDENTITYDATA

Následující vlastnosti vyžadují, aby vlastnost XMSC_WMQ_MQMD_MESSAGE_CONTEXT byla při odesílání zprávy nastavena na hodnotu vlastnosti XMSC_WMQ_MDCTX_SET_ALL_CONTEXT, aby měla požadovaný efekt:

- JMS_IBM_MQMD_PUTAPPLTYPE
- JMS_IBM_MQMD_PUTAPPLNAME
- JMS_IBM_MQMD_PUTDATE
- JMS_IBM_MQMD_PUTTIME
- JMS_IBM_MQMD_APPLORIGINDATA

XMSC_WMQ_MQMD_READ_ENABLED

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Tato vlastnost určuje, zda může aplikace XMS extrahovat hodnoty polí MQMD.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_READ_ENABLED_NO
Při odesílání zpráv nejsou vlastnosti JMS_IBM_MQMD* pro odeslanou zprávu aktualizovány tak, aby odrážely aktualizované hodnoty polí v deskriptoru MQMD.

Při příjmu zpráv nejsou v přijaté zprávě k dispozici žádné vlastnosti JMS_IBM_MQMD*, a to ani v případě, že některé z nich nebo všechny jsou nastaveny odesilatelem.

XMSC_WMQ_READ_ENABLED_YES
Při odesílání zpráv jsou všechny vlastnosti JMS_IBM_MQMD* pro odeslanou zprávu aktualizovány tak, aby odrážely aktualizované hodnoty polí v deskriptoru MQMD, včetně vlastností, které odesilatel explicitně nenastavil.

Při příjmu zpráv jsou pro přijatou zprávu k dispozici všechny vlastnosti JMS_IBM_MQMD*, včetně vlastností, které odesilatel explicitně nenastavil.

Standardně je tato vlastnost nastavena na hodnotu XMSC_WMQ_READ_ENABLED_NO.

XMSC_WMQ_MQMD_WRITE_ENABLED

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Tato vlastnost určuje, zda může aplikace XMS nastavit hodnoty polí MQMD.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_WRITE_ENABLED_NO
Všechny vlastnosti JMS_IBM_MQMD* jsou ignorovány a jejich hodnoty nejsou zkopírovány do základní struktury MQMD.

XMSC_WMQ_WRITE_ENABLED_YES
Vlastnosti JMS_IBM_MQMD* jsou zpracovány. Jejich hodnoty jsou zkopírovány do podkladové struktury MQMD.

Standardně je tato vlastnost nastavena na hodnotu XMSC_WMQ_WRITE_ENABLED_NO.

XMSC_WMQ_PUT_ASYNC_ALLOWED

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Tato vlastnost určuje, zda producenti zpráv mohou používat k odesílání zpráv do tohoto místa určení asynchronní operace put.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_DEST
Zjistěte, zda jsou povolena asynchronní vložení, odkazem na definici fronty nebo tématu.

XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_Q_DEF
Určete, zda jsou povolena asynchronní vložení, odkazem na definici fronty.

XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_TOPIC_DEF
Zjistěte, zda jsou povolena asynchronní vložení, na základě definice tématu.

XMSC_WMQ _PUT_ASYNC_ALLOWED_DISABLED
Asynchronní vložení nejsou povolena.

XMSC_WMQ _PUT_ASYNC_ALLOWED_ENABLED
Asynchronní vložení jsou povolena.

Standardně je tato vlastnost nastavena na hodnotu XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_DEST.

Poznámka: Tato vlastnost není relevantní, když se aplikace připojuje k produktu WebSphere Application Server service integration bus.

XMSC_WMQ_READ_AHEAD_ALLOWED

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Tato vlastnost určuje, zda mohou příjemci zpráv a zprostředkovatelé front používat dopředné čtení k načtení netrvalých netransakčních zpráv z tohoto cíle do interní vyrovnávací paměti před jejich přijetím.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_READ_AHEAD_ALLOWED_AS_Q_DEF
Určete, zda je dopředné čtení povoleno odkazem na definici fronty.

XMSC_WMQ_READ_AHEAD_ALLOWED_AS_ TOPIC _DEF
Určete, zda je dopředné čtení povoleno odkazem na definici tématu.

XMSC_WMQ_READ_AHEAD_ALLOWED_AS_DEST
Určete, zda je dopředné čtení povoleno odkazem na definici fronty nebo tématu.

XMSC_WMQ_READ_AHEAD_ALLOWED_DISABLED
Dopředné čtení není povoleno při příjmu nebo procházení zpráv.

XMSC_WMQ_READ_AHEAD_ALLOWED_ENABLED
Dopředné čtení je povoleno.

Standardně je tato vlastnost nastavena na hodnotu XMSC_WMQ _READ_AHEAD_ALLOWED_AS_DEST.

XMSC_WMQ_READ_AHEAD_CLOSE_POLICY

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Tato vlastnost určuje, zda budou zprávy doručovány do asynchronního modulu listener zpráv, co se stane se zprávami v interní vyrovnávací paměti pro čtení při zavření spotřebitele zpráv.

Tuto vlastnost lze použít při zadávání voleb zavření fronty při příjmu zpráv z místa určení a nelze ji použít při odesílání zpráv do místa určení.

Tato vlastnost je pro prohlížeče front ignorována, protože během procházení jsou zprávy stále k dispozici ve frontách.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT
Před návratem se dokončí pouze aktuální vyvolání modulu listener pro zprávy, což potenciálně zanechá zprávy ve vnitřní vyrovnávací paměti dopředného čtení, které se pak vyřadí.

XMSC_WMQ_READ_AHEAD_CLOSE_POLICY_DELIVER_ALL
Všechny zprávy v interní vyrovnávací paměti dopředného čtení jsou doručeny modulu listener pro zprávy aplikace před návratem.

Standardně je tato vlastnost nastavena na hodnotu XMSC_WMQ _READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT.

Poznámka:

Nestandardní ukončení aplikace
Všechny zprávy ve vyrovnávací paměti dopředného čtení jsou při náhlém ukončení aplikace XMS ztraceny.

Důsledky pro transakce
Dopředné čtení je zakázáno, když aplikace používají transakce. Takže aplikace nevidí žádný rozdíl v chování, když používají transakční relace.

Důsledky režimů seance
Dopředné čtení je povoleno v netransakční relaci, když jsou režimy potvrzení buď XMSC_AUTO_KVITOVAT, nebo XMSC_DUPS_OK_KVITOVAT. Dopředné čtení je zakázáno, pokud je režim potvrzení relace nastaven na XMSC_CLIENT_KVITOVAT bez ohledu na transakční nebo netransakční relace.

Důsledky pro prohlížeče front a selektory prohlížečů front
Prohlížeče front a selektory prohlížečů front, používané v aplikacích XMS , získají výhodu výkonu z dopředného čtení. Zavření prohlížeče front nesníží výkon, protože zpráva je stále k dispozici ve frontě pro další operace. Pro prohlížeče front a selektory prohlížečů front nejsou žádné jiné důsledky kromě výkonnostních výhod čtení napřed.

XMSC_WMQ_HOST_NAME

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: HOSTNAME

Krátký název nástroje pro administraci JMS: HOST

Název nebo adresa IP hostitele systému, na kterém je spuštěn správce front.

Tato vlastnost se používá pouze v případě, že se aplikace připojuje ke správci front v režimu klienta. Tato vlastnost se používá s vlastností XMSC_WMQ_PORT k identifikaci správce front.

Výchozí hodnota vlastnosti je `localhost`.

XMSC_WMQ_LOCAL_ADDRESS

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: LOCALADDRESS

Krátký název nástroje pro administraci JMS: LA

Pro připojení ke správci front tato vlastnost určuje rozhraní lokální sítě, lokálního portu nebo rozsahu lokálních portů, nebo obojí.

Hodnota vlastnosti je řetězec s následujícím formátem:

[*název_hostitele*] [(*low_port*) [,*high_port*])]

Význam proměnných je následující:

název_hostitele
Název hostitele nebo adresa IP lokálního síťového rozhraní, které se má použít pro připojení.

Poskytnutí těchto informací je nezbytné pouze v případě, že systém, na kterém je aplikace spuštěna, má dvě nebo více síťových rozhraní a musíte být schopni určit, které rozhraní musí být pro připojení použito. Pokud má systém pouze jedno síťové rozhraní, lze použít pouze toto rozhraní. Pokud má systém dvě nebo více síťových rozhraní a neuvedete, které rozhraní se musí použít, rozhraní se vybere náhodně.

low_port
Číslo lokálního portu, který se má použít pro připojení.

Je-li zadána také hodnota *high_port* , je hodnota *low_port* interpretována jako nejnižší číslo portu v rozsahu čísel portů.

vysoký_port
Nejvyšší číslo portu v rozsahu čísel portů. Pro připojení musí být použit jeden z portů v uvedeném rozsahu.

Maximální délka řetězce je 48 znaků.

Zde je několik příkladů platných hodnot vlastnosti:

JUPITER
9.20.4.98
JUPITER (1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)

Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu klienta.

XMSC_WMQ_MESSAGE_SELECTION

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Určuje, zda výběr zpráv provádí klient XMS nebo zprostředkovatel.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_MSEL_CLIENT	Výběr zpráv provádí klient XMS .
XMSC_WMQ_MSEL_BROKER	Výběr zpráv provádí zprostředkovatel.

Výchozí hodnota je XMSC_WMQ_MSEL_CLIENT.

Tato vlastnost je relevantní pouze v doméně publikování/odběru. Výběr zpráv zprostředkovatelem není podporován, pokud je vlastnost XMSC_WMQ_BROKER_VERSION nastavena na hodnotu XMSC_WMQ_BROKER_V1.

XMSC_WMQ_MSG_BATCH_SIZE

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Maximální počet zpráv, které mají být načteny z fronty v jedné dávce, v případě asynchronního doručování zpráv.

Když aplikace používá asynchronní doručování zpráv, klient XMS za určitých podmínek načte dávku zpráv z fronty před individuálním postoupením každé zprávy aplikaci. Tato vlastnost určuje maximální počet zpráv, které mohou být v dávce.

Hodnota vlastnosti je kladné celé číslo a výchozí hodnota je 10. Zvažte nastavení vlastnosti na jinou hodnotu pouze v případě, že máte specifický problém s výkonem, který musíte řešit.

Je-li aplikace připojena ke správci front prostřednictvím sítě, zvýšení hodnoty této vlastnosti může snížit režijní náklady sítě a dobu odezvy, ale zvýšit množství paměti potřebné k uložení zpráv v klientském systému. Naopak snížení hodnoty této vlastnosti může zvýšit režijní náklady sítě a dobu odezvy, ale snížit množství paměti potřebné k uložení zpráv.

XMSC_WMQ_POLLING_INTERVAL

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Pokud každý modul listener pro zprávy v rámci relace nemá ve frontě žádnou odpovídající zprávu, jedná se o maximální interval v milisekundách, který uplyne předtím, než se každý modul listener pro zprávy znovu pokusí získat zprávu z příslušné fronty.

Pokud se často stává, že pro žádný z modulů listener pro zprávy v relaci není k dispozici žádná vhodná zpráva, zvažte zvýšení hodnoty této vlastnosti.

Hodnota vlastnosti je kladné celé číslo. Výchozí hodnota je 5000.

XMSC_WMQ_PORT

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: PORT

Krátký název nástroje pro administraci JMS: PORT

Číslo portu, na kterém správce front naslouchá příchozím požadavkům.

Tato vlastnost se používá pouze v případě, že se aplikace připojuje ke správci front v režimu klienta. Tato vlastnost se používá spolu s vlastností XMSC_WMQ_HOST_NAME k identifikaci správce front.

Výchozí hodnota vlastnosti je XMSC_WMQ_DEFAULT_CLIENT_PORT nebo 1414.

XMSC_WMQ_PROVIDER_VERSION

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Verze, vydání, úroveň modifikace a opravný balík správce front, ke kterému se tato aplikace hodlá připojovat. Platné hodnoty pro tuto vlastnost jsou:

- Nespecifikováno

Nebo řetězec v jednom z následujících formátů

- V.R.M.F
- V.R.M
- V.R
- V

kde V, R, M a F jsou celá čísla větší nebo rovná nule.

Standardně je tato vlastnost nastavena na hodnotu "nespecifikováno".

Verze klienta IBM MQ také hraje hlavní roli v tom, zda může klientská aplikace XMS používat IBM MQ specifické funkce.

Poznámka: Systémová vlastnost XMSC_WMQ_OVERRIDEPROVIDERVERSION potlačuje vlastnost XMSC_WMQ_PROVIDER_VERSION. Tuto vlastnost lze použít, pokud nemůžete změnit nastavení továrny připojení.

XMSC_WMQ_PUB_ACK_INTERVAL

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Počet zpráv publikovaných vydavatelem, než klient XMS vyžádá potvrzení od zprostředkovatele.

Pokud hodnotu této vlastnosti snížíte, klient požaduje potvrzení častěji, a proto výkon vydavatele klesá. Zvýšíte-li tuto hodnotu, bude klientovi trvat déle, než ohlásí výjimku při selhání zprostředkovatele.

Hodnota vlastnosti je kladné celé číslo. Výchozí hodnota je 25.

XMSC_WMQ_QMGR_CCSID

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Identifikátor (CCSID) kódované znakové sady nebo kódové stránky, ve které jsou pole znakových dat definovaná v rozhraní MQI (Message Queue Interface) vyměňována mezi klientem XMS a klientem IBM MQ . Tato vlastnost se nevztahuje na řetězce znakových dat v tělech zpráv.

Když se aplikace XMS připojuje ke správci front v režimu klienta, klient XMS odkazuje na klienta IBM MQ . Informace vyměňované mezi dvěma klienty obsahují pole znakových dat, která jsou definována v rozhraní MQI. Za normálních okolností klient IBM MQ předpokládá, že tato pole jsou na kódové stránce systému,

na kterém jsou klienti spuštěni. Pokud klient XMS poskytuje a očekává přijetí těchto polí na jiné kódové stránce, musíte tuto vlastnost nastavit tak, aby informovala klienta IBM MQ .

Když klient IBM MQ předává tato pole znakových dat správci front, musí být data v nich obsažená v případě potřeby převedena na kódovou stránku používanou správcem front. Podobně, když klient IBM MQ obdrží tato pole od správce front, data v nich obsažená musí být v případě potřeby převedena na kódovou stránku, na které klient XMS očekává přijetí dat. Klient IBM MQ používá tuto vlastnost k provedení těchto převodů dat.

Standardně není vlastnost nastavena.

Nastavení této vlastnosti je ekvivalentní nastavení proměnné prostředí MQCCSID pro klienta IBM MQ , který podporuje nativní aplikace klienta IBM MQ . Další informace o této proměnné prostředí viz MQCCSID.

XMSC_WMQ_QUEUE_MANAGER

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: QMANAGER

Krátký název nástroje pro administraci JMS: QMGR

Název správce front, s nímž má být navázáno připojení.

Standardně není vlastnost nastavena.

XMSC_WMQ_RECEIVE_CCSID

Cílová vlastnost, která nastavuje cílový CCSID pro převod zpráv správce front. Hodnota je ignorována, pokud není parametr XMSC_WMQ_RECEIVE_CONVERSION nastaven na hodnotu WMQ_RECEIVE_CONVERSION_QMGR.

Datový typ:
Celé číslo

Hodnota:
Jakékoli kladné celé číslo.

Výchozí hodnota je 1208.

Zadání hodnoty GMO_CONVERT ve zprávě je volitelné. Je-li zadána hodnota GMO_CONVERT , dojde k převodu podle zadané hodnoty.

XMSC_WMQ_RECEIVE_CONVERSION

Cílová vlastnost, která určuje, zda má správce front provést převod dat.

Datový typ:
Celé číslo

Hodnoty:
XMSC_WMQ_RECEIVE_CONVERSION_CLIENT_MSG (VÝCHOZÍ): Proveďte převod dat pouze na klientovi XMS . Konverze se provádí vždy pomocí kódové stránky 1208.

XMSC_WMQ_RECEIVE_CONVERSION_QMGR: Před odesláním zprávy klientovi XMS proveďte převod dat ve správci front.

XMSC_WMQ_RECEIVE_EXIT

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Identifikuje uživatelskou proceduru pro přijetí zprávy kanálu, která má být spuštěna.

Hodnota vlastnosti je řetězec, který identifikuje uživatelskou proceduru pro příjem kanálu a má následující formát:

libraryName(entryPointNázev)

kde:

- **libraryName** je úplná cesta ke spravované uživatelské proceduře `.dll`
- `entryPointNázev` je název třídy kvalifikovaný oborem názvů.

Například: `C:\MyReceiveExit.dll(MyReceiveExitNameSpace.MyReceiveExitClassName)`

Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu spravovaného klienta. Podporovány jsou také pouze spravované uživatelské procedury.

XMSC_WMQ_RECEIVE_EXIT_INIT

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Uživatelská data, která jsou předána uživatelské proceduře pro přijetí zprávy kanálu při volání.

Hodnota vlastnosti je řetězec. Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu spravovaného klienta a je nastavena vlastnost "XMSC_WMQ_RECEIVE_EXIT" na stránce 2071 .

XMSC_WMQ_RESOLVED_QUEUE_MANAGER

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Tato vlastnost se používá k získání názvu správce front, ke kterému je připojena.

Při použití s tabulkou definic kanálů klienta (CCDT) se může tento název lišit od názvu správce front určeného v továrně připojení.

XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Tato vlastnost je po připojení naplněna ID správce front.

XMSC_WMQ_SECURITY_EXIT

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Identifikuje uživatelskou proceduru pro zabezpečení zprávy kanálu.

Hodnota vlastnosti je řetězec, který identifikuje uživatelskou proceduru pro zabezpečení zprávy kanálu a má následující formát:

libraryName(entryPointNázev)

kde:

- **libraryName** je úplná cesta ke spravovanému souboru .dll uživatelské procedury.
- entryPointNázev je název třídy kvalifikovaný oborem názvů.

Například C:\MySecurityExit.dll(MySecurityExitNameSpace.MySecurityExitClassName)

Maximální délka řetězce je 128 znaků.

Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu spravovaného klienta. Podporovány jsou také pouze spravované uživatelské procedury.

XMSC_WMQ_SECURITY_EXIT_INIT

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Uživatelská data, která jsou předána uživatelské proceduře pro zabezpečení zprávy kanálu při volání.

Maximální délka řetězce uživatelských dat je 32 znaků.

Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu spravovaného klienta a je nastavena vlastnost “XMSC_WMQ_SECURITY_EXIT” na stránce 2072 .

XMSC_WMQ_SEND_EXIT

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Identifikuje uživatelskou proceduru pro odeslání zprávy kanálu.

Hodnota vlastnosti je řetězec. Uživatelská procedura odeslání kanálu má následující formát:

libraryName(entryPointNázev)

kde:

- **libraryName** je úplná cesta ke spravovanému souboru .dll uživatelské procedury.
- entryPointNázev je název třídy kvalifikovaný oborem názvů.

Například: C:\MySendExit.dll(MySendExitNameSpace.MySendExitClassName)

Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu spravovaného klienta. Podporovány jsou také pouze spravované uživatelské procedury.

XMSC_WMQ_SEND_EXIT_INIT

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Uživatelská data, která jsou předána uživatelským procedurám pro odeslání zprávy kanálu při volání.

Hodnota vlastnosti je řetězec jedné nebo více položek uživatelských dat oddělených čárkami. Standardně není vlastnost nastavena.

Pravidla pro určení uživatelských dat, která jsou předána do posloupnosti uživatelských procedur pro odeslání kanálu, jsou stejná jako pravidla pro určení uživatelských dat, která jsou předána do posloupnosti uživatelských procedur pro příjem kanálu. Pravidla proto viz "XMSC_WMQ_RECEIVE_EXIT_INIT" na stránce 2072.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu spravovaného klienta a je nastavena vlastnost "XMSC_WMQ_SEND_EXIT" na stránce 2073 .

XMSC_WMQ_SEND_CHECK_COUNT

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Počet povolených odeslaných volání mezi kontrolou asynchronních chyb vložení během jedné netransakční relace XMS.

Standardně je tato vlastnost nastavena na hodnotu 0.

XMSC_WMQ_SHARE_CONV_ALLOWED

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: SHARECONVALLOWED

Krátký název nástroje pro administraci JMS: SCALD

Zda může připojení klienta sdílet svůj soket s jinými připojeními nejvyšší úrovně XMS ze stejného procesu ke stejnému správci front, pokud se shodují definice kanálů. Tato vlastnost slouží k povolení úplné izolace produktu Connections v samostatných soketech, pokud je to vyžadováno pro vývoj aplikací, údržbu nebo provozní důvody. Nastavení této vlastnosti pouze označuje XMS , aby byl základní soket sdílený. Neoznačuje, kolik připojení sdílí jeden soket. Počet připojení sdílejících soket je určen hodnotou SHARECNV, která je vyjednána mezi klientem IBM MQ a serverem IBM MQ .

Aplikace může nastavit následující pojmenované konstanty pro nastavení vlastnosti:

- XMSC_WMQ_SHARE_CONV_ALLOWED_FALSE-Připojení nesdílejí soket.
- XMSC_WMQ_SHARE_CONV_ALLOWED_TRUE-Připojení sdílí soket.

Standardně je vlastnost nastavena na hodnotu XMSC_WMQ_SHARE_CONV_ALLOWED_ENABLED.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu klienta.

XMSC_WMQ_SSL_CERT_STORES

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Umístění serverů obsahující seznamy odvolaných certifikátů (CRL), které mají být použity v připojení SSL ke správci front.

Hodnota vlastnosti je seznam jedné nebo více adres URL oddělených čárkami. Každá URL má následující formát:

```
[user[/password]@]ldap://[serveraddress][:portnum][,...]
```

Tento formát je kompatibilní se základním formátem MQJMS, ale je od něj rozšířen.

Je platné mít prázdný `serveraddress`. V tomto případě produkt XMS předpokládá, že hodnota je řetězec "localhost".

Příklad seznamu je:

```
myuser/mypassword@ldap://server1.mycom.com:389
ldap://server1.mycom.com
ldap://
ldap://:389
```

Pouze pro .NET : Ze serveru IBM MQ 8.0spravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT_UNMANAGED) podporují připojení TLS/SSL.

Standardně není vlastnost nastavena.

Související pojmy

Podpora SSL a TLS pro nespravovaného klienta .NET

Podpora SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_CIPHER_SPEC

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název specifikace CipherSpec, která má být použita v zabezpečeném připojení ke správci front.

Specifikace šifer, které můžete použít s podporou TLS v produktu IBM MQ , jsou uvedeny v následující tabulce. Požadujete-li osobní certifikát, určíte velikost klíče pro dvojici veřejný a soukromý klíč. Velikost klíče, která se používá během navázání komunikace SSL, je velikost uložená v certifikátu, pokud není určena CipherSpec, jak je uvedeno v tabulce. Standardně není tato vlastnost nastavena.

Název specifikace šifrování	Použitý protokol	Hašovací algoritmus	Šifrovací algoritmus	Šifrování bitů	FIPS[1]	Sada B 128 bitů	Sada B 192 bitů
TLS_RSA_WITH_AES_128_CBC_SHA	TLS 1.0	SHA-1	AES	128	Ano	Ne	Ne
TLS_RSA_WITH_AES_256_CBC_SHA[2]	TLS 1.0	SHA-1	AES	256	Ano	Ne	Ne
TLS_RSA_WITH_DES_CBC_SHA	TLS 1.0	SHA-1	DES	56	Ne	Ne	Ne
TLS_RSA_WITH_3DES_EDE_CBC_SHA[4]	TLS 1.0	SHA-1	3DES	168	Ano	Ne	Ne
TLS_RSA_WITH_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Ano	Ne	Ne
TLS_RSA_WITH_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Ano	Ne	Ne
TLS_RSA_WITH_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Ano	Ne	Ne

Název specifikace šifrování	Použitý protokol	Hašovací algoritmus	Šifrovací algoritmus	Šifrování bitů	FIPS[1]	Sada B 128 bitů	Sada B 192 bitů
TLS_RSA_WITH_AES_256_CBC_SHA256	TLS 1.2	SHA-256	AES	256	Ano	Ne	Ne
ECDHE_ECDSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Ne	Ne	Ne
ECDHE_ECDSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Ano	Ne	Ne
ECDHE_RSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Ne	Ne	Ne
ECDHE_RSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	Ano	Ne	Ne
ECDHE_ECDSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Ano	Ne	Ne
ECDHE_ECDSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	Ano	Ne	Ne
ECDHE_RSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	Ano	Ne	Ne
ECDHE_RSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	Ano	Ne	Ne
ECDHE_ECDSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Ano	Ano	Ne
ECDHE_ECDSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Ano	Ne	Ano
ECDHE_RSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	Ano	Ne	Ne
ECDHE_RSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	Ano	Ne	Ne
TLS_RSA_WITH_NULL_SHA256	TLS 1.2	SHA-256	Není	0	Ne	Ne	Ne
ECDHE_RSA_NULL_SHA256	TLS 1.2	SHA-256	Není	0	Ne	Ne	Ne
ECDHE_ECDSA_NULL_SHA256	TLS 1.2	SHA-256	Není	0	Ne	Ne	Ne
TLS_RSA_WITH_NULL_NULL	TLS 1.2	Není	Není	0	Ne	Ne	Ne
TLS_RSA_WITH_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	Ne	Ne	Ne

Název specifikace šifrování	Použitý protokol	Hašovací algoritmus	Šifrovací algoritmus	Šifrování bitů	FIPS[1]	Sada B 128 bitů	Sada B 192 bitů

Notes:

1. Uvádí, zda CipherSpec vyhovuje standardu FIPS (Federal Information Processing Standards) 140-2. Vysvětlení standardu FIPS a informace o tom, jak nakonfigurovat produkt IBM MQ pro operaci vyhovující standardu FIPS 140-2, naleznete v tématu FIPS (Federal Information Processing Standards).
2. Tuto CipherSpec nelze použít k zabezpečení připojení z produktu IBM MQ Explorer ke správci front, pokud nejsou příslušné soubory zásad bez omezení použity na prostředí JRE používané produktem IBM MQ Explorer.
3. Tato specifikace šifrování byla certifikována FIPS 140-2 před 19. květnem 2007.
4. Je-li produkt IBM MQ nakonfigurovaný pro operace odpovídající standardu FIPS 140-2, lze tuto specifikaci šifrování použít k přenosu až 32 GB dat, než bude připojení ukončeno chybou AMQ9288. Chcete-li se vyhnout této chybě, buď se vyvarujte použití trojitého DES (který je zamítnutý), nebo povolte reset tajného klíče při použití této CipherSpec v konfiguraci FIPS 140-2.

Související pojmy

Integrita dat zpráv

Související úlohy

Zabezpečení

Určení CipherSpecs

XMSC_WMQ_SSL_CIPHER_SUITE

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název sady CipherSuite, která má být použita v rámci připojení TLS ke správci front. Protokol použitý při vyjednávání zabezpečeného připojení závisí na určené sadě CipherSuite.

Tato vlastnost má následující kanonické hodnoty:

- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA
- SSL_RSA_EXPORT1024_WITH_RC4_56_SHA
- SSL_RSA_EXPORT_WITH_RC4_40_MD5
- SSL_RSA_WITH_RC4_128_MD5
- SSL_RSA_WITH_RC4_128_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_WITH_AES_128_CBC_SHA
- SSL_RSA_WITH_AES_256_CBC_SHA
- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA

Tuto hodnotu lze zadat jako alternativu k parametru XMSC_WMQ_SSL_CIPHER_SPEC.

Je-li pro parametr XMSC_WMQ_SSL_CIPHER_SPECzadána neprázdná hodnota, přepíše tato hodnota nastavení pro parametr XMSC_WMQ_SSL_CIPHER_SUITE. Pokud XMSC_WMQ_SSL_CIPHER_SPEC nemá hodnotu, použije se hodnota XMSC_WMQ_SSL_CIPHER_SUITE jako šifrovací sada, která má být předána IBM Global Security Kit (GSKit). V tomto případě je hodnota mapována na ekvivalentní hodnotu

CipherSpec , jak je popsáno v části CipherSuite a CipherSpec pro připojení produktu XMS ke správci front IBM MQ.

Pokud jsou oba parametry XMSC_WMQ_SSL_CIPHER_SPEC a XMSC_WMQ_SSL_CIPHER_SUITE prázdné, pole pChDef->SSLCipherSpec je vyplněno mezerami.

Pouze pro .NET : Ze serveru IBM MQ 8.0spravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT_UNMANAGED) podporují připojení TLS/SSL.

Standardně není vlastnost nastavena.

Související pojmy

Podpora SSL a TLS pro nespravovaného klienta .NET

Podpora SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_CRYPTO_HW

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Podrobnosti konfigurace šifrovacího hardwaru připojeného k systému klienta.

Tato vlastnost má následující kanonické hodnoty:

- GSK_ACCELERATOR_RAINBOW_CS_OFF
- GSK_ACCELERATOR_RAINBOW_CS_ON
- GSK_ACCELERATOR_NCIPHER_NF_OFF
- GSK_ACCELERATOR_NCIPHER_NF_ON

Pro šifrovací hardware PKCS11 existuje speciální formát (kde DriverPath, TokenLabela TokenPassword jsou řetězce určené uživatelem):

```
GSK_PKCS11=PKCS#11 DriverPath; PKCS#11 TokenLabel;PKCS#11 TokenPassword
```

XMS neinterpretuje ani nemění obsah řetězce. Zkopíruje zadanou hodnotu až do limitu 256 jednobajtových znaků do produktu MQSCO.CryptoHardware hardwaru šifrování.

Pouze pro .NET : Ze serveru IBM MQ 8.0spravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT_UNMANAGED) podporují připojení TLS/SSL.

Standardně není vlastnost nastavena.

Související pojmy

Podpora SSL a TLS pro nespravovaného klienta .NET

Podpora SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_FIPS_REQUIRED

Datový typ:
Logická hodnota

Majetek společnosti:
ConnectionFactory

Hodnota této vlastnosti určuje, zda aplikace může nebo nemůže používat šifrovací sady, které nevyhovují standardu FIPS. Je-li tato vlastnost nastavena na hodnotu true, pro připojení klientského serveru se používají pouze algoritmy FIPS.

Tato vlastnost může mít následující hodnoty, které se překládají na dvě kanonické hodnoty pro MQSCO.FipsRequired:

Tabulka 880. Tabulka hodnot pro produkt MQSCO.FlipsRequired

Hodnota	**Popis**	**Odpovídající hodnota MQSCO.FipsRequired**
ne	Lze použít libovolnou specifikaci CipherSpec .	MQSSL_FIPS_NO (výchozí)
ano	V CipherSpec , které se používají pro toto připojení klienta, lze použít pouze šifrovací algoritmy s certifikací FIPS.	MQSSL_FIPS_YES

Produkt XMS zkopíruje příslušnou hodnotu do produktu MQSCO.FipsRequired před voláním MQCONNX.

Pouze pro .NET : Ze serveru IBM MQ 8.0spravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT_UNMANAGED) podporují připojení TLS/SSL.

Související pojmy

Podpora SSL a TLS pro nespravovaného klienta .NET

Podpora SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_KEY_REPOSITORY

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty.

Produkt XMS zkopíruje řetězec až do limitu 256 jednobajtových znaků do produktu MQSCO.KeyRepository . Produkt IBM MQ interpretuje tento řetězec jako název souboru včetně úplné cesty.

Pouze pro .NET : Ze serveru IBM MQ 8.0spravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT_UNMANAGED) podporují připojení TLS/SSL.

Standardně není vlastnost nastavena.

Související pojmy

Podpora SSL a TLS pro nespravovaného klienta .NET

Podpora SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_KEY_RESETCOUNT

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Hodnota KeyResetCount představuje celkový počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace SSL, než je znovu vyjednán tajný klíč. Počet bajtů zahrnuje řídicí informace odeslané MCA.

XMS zkopíruje hodnotu, kterou jste pro tuto vlastnost dodali, do produktu MQSCO.KeyResetCount před voláním MQCONNX.

Parametr MQSCO.KeyRestCount je k dispozici pouze od IBM MQ verze 6. Je-li IBM MQ verze 5.3, pokud je tato vlastnost nastavena, produkt XMS se nepokusí o připojení ke správci front a místo toho vygeneruje příslušnou výjimku.

Pouze pro .NET : Ze serveru IBM MQ 8.0spravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT_UNMANAGED) podporují připojení TLS/SSL.

Výchozí hodnota této vlastnosti je nula, což znamená, že tajné klíče nejsou nikdy znovu vyjednány.

Související pojmy

Podpora SSL a TLS pro nespravovaného klienta .NET

Podpora SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_PEER_NAME

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název typu peer, které se použije při připojení SSL ke správci front.

Pro tuto vlastnost není k dispozici žádný seznam kanonických hodnot. Místo toho musíte sestavit tento řetězec podle pravidel pro SSLPEER.

Příkladem názvu typu peer je:

```
"CN=John Smith, O=IBM ,OU=Test , C=GB"
```

Produkt XMS zkopíruje řetězec do správné jednobajtové kódové stránky a umístí správné hodnoty do polí MQCD.SSLPeerNamePtr a MQCD.SSLPeerNameLength před voláním MQCONNX.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front v režimu klienta.

Pouze pro .NET : Ze serveru IBM MQ 8.0spravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k serveru IBM MQ (WMQ_CM_CLIENT_UNMANAGED) podporují připojení TLS/SSL.

Standardně není vlastnost nastavena.

Související pojmy

Podpora SSL a TLS pro nespravovaného klienta .NET

Podpora SSL a TLS pro spravovaného klienta .NET

Související odkazy

SSLPEERNAME

XMSC_WMQ_SYNCPOINT_ALL_GETS

Datový typ:
System.Boolean

Majetek společnosti:
ConnectionFactory

Určuje, zda musí být všechny zprávy načteny z front v rámci řízení synchronizačního bodu.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
ne	Jsou-li okolnosti vhodné, může klient produktu XMS načítat zprávy z front mimo řízení synchronizačního bodu.

Platná hodnota	Význam
ano	Klient XMS musí načíst všechny zprávy z front v rámci řízení synchronizačního bodu.

Výchozí hodnota je false.

XMSC_WMQ_TARGET_CLIENT

Datový typ:
System.Int32

Majetek společnosti:
Místo určení

Název použitý v identifikátoru URI:
targetClient

Určuje, zda zprávy odeslané do cíle obsahují záhlaví MQRFH2.

Pokud aplikace odešle zprávu obsahující záhlaví MQRFH2 , přijímající aplikace musí být schopna záhlaví zpracovat.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_TARGET_DEST_JMS	Zprávy odeslané do místa určení obsahují záhlaví MQRFH2 . Tuto hodnotu zadejte, pokud aplikace odesílá zprávy jiné aplikaci XMS , aplikaci IBM MQ classes for JMS nebo nativní aplikaci IBM MQ , která je navržena pro zpracování záhlaví MQRFH2 .
XMSC_WMQ_TARGET_DEST_MQ	Zprávy odeslané do místa určení neobsahují záhlaví MQRFH2 . Tuto hodnotu zadejte, pokud aplikace odesílá zprávy do nativní aplikace IBM MQ , která není navržena pro zpracování záhlaví MQRFH2 .

Výchozí hodnota je XMSC_WMQ_TARGET_DEST_JMS.

XMSC_WMQ_TEMP_Q_PREFIX

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Předpona použitá k vytvoření názvu dynamické fronty IBM MQ , která se vytvoří, když aplikace vytvoří XMS dočasnou frontu.

Pravidla pro vytváření předpony jsou stejná jako pravidla pro vytváření obsahu pole **DynamicQName** v deskriptoru objektu, ale poslední neprázdný znak musí být hvězdička (*). Není-li vlastnost nastavena, použije se hodnota `CSQ.*` na z/OS a `AMQ.*` na ostatních platformách. Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v doméně typu point-to-point.

XMSC_WMQ_TEMP_TOPIC_PREFIX

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory, cíl

Při vytváření dočasných témat produkt XMS vygeneruje řetězec tématu ve tvaru "`TEMP/TEMPTOPICPREFIX/unique_id`", nebo pokud tato vlastnost obsahuje výchozí hodnotu, vygeneruje

se tento řetězec "TEMP/unique_id". Určení neprázdné hodnoty umožní definování specifických modelových front za účelem vytvoření spravovaných front pro odběratele dočasných témat vytvořených v rámci tohoto připojení.

Jakýkoli nenulový řetězec, který se skládá pouze z platných znaků pro řetězec tématu IBM MQ , je platnou hodnotou pro tuto vlastnost.

Standardně je tato vlastnost nastavena na "" (prázdný řetězec).

Poznámka: Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_TEMPORARY_MODEL

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název modelové fronty IBM MQ , ze které se vytvoří dynamická fronta, když aplikace vytvoří XMS dočasnou frontu.

Výchozí hodnota vlastnosti je SYSTEM.DEFAULT.MODEL.QUEUE.

Tato vlastnost je relevantní pouze v doméně typu point-to-point.

XMSC_WMQ_WILDCARD_FORMAT

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory, cíl

Tato vlastnost určuje, která verze syntaxe zástupných znaků se má použít.

Při použití publikování/odběru s IBM MQ '*' a '?' jsou považovány za zástupné znaky. Zatímco '#' a '+' jsou považovány za zástupné znaky při použití publikování s odběrem IBM Integration Bus. Tato vlastnost nahrazuje vlastnost XMSC_WMQ_BROKER_VERSION.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_WILDCARD_TOPIC_ONLY
Rozpozná pouze zástupné znaky na úrovni tématu, tj. '#' a '+' se považují za zástupné znaky. Tato hodnota je stejná jako hodnota XMSC_WMQ_BROKER_V2.

XMSC_WMQ_WILDCARD_CHAR_ONLY
Rozpozná pouze zástupné znaky znaků, tj. "*" a "?" jsou považovány za zástupné znaky. Tato hodnota je stejná jako hodnota XMSC_WMQ_BROKER_V1.

Standardně je tato vlastnost nastavena na hodnotu XMSC_WMQ_WILDCARD_TOPIC_ONLY.

XMSC_WPM_BUS_NAME

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory a cíl

Název použitý v identifikátoru URI:
busName

U továrny připojení se jedná o název sběrnice SIBus, ke které se aplikace připojuje, nebo pro cíl, název sběrnice SIBus, ve které cíl existuje.

Pro místo určení, které je tématem, je tato vlastnost názvem sběrnice SIBus, ve které existuje přidružený prostor tématu. Tento prostor tématu je určen vlastností XMSC_WPM_TOPIC_SPACE .

Není-li vlastnost pro místo určení nastavena, předpokládá se, že fronta nebo přidružený prostor tématu existují ve sběrnici pro integraci služeb, ke které se aplikace připojuje.

Standardně není vlastnost nastavena.

XMSC_WPM_CONNECTION_PROTOCOL

Datový typ:
System.Int32

Majetek společnosti:
Připojení

Komunikační protokol použitý pro připojení k jádru systému zpráv. Tato vlastnost je jen pro čtení.

Možné hodnoty vlastnosti jsou následující:

Hodnota	Význam
XMSC_WPM_CP_HTTP	Připojení používá protokol HTTP přes TCP/IP.
XMSC_WPM_CP_TCP	Připojení používá protokol TCP/IP.

XMSC_WPM_CONNECTION_PROXIMITY

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Nastavení blízkosti připojení pro připojení. Tato vlastnost určuje, jak blízko musí být stroj systému zpráv, ke kterému se aplikace připojuje, k serveru samozavedení.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Nastavení blízkosti připojení
XMSC_WPM_CONNECTION_PROXIMITY_BUS	Sběrnice
XMSC_WPM_CONNECTION_PROXIMITY_CLUSTER	Klastr
XMSC_WPM_CONNECTION_PROXIMITY_HOST	Hostitel
XMSC_WPM_CONNECTION_PROXIMITY_SERVER	Server

Výchozí hodnota je XMSC_WPM_CONNECTION_PROXIMITY_BUS.

XMSC_WPM_DUR_SUB_HOME

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název použitý v identifikátoru URI:
durableSubscriptionDomů

Název jádra systému zpráv, v nichž jsou spravovány všechny trvalé odběry pro připojení nebo cíl. Zprávy, které mají být doručeny trvalým odběratelům, jsou uloženy v bodě publikování téhož stroje systému zpráv.

Před vytvořením trvalého odběratele, který používá připojení, musí být pro připojení zadán domovský adresář trvalého odběru. Libovolná hodnota zadaná pro místo určení přepíše hodnotu zadanou pro připojení.

Standardně není vlastnost nastavena.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WPM_HOST_NAME

Datový typ:
Řetězec

Majetek společnosti:
Připojení

Název hostitele nebo adresa IP systému, který obsahuje jádro systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.

XMSC_WPM_LOCAL_ADDRESS

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Pro připojení ke sběrnici SIBus tato vlastnost určuje rozhraní lokální sítě, lokálního portu nebo rozsahu lokálních portů, nebo obojí.

Hodnota vlastnosti je řetězec s následujícím formátem:

[*název_hostitele*] [(*low_port*) [,*high_port*])]

Význam proměnných je následující:

název_hostitele
Název hostitele nebo adresa IP lokálního síťového rozhraní, které se má použít pro připojení.

Poskytnutí těchto informací je nezbytné pouze v případě, že systém, na kterém je aplikace spuštěna, má dvě nebo více síťových rozhraní a musíte být schopni určit, které rozhraní musí být pro připojení použito. Pokud má systém pouze jedno síťové rozhraní, lze použít pouze toto rozhraní. Pokud má systém dvě nebo více síťových rozhraní a neuvedete, které rozhraní se musí použít, rozhraní se vybere náhodně.

low_port
Číslo lokálního portu, který se má použít pro připojení.

Je-li zadána také hodnota *high_port* , je hodnota *low_port* interpretována jako nejnižší číslo portu v rozsahu čísel portů.

vysoký_port
Nejvyšší číslo portu v rozsahu čísel portů. Pro připojení musí být použit jeden z portů v uvedeném rozsahu.

Zde je několik příkladů platných hodnot vlastnosti:

JUPITER
9.20.4.98
JUPITER (1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)

Standardně není vlastnost nastavena.

XMSC_WPM_ME_NAME

Datový typ:
Řetězec

Majetek společnosti:
Připojení

Název jádra systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.

XMSC_WPM_NON_PERSISTENT_MAP

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Úroveň spolehlivosti přechodných zpráv, které se odesílají pomocí připojení.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Úroveň spolehlivosti
XMSC_WPM_MAPPING_AS_DESTINATION	Určuje se podle výchozí úrovně spolehlivosti určené pro frontu nebo prostor tématu ve sběrnici SIBus.
XMSC_WPM_MAPPING_BEST_PRACT_NON_Trvalý	Dočasné nejlepší úsilí
XMSC_WPM_MAPPING_EXPRESS_NON_Trvalý	Expresní přechodné
XMSC_WPM_MAPPING_RELIABLE_NON_Trvalý	Spolehlivý, dočasný
XMSC_WPM_MAPPING_RELIABLE_PERSISTENT	Spolehlivá trvalá
XMSC_WPM_MAPPING_ASSURED_PERSISTENT	Zajištěné trvalé

Výchozí hodnota je XMSC_WPM_MAPPING_EXPRESS_NON_PERSISTENT.

XMSC_WPM_PERSISTENT_MAP

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Úroveň spolehlivosti trvalých zpráv, které se odesílají pomocí připojení.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Úroveň spolehlivosti
XMSC_WPM_MAPPING_AS_DESTINATION	Určuje se podle výchozí úrovně spolehlivosti určené pro frontu nebo prostor tématu ve sběrnici SIBus.
XMSC_WPM_MAPPING_BEST_PRACT_NON_Trvalý	Dočasné nejlepší úsilí
XMSC_WPM_MAPPING_EXPRESS_NON_Trvalý	Expresní přechodné

Platná hodnota	Úroveň spolehlivosti
XMSC_WPM_MAPPING_RELIABLE_NON_ Trvalý	Spolehlivý, dočasný
XMSC_WPM_MAPPING_RELIABLE_PERSISTENT	Spolehlivá trvalá
XMSC_WPM_MAPPING_ASSURED_PERSISTENT	Zajištěné trvalé

Výchozí hodnota je XMSC_WPM_MAPPING_RELIABLE_PERSISTENT.

XMSC_WPM_PORT

Datový typ:
System.Int32

Majetek společnosti:
Připojení

Číslo portu naslouchaného v jádru systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.

XMSC_WPM_PROVIDER_KONCOVÉ body

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Sekvence jednoho nebo více adres koncového bodu zaváděcích serverů. Adresy koncových bodů jsou odděleny čárkami.

Server samozavedení je aplikační server, který je zodpovědný za výběr stroje systému zpráv, ke kterému se aplikace připojuje. Adresa koncového bodu serveru samozavedení má následující formát:

název_hostitele:*číslo_portu*:*název_řetězu*

Význam komponent adresy koncového bodu je následující:

název_hostitele
Název hostitele nebo adresa IP systému, na kterém je umístěn server samozavedení. Pokud není uveden žádný název hostitele nebo adresa IP, předvolba je `localhost`.

číslo_portu
Číslo portu, na kterém server samozavedení naslouchá příchozím požadavkům. Není-li uvedeno žádné číslo portu, předvolba je 7276.

název_řetězu
Název transportního řetězu samozavedení používaného serverem samozavedení. Platné hodnoty jsou:

Platná hodnota	Název transportního řetězu samozavedení
XMSC_WPM_BOOTSTRAP_HTTP	BootstrapTunneledSystém zpráv
XMSC_WPM_BOOTSTRAP_HTTPS	BootstrapTunneledSecureMessaging
XMSC_WPM_BOOTSTRAP_SSL	BootstrapSecureSystém zpráv
XMSC_WPM_BOOTSTRAP_TCP	BootstrapBasicSystém zpráv

Není-li uveden žádný název, výchozí hodnota je XMSC_WPM_BOOTSTRAP_TCP.

Není-li zadána žádná adresa koncového bodu, výchozí hodnota je localhost:7276:BootstrapBasicMessaging.

XMSC_WPM_SSL_CIPHER_SUITE

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název sady CipherSuite , která má být použita pro připojení TLS ke stroji systému zpráv WebSphere Application Server service integration bus . Protokol použitý při vyjednávání zabezpečeného připojení závisí na určené sadě CipherSuite.

Tabulka 881. Volby CipherSuite pro připojení ke stroji systému zpráv WebSphere Application Server service integration bus

šifrovací sada	**Použitý protokol**
TLS_RSA_WITH_DES_CBC_SHA	TLSv1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	TLSv1
TLS_RSA_WITH_AES_128_CBC_SHA	TLSv1
TLS_RSA_WITH_AES_256_CBC_SHA	TLSv1

Notes:

1. Windows TLS_RSA_WITH_AES_128_CBC_SHA a TLS_RSA_WITH_AES_256_CBC_SHA CipherSuites jsou podporovány pouze na systémech Windows . (Toto je diktováno GSKit.)
2. Deprecated TLS_RSA_WITH_3DES_EDE_CBC_SHA je zamítnutý. Přesto jej lze použít k přenosu až 32 GB dat před ukončením připojení s chybou AMQ9288. Chcete-li se vyhnout této chybě, musíte se vyhnout použití trojitého DES nebo povolit reset tajného klíče při použití této CipherSpec.

Pro tuto vlastnost neexistuje výchozí nastavení. Chcete-li použít zabezpečení SSL nebo TLS, musíte zadat hodnotu této vlastnosti, jinak se vaše aplikace nebude moci úspěšně připojit k serveru.

XMSC_WPM_SSL_FIPS_REQUIRED

Poznámka: V systému AIX, Linux, and Windowsposkytuje produkt IBM MQ kompatibilitu se standardem FIPS 140-2 prostřednictvím šifrovacího modulu IBM Crypto for C (ICC) . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C (ICC) certifikát a měli by si být vědomi všech doporučení poskytnutých NIST. Náhradní modul FIPS 140-3 momentálně probíhá a jeho stav lze zobrazit jeho vyhledáním v modulech NIST CMVP v seznamu procesů.

Datový typ:
Logická hodnota

Majetek společnosti:
ConnectionFactory

Hodnota této vlastnosti určuje, zda aplikace může nebo nemůže používat šifrovací sady, které nevyhovují standardu FIPS. Je-li tato vlastnost nastavena na hodnotu true, budou pro připojení typu klient-server použity pouze algoritmy FIPS.Nastavení hodnoty této vlastnosti na hodnotu TRUE zabrání aplikaci v použití šifrovacích sad, které nejsou kompatibilní se standardem FIPS.

Standardně je vlastnost nastavena na hodnotu FALSE (to znamená, že režim FIPS je vypnutý).

XMSC_WPM_SSL_KEY_REPOSITORY

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Cesta k souboru, který je souborem svazku klíčů obsahujícím veřejné nebo soukromé klíče, které mají být použity v zabezpečeném připojení.

Nastavení vlastnosti souboru svazku klíčů na speciální hodnotu parametru XMSC_WPM_SSL_MS_CERTIFICATE_STORE určuje použití databáze klíčů Microsoft Windows . Použití databáze klíčů Microsoft Windows , která se nachází v části **Ovládací panely** > **Možnosti Internetu** > **Obsah** > **Certifikáty**, odstraňuje potřebu samostatné databáze souborů s klíči. Použití této konstanty v systému Windows x64 a na jiných platformách není povoleno.

Standardně není vlastnost nastavena.

XMSC_WPM_SSL_KEYRING_LABEL

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Certifikát, který má být použit při ověřování na serveru. Není-li uvedena žádná hodnota, použije se výchozí certifikát.

Standardně není vlastnost nastavena.

XMSC_WPM_SSL_KEYRING_PW

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Heslo pro soubor svazku klíčů.

Tuto vlastnost lze použít jako alternativu ke konfiguraci hesla pro soubor svazku klíčů pomocí parametru XMSC_WPM_SSL_KEYRING_STASH_FILE .

Standardně není vlastnost nastavena.

XMSC_WPM_SSL_KEYRING_STASH_FILE

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název binárního souboru obsahujícího heslo k souboru úložiště klíčů.

Tuto vlastnost lze použít jako alternativu ke konfiguraci hesla pro soubor svazku klíčů pomocí parametru XMSC_WPM_SSL_KEYRING_PW .

Standardně není vlastnost nastavena.

XMSC_WPM_TARGET_GROUP

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název cílové skupiny jader systému zpráv. Povahu cílové skupiny určuje vlastnost XMSC_WPM_TARGET_TYPE .

Tuto vlastnost nastavte, chcete-li omezit hledání stroje systému zpráv na podskupinu strojů systému zpráv ve sběrnici SIBus. Chcete-li, aby se vaše aplikace mohla připojit k libovolnému stroji systému zpráv ve sběrnici pro integraci služeb, nenastavujte tuto vlastnost.

Standardně není vlastnost nastavena.

XMSC_WPM_TARGET_VÝZNAM

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Významnost cílové skupiny jader systému zpráv.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WPM_CÍLOVÝ_VÝZNAM_ Preferovaný	Stroj systému zpráv v cílové skupině je vybrán, pokud je k dispozici. Jinak je vybrán stroj systému zpráv mimo cílovou skupinu za předpokladu, že je ve stejné sběrnici pro integraci služeb.
XMSC_WPM_CÍLOVÝ_VÝZNAM_ POVINNÉ	Vybraný stroj systému zpráv musí být v cílové skupině. Není-li stroj systému zpráv v cílové skupině k dispozici, proces připojení se nezdaří.

Výchozí hodnota vlastnosti je XMSC_WPM_TARGET_VÝZNAM_PREFERRED.

XMSC_WPM_TARGET_TRANSPORT_CHAIN

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Název transportu příchozích požadavků, který musí aplikace používat pro připojení k jádru systému zpráv.

Hodnotou vlastnosti může být název libovolného příchozího transportního řetězu, který je k dispozici na aplikačním serveru, který je hostitelem stroje systému zpráv. Pro jeden z předdefinovaných příchozích transportních řetězů je poskytnuta následující pojmenovaná konstanta:

pojmenovaná konstanta	Název dopravního řetězce
XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC	InboundBasicSystém zpráv

Výchozí hodnota vlastnosti je XMSC_WPM_TARGET_TRANSPORT_CHAIN_BASIC.

XMSC_WPM_TARGET_TYPE

Datový typ:
System.Int32

Majetek společnosti:
ConnectionFactory

Typ cílové skupiny jader systému zpráv. Tato vlastnost určuje povahu cílové skupiny identifikované vlastností XMSC_WPM_TARGET_GROUP .

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WPM_TARGET_TYPE_BUSMEMBER	Název cílové skupiny je název člena sběrnice. Cílová skupina je všechny stroje systému zpráv ve členu sběrnice.

Platná hodnota	Význam
XMSC_WPM_TARGET_TYPE_CUSTOM	Název cílové skupiny je název skupiny strojů systému zpráv definované uživatelem. Cílová skupina je všechny stroje systému zpráv, které jsou registrovány s uživatelem definovanou skupinou.
XMSC_WPM_TARGET_TYPE_ME	Název cílové skupiny je název stroje systému zpráv. Cílová skupina je určený stroj systému zpráv.

Standardně není vlastnost nastavena.

XMSC_WPM_TEMP_Q_PREFIX

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Předpona použitá k vytvoření názvu dočasné fronty, která se vytvoří ve sběrnici pro integraci služeb, když aplikace vytvoří XMS dočasnou frontu. Předpona může obsahovat až 12 znaků.

Název dočasné fronty začíná znaky "_Q" následovanými předponou. Zbytek názvu se skládá ze znaků generovaných systémem.

Standardně není vlastnost nastavena, což znamená, že název dočasné fronty nemá předponu.

Tato vlastnost je relevantní pouze v doméně typu point-to-point.

XMSC_WPM_TEMP_TOPIC_PREFIX

Datový typ:
Řetězec

Majetek společnosti:
ConnectionFactory

Předpona používaná k vytvoření názvu dočasného tématu vytvořeného aplikací. Předpona může obsahovat až 12 znaků.

Název dočasného tématu začíná znaky "_T" následovanými předponou. Zbytek názvu se skládá ze znaků generovaných systémem.

Standardně není vlastnost nastavena, což znamená, že název dočasného tématu nemá předponu.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WPM_TOPIC_SPACE

Datový typ:
Řetězec

Majetek společnosti:
Místo určení

Název použitý v identifikátoru URI:
topicSpace

Název prostoru témat, který obsahuje dané téma. Tuto vlastnost může mít pouze místo určení, které je tématem.

Standardně není vlastnost nastavena, což znamená, že se předpokládá výchozí prostor tématu.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

Managed File Transfer reference vývojových aplikací

Referenční informace, které vám pomohou s vývojem aplikací pro produkt Managed File Transfer.

Příklady použití příkazu fteCreateTransfer ke spuštění programů

Pomocí příkazu **fteCreateTransfer** můžete určit programy, které se mají spustit před nebo po přenosu.

Kromě použití **fteCreateTransfer**existují i jiné způsoby, jak vyvolat program před nebo po přenosu. Další informace naleznete v tématu Určení programů, které mají být spuštěny s produktem MFT.

Všechny tyto příklady používají k určení programu následující syntaxi:

```
[type:]commandspec[,[retrycount][,[retrywait][,successrc]]]
```

Další informace o této syntaxi viz **fteCreateTransfer**: start a new file transfer.

Spuštění spustitelného programu

Následující příklad uvádí spustitelný program s názvem `mycommand` a předává programu dva argumenty, a a b.

```
mycommand(a,b)
```

Chcete-li spustit tento program na zdrojovém agentovi AGENT1 před spuštěním přenosu, použijte tento příkaz:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -presrc mycommand(a,b)
destinationSpecification sourceSpecification
```

Spuštění a opakování spustitelného programu

Následující příklad uvádí spustitelný program s názvem `simple`, který nepřebírá žádné argumenty. Pro parametr `retrycount` je zadána hodnota 1 a pro parametr `retrywait`je zadána hodnota 5. Tyto hodnoty znamenají, že program bude zopakován jednou, pokud nevrátí úspěšný návratový kód, po pěti sekundách čekání. Pro parametr `successrc`není zadána žádná hodnota, takže jediným úspěšným návratovým kódem je výchozí hodnota 0.

```
executable:simple,1,5
```

Chcete-li spustit tento program na zdrojovém agentovi AGENT1 po dokončení přenosu, použijte následující příkaz:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -postsrc executable:simple,1,5
destinationSpecification sourceSpecification
```

Spuštění skriptu Ant a určení úspěšných návratových kódů

Následující příklad uvádí skript Ant s názvem `myscript` a předává skriptu dvě vlastnosti. Skript se spouští pomocí příkazu **fteAnt** . Hodnota `successrc` je uvedena jako >2&<7&!5|0|14, která uvádí, že návratové kódy 0, 3, 4, 6 a 14 označují úspěch.

```
antscript:myscript(prop1=fred,prop2=bob),,,>2&<7&!5|0|14
```

Chcete-li spustit tento program na cílovém agentovi AGENT2 před spuštěním přenosu, použijte následující příkaz:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -predst
"antscript:myscript(prop1=fred,prop2=bob),,,>2&<7&!5|0|14"destinationSpecification sourceSpecification
```

Spuštění skriptu Ant a určení cílů, které se mají volat

Následující příklad uvádí skript Ant nazvaný `script2` a dva cíle, `target1` a `target2`, které se mají volat. Vlastnost `prop1` je také předána s hodnotou `recmfm(F,B)`. Závorky, čárky (,) a zpětná lomítka (\) jsou speciální znaky v příkazech MFT a musí být uvozeny zpětným lomítkem (\). Windows Cesty k souborům v systému Windows lze zadat buď pomocí dvojitých zpětných lomítek (\\) jako oddělovače, nebo pomocí jednoduchých dopředných lomítek (/). V následujícím příkladu jsou čárka (,) a závorky uvozeny znakem zpětného lomítka (\).

```
antscript:script2(target1,target2,prop1=recmfm\(F\,B\)),,,>2&<7&!5|0|14
```

Chcete-li spustit tento program na cílovém agentovi AGENT2 po dokončení přenosu, použijte následující příkaz:

```
fteCreateTransfer -sa AGENT1 -da AGENT2
-postdst "antscript:script2(target1,target2,prop1=recmfm\(F\,B\)),,,>2&<7&!5|0|14"
destinationSpecification sourceSpecification
```

Použití metadat ve skriptu Ant

Úlohu Ant můžete určit jako libovolné z následujících volání přenosu:

- Na zdroji před zpracováním
- Na zdroji po zpracování
- předurčení
- místo určení

Při spuštění úlohy Ant jsou metadata uživatele přenosu zpřístupněna pomocí proměnných prostředí. K těmto datům můžete přistupovat například pomocí následujícího kódu:

```
<property environment="environment" />
<echo>${environment.mymetadata}</echo>
```

kde `mymetadata` je název některých metadat vložených do přenosu.

Spuštění skriptu JCL

Následující příklad určuje skript JCL s názvem ZOSBATCH. Pro parametr `retrycount`je zadána hodnota 3, pro parametr `retrywait` je zadána hodnota 30 a pro parametr `successrc`je zadána hodnota 0. Tyto hodnoty znamenají, že se skript zopakuje třikrát, pokud nevrátí úspěšný návratový kód 0, s čekáním třicet sekund mezi každým pokusem.

```
jcl:ZOSBATCH,3,30,0
```

kde ZOSBATCH je členem PDS s názvem MYSYS.JCLa soubor `agent.properties` obsahuje řádek `commandPath=....://'MYSYS.JCL':...`

Chcete-li spustit tento program na zdrojovém agentovi AGENT1 po dokončení přenosu, použijte následující příkaz:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -postsrc jcl:ZOSBATCH,3,30,0
destinationSpecification sourceSpecification
```

Související úlohy

Určení programů, které se mají spustit s MFT

Související odkazy

`fteCreateTransfer`: spuštění nového přenosu souborů

fteAnt: spuštění Ant úloh v MFT

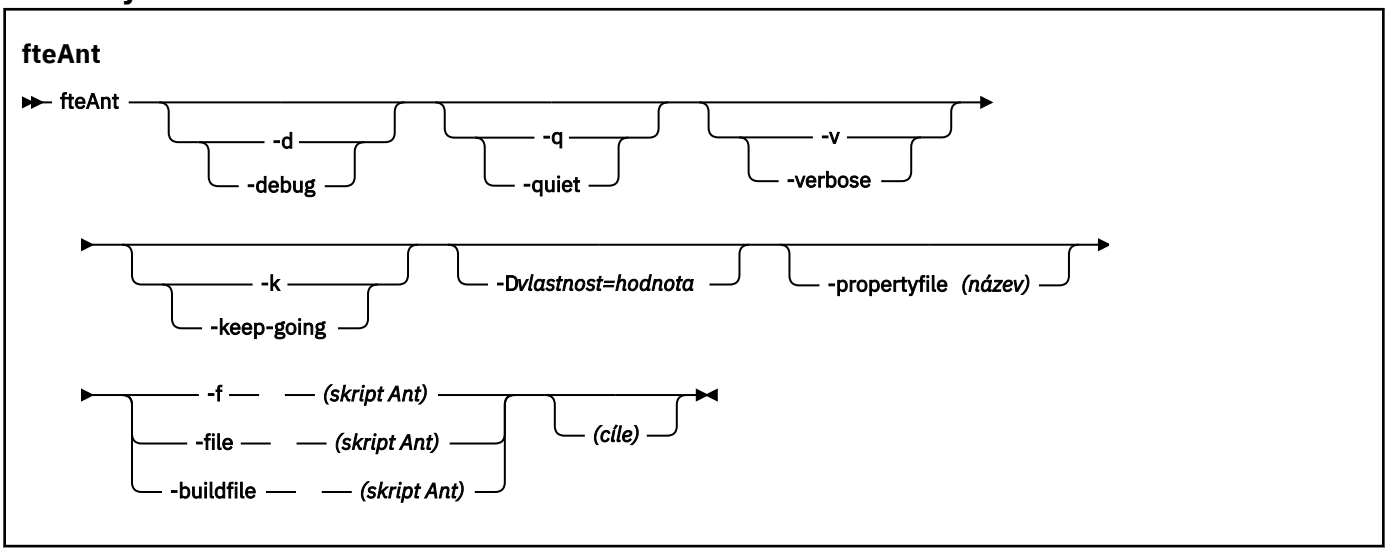
Příkaz **fteAnt** spouští skripty Ant v prostředí, které má k dispozici úlohy produktu Managed File Transfer Ant . Na rozdíl od standardního příkazu **ant** produkt **fteAnt** vyžaduje, abyste definovali skriptový soubor.

Úlohy a vnořené parametry produktu MFT Ant

Produkt Managed File Transfer poskytuje řadu úloh systému Ant , které můžete použít pro přístup ke schopnostem přenosu souborů. K dispozici je také sada vnořených parametrů; tyto parametry popisují vnořené sady prvků, které jsou společné pro několik dodaných úloh Ant.

Syntaxe příkazu **fteAnt** , parametry, příklad použití a návratové kódy jsou popsány ve zbytku tohoto tématu. Podrobnosti o úlohách Ant a vnořených parametrech, které poskytuje produkt MFT viz dílčí témata.

Syntaxe fteAnt



Parametry

-debug nebo -d
Volitelné. Generovat výstup ladění.

-tichý nebo -q
Volitelné. Generovat minimální výstup.

-verbose nebo -v
Volitelné. Generovat podrobný výstup.

-pokračovat nebo -k
Volitelné. Proveďte všechny cíle, které nejsou závislé na nezdařených cílech.

-D *vlastnost=hodnota*
Volitelné. Použijte *hodnotu* pro danou *vlastnost*. Vlastnosti nastavené pomocí parametru **-D** mají přednost před vlastnostmi nastavenými v souboru vlastností.

Pomocí vlastnosti **com.ibm.wmqfte.propertyset** můžete určit sadu voleb konfigurace, které se používají pro úlohy Ant . Jako hodnotu této vlastnosti použijte název jiného než výchozího koordinačního správce front. Úlohy produktu Ant poté používají sadu voleb konfigurace, které jsou přidruženy k tomuto nevýchozímu koordinačnímu správci front. Pokud tuto vlastnost nezadáte, použije se výchozí sada voleb konfigurace, která je založena na výchozím koordinačním správci front. Zadáte-li atribut **cmdqm** pro úlohu Ant , bude mít tento atribut přednost před sadou voleb konfigurace, které jsou určeny pro příkaz **fteAnt** . Toto chování platí bez ohledu na to, zda používáte výchozí sadu voleb konfigurace nebo zda určujete sadu s vlastností **com.ibm.wmqfte.propertyset** .

-propertyfile (*název*)
Volitelné. Načtěte všechny vlastnosti ze souboru s předností vlastností **-D** .

-f (*skript Ant*), -file (*skript Ant*) nebo -buildfile (*skript Ant*)
Povinné Určuje název skriptu Ant , který má být spuštěn.

cíle
Volitelné. Název jednoho nebo více cílů, které se mají spustit ze skriptu Ant. Pokud neuvedete hodnotu pro tento parametr, spustí se výchozí cíl pro skript.

-version
Volitelné. Zobrazí příkaz Managed File Transfer a verze Ant .

-Vážně? nebo -h
Volitelné. Zobrazuje syntaxi příkazu.

Příklad

V tomto příkladu se spustí cíl **copy** ve skriptu Ant `fte_script.xml` a příkaz zapíše výstup ladění do standardního výstupu.

```
fteAnt -d -f fte_script.xml copy
```

Návratové kódy

0
Příkaz byl úspěšně dokončen.

1
Příkaz byl neúspěšně ukončen.

Další návratové kódy stavu lze také zadat ze skriptů Ant, například pomocí úlohy Ant selhání.

Další informace viz Selhání .

Související pojmy
Začínáme s použitím skriptů Ant s produktem MFT
Související úlohy
Použití Apache Ant s MFT
Související odkazy
Ukázkové úlohy Ant pro MFT

fte: úloha awaitVýsledek Ant

Čeká na dokončení operace **fte:filecopy**, **fte:filemove**nebo **fte:call** .

Atributy

ID
Povinné Identifikuje přenos, od kterého se má čekat na výsledek. Obvykle se jedná o vlastnost nastavenou atributem idProperty úloh fte: filecopy, fte: filemovenebo fte: call .

rcproperty
Povinné Pojmenuje vlastnost, do které se má uložit návratový kód úlohy **fte:awaitoutcome** .

časový limit
Volitelné. Maximální doba v sekundách, po kterou se čeká na dokončení operace. Minimální časový limit je jedna sekunda. Pokud neuvedete hodnotu časového limitu, úloha **fte:awaitoutcome** bude čekat navždy na výsledek operace, která má být určena.

Příklad

V tomto příkladu je spuštěna kopie souboru a její identifikátor je uložen ve vlastnosti `copy.id` . Během zpracování kopie může dojít k dalšímu zpracování. Příkaz **fte:awaitoutcome** se používá k čekání na

dokončení operace kopírování. Příkaz **fte:awaitoutcome** identifikuje, na kterou operaci se má čekat s použitím identifikátoru uloženého ve vlastnosti `copy.id` . **fte:awaitoutcome** ukládá návratový kód označující výsledek operace kopírování do vlastnosti s názvem `copy.result`.

```
<-- issue a file copy request -->
<fte:filecopy
src="AGENT1@QM1"
dst="AGENT2@QM2"
idproperty="copy.id"
outcome="defer">

<fte:filespec
  srcfilespec="/home/fteuser1/file.bin"
  dstdir="/home/fteuser2"/>

</fte:filecopy>

<fte:awaitoutcome id="${copy.id}" rcProperty="copy.rc"/>

<echo>Copy id=${copy.id} rc=${copy.rc}</echo>
```

Související úlohy

Použití Apache Ant s MFT

fte: úloha volání Ant

Pomocí úlohy **fte:call** můžete vzdáleně volat skripty a programy.

Tato úloha vám umožňuje odeslat požadavek **fte:call** agentovi. Agent zpracuje tento požadavek spuštěním skriptu nebo programu a vrátí výsledek. Příkazy, které se mají volat, musí být přístupné pro agenta. Ujistěte se, že hodnota vlastnosti commandPath v souboru `agent.properties` zahrnuje umístění příkazů, které se mají volat. Veškeré informace o cestě určené vnořeným prvkem příkazu musí být relativní vzhledem k umístěním určeným vlastností commandPath . Standardně je commandPath prázdný, takže agent nemůže volat žádné příkazy. Další informace o této vlastnosti viz commandPath MFT property.

Další informace o souboru `agent.properties` viz Soubor MFT `agent.properties`.

Atributy

agent
Povinné Uvádí agenta, na kterého se má odeslat požadavek **fte:call** . Zadejte informace o agentovi ve formátu: *agentname*@*qmgrname* , kde *agentname* je název agenta a *qmgrname* je název správce front, ke kterému je tento agent přímo připojen.

cmdqm
Volitelné. Správce front příkazů, na kterého má být odeslán požadavek. Zadejte tyto informace ve formátu *qmgrname*@*host*@*port*@*channel*, kde:

- *qmgrname* je název správce front.
- *host* je volitelný název hostitele systému, kde je spuštěn správce front.
- *port* je volitelné číslo portu, na kterém správce front naslouchá.
- *channel* je volitelný kanál SVRCONN, který se má použít.

Pokud vynecháte informace o *host*, *port*nebo *channel* pro správce front příkazů, použijí se informace o připojení uvedené v souboru `command.properties` .

Upozornění: Není-li uvedena žádná hodnota pro:

- Proměnná *host* , používá se režim vazeb.
- Proměnná *port* , použije se hodnota 1414.
- Proměnná *channel* , SYSTEM.DEF.SVRCONN .

Další informace viz Soubor MFT command.properties .

Nemůžete však přeskočit atributy uprostřed, například `qmgrname@host@@channel`. Můžete mít například `qmgrname@host`, `qmgrname@host@portnebo qmgrname@hostport@@channel`.

MFT rozdělí daný atribut pomocí oddělovače `@` . V závislosti na počtu nalezených tokenů bere první token jako *qmgrname*, druhý jako *hostitel*, třetí jako *port* a nakonec *kanál*.

Další informace viz Soubor MFT command.properties.

Pomocí vlastnosti **`com.ibm.wmqfte.propertySet`** můžete určit, který soubor `command.properties` se má použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost `com.ibm.wmqfte.ant.commandQueueManager` , pokud je tato vlastnost nastavena. Není-li vlastnost `com.ibm.wmqfte.ant.commandQueueManager` nastavena, dojde k pokusu o připojení k výchozímu správci front definovanému v souboru `command.properties` . Formát vlastnosti `com.ibm.wmqfte.ant.commandQueueManager` je stejný jako atribut cmdqm , tj. `qmgrname@host@port@channel`.

idproperty idproperty
Volitelné, pokud jste nezadali `outcome` hodnotu `defer`. Určuje název vlastnosti, ke které má být přiřazen identifikátor přenosu. Identifikátory přenosu jsou generovány v okamžiku, kdy je odeslán požadavek na přenos, a můžete použít identifikátory přenosu ke sledování průběhu přenosu, diagnostice problémů s přenosem a zrušení přenosu.

Tuto vlastnost nelze určit, pokud jste zadali také vlastnost `outcome` s hodnotou `ignore`. Pokud jste však zadali také vlastnost `outcome` `defer`, musíte zadat hodnotu `idproperty` .

JOBNAME
Volitelné. Přiřadí název úlohy k požadavku **`fte:call`** . Názvy úloh můžete použít k vytvoření logických skupin přenosů. Pomocí úlohy “fte: úloha uuid Ant” na stránce 2107 vygenerujte pseudojedinečné názvy úloh. Pokud nepoužijete atribut jobname , úloha standardně použije hodnotu vlastnosti `com.ibm.wmqfte.ant.jobName` , pokud je tato vlastnost nastavena. Pokud tuto vlastnost nenastavíte, k požadavku **`fte:call`** nebude přidružen žádný název úlohy.

původní uživatel
Volitelné. Uvádí identifikátor původního uživatele, který se má přidružit k požadavku **`fte:call`** . Pokud nepoužijete atribut originuser , úloha standardně použije ID uživatele, které se používá ke spuštění skriptu Ant.

výsledek
Volitelné. Určuje, zda úloha čeká na dokončení operace **`fte:call`** , než vrátí řízení skriptu Ant . Zadejte jednu z následujících voleb:

čekání na
Úloha před návratem čeká na dokončení operace **`fte:call`** . Je-li zadána hodnota `outcome` z `await` , je atribut `idproperty` nepovinný.

defer
Úloha se vrátí po odeslání požadavku **`fte:call`** a předpokládá, že výsledek operace volání bude zpracován později pomocí úloh awaitvýsledek nebo ignorevýsledek . Je-li zadána hodnota `outcome` z hodnoty `defer` , je vyžadován atribut `idproperty` .

Ignorovat
Pokud není výsledek operace **`fte:call`** důležitý, můžete zadat hodnotu `ignore`. Úloha se poté vrátí ihned po odeslání požadavku **`fte:call`** bez přidělení prostředků pro sledování výsledku příkazu. Je-li zadána hodnota `outcome` z `ignore` , nelze zadat atribut `idproperty` .

Pokud neuvedete atribut výsledek , úloha standardně použije hodnotu `await`.

rcproperty (vlastnost)
Volitelné. Určuje název vlastnosti, ke které má být přiřazen kód výsledku požadavku **`fte:call`** . Výsledkový kód odráží celkový výsledek požadavku **`fte:call`** .

Tuto vlastnost nelze zadat, pokud jste zadali také vlastnost `outcome` `ignore` nebo `defer`. Pokud jste však zadali výsledek `await`, musíte zadat hodnotu `rcproperty` .

Parametry určené jako vnořené prvky

fte: příkaz
Uvádí příkaz, který má agent volat. K dané operaci**fte:call** můžete přidružit pouze jeden prvek `fte:command` . Příkaz, který má být volán, musí být umístěn v cestě určené vlastností commandPath v souboru `agent.properties` agenta.

fte: metadata
Můžete určit metadata, která se mají přidružit k operaci volání. Tato metadata jsou zaznamenána ve zprávách protokolu generovaných operací volání. K danému prvku přenosu lze přidružit pouze jeden blok metadat. Tento blok však může obsahovat mnoho částí metadat.

Příklad

Tento příklad ukazuje, jak volat příkaz na adrese AGENT1 spuštěný ve správci front QM1. Příkaz, který se má volat, je skript `command.sh`a skript se volá s jedním argumentem `xyz`. Příkaz `command.sh` je umístěn na cestě určené vlastností commandPath v souboru `agent.properties` agenta.

```
<fte:call cmdqm="QM0@localhost@1414@SYSTEM.DEF.SVRCONN"
            agent="AGENT1@QM1"
            rcproperty="call.rc"
            origuser="bob"
            jobname="${job.id}">

    <fte:command command="command.sh" successrc="1" retrycount="5" retrywait="30">
        <fte:arg value="xyz"/>
    </fte:command>

    <fte:metadata>
        <fte:entry name="org.foo.accountName" value="BDG3R"/>
    </fte:metadata>

</fte:call>
```

Související úlohy

Použití Apache Ant s MFT

fte: zrušit úlohu Ant

Zruší spravovaný přenos nebo volání Managed File Transfer . Spravovaný přenos mohl být vytvořen pomocí úloh **fte:filecopy** nebo **fte:filemove** . Je možné, že bylo vytvořeno spravované volání pomocí úlohy **fte:call** .

Atributy

agent
Povinné Uvádí agenta, na kterého se má odeslat požadavek **fte:cancel** . Hodnota je ve formátu: *agentname*@*qmgrname* , kde *agentname* je název agenta a *qmgrname* je název správce front, ke kterému je tento agent přímo připojen.

cmdqm
Volitelné. Správce front příkazů, na kterého má být odeslán požadavek. Zadejte tyto informace ve formátu *qmgrname*@*host*@*port*@*channel*, kde:

- *qmgrname* je název správce front.
- *host* je volitelný název hostitele systému, kde je spuštěn správce front.
- *port* je volitelné číslo portu, na kterém správce front naslouchá.
- *channel* je volitelný kanál SVRCONN, který se má použít.

Pokud vynecháte informace o *host*, *port*nebo *channel* pro správce front příkazů, použijí se informace o připojení uvedené v souboru `command.properties` .

Upozornění: Není-li uvedena žádná hodnota pro:

- Proměnná *host* , používá se režim vazeb.

- Proměnná *port* , použije se hodnota 1414.
- Proměnná *channel* , SYSTEM.DEF.SVRCONN .

Další informace viz Soubor MFT command.properties .

Nemůžete však přeskočit atributy uprostřed, například `qmgrname@host@@channel`. Můžete mít například `qmgrname@host`, `qmgrname@host@portnebo qmgrname@hostport@@channel`.

MFT rozdělí daný atribut pomocí oddělovače `@` . V závislosti na počtu nalezených tokenů bere první token jako *qmgrname*, druhý jako *hostitel*, třetí jako *port* a nakonec *kanál*.

Další informace viz Soubor MFT command.properties.

Pomocí vlastnosti **`com.ibm.wmqfte.propertySet`** můžete určit, který soubor `command.properties` se má použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost `com.ibm.wmqfte.ant.commandQueueManager` , pokud je tato vlastnost nastavena. Není-li vlastnost `com.ibm.wmqfte.ant.commandQueueManager` nastavena, dojde k pokusu o připojení k výchozímu správci front definovanému v souboru `command.properties` . Formát vlastnosti `com.ibm.wmqfte.ant.commandQueueManager` je stejný jako atribut cmdqm , tj. `qmgrname@host@port@channel`.

ID
Povinné Určuje identifikátor přenosu, který má být zrušen. Identifikátory přenosu jsou generovány v okamžiku, kdy je požadavek na přenos odeslán úlohami fte: filecopy a fte: filemove .

původní uživatel
Volitelné. Uvádí identifikátor původního uživatele, který se má přidružit k požadavku **`cancel`** . Pokud se atribut originuser nepoužije, standardně se použije ID uživatele, které se použije ke spuštění skriptu Ant .

Příklad

Příklad odešle požadavek **`fte:cancel`** správci front příkazů `qm0`. Cílem požadavku **`fte:cancel`** je `agent1` ve správci front `qm1` identifikátor přenosu naplněný proměnnou `transfer.id` . Požadavek se spustí pomocí ID uživatele "bob" .

```
<fte:cancel cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
            agent="agent1@qm1"
            id="${transfer.id}"
            originuser="bob"/>
```

Související úlohy

Použití Apache Ant s MFT

fte: úloha filecopy Ant

Úloha **`fte:filecopy`** kopíruje soubory mezi agenty Managed File Transfer . Soubor není odstraněn ze zdrojového agenta.

Atributy

cmdqm
Volitelné. Správce front příkazů, na kterého má být odeslán požadavek. Zadejte tyto informace ve formátu *`qmgrname@host@port@channel`*, kde:

- *`qmgrname`* je název správce front.
- *`host`* je volitelný název hostitele systému, kde je spuštěn správce front.
- *`port`* je volitelné číslo portu, na kterém správce front naslouchá.
- *`channel`* je volitelný kanál SVRCONN, který se má použít.

Pokud vynecháte informace o *host*, *port*nebo *channel* pro správce front příkazů, použijí se informace o připojení uvedené v souboru command.properties .

Upozornění: Není-li uvedena žádná hodnota pro:

- Proměnná *host* , používá se režim vazeb.
- Proměnná *port* , použije se hodnota 1414.
- Proměnná *channel* , SYSTEM.DEF.SVRCONN .

Další informace viz Soubor MFT command.properties .

Nemůžete však přeskočit atributy uprostřed, například qmgrname@host@@channel. Můžete mít například qmgrname@host, qmgrname@host@portnebo qmgrname@hostport@@channel.

MFT rozdělí daný atribut pomocí oddělovače @ . V závislosti na počtu nalezených tokenů bere první token jako *qmgrname*, druhý jako *hostitel*, třetí jako *port* a nakonec *kanál*.

Další informace viz Soubor MFT command.properties.

Pomocí vlastnosti **com.ibm.wmqfte.propertySet** můžete určit, který soubor command.properties se má použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost com.ibm.wmqfte.ant.commandQueueManager , pokud je tato vlastnost nastavena. Není-li vlastnost com.ibm.wmqfte.ant.commandQueueManager nastavena, dojde k pokusu o připojení k výchozímu správci front definovanému v souboru command.properties . Formát vlastnosti com.ibm.wmqfte.ant.commandQueueManager je stejný jako atribut cmdqm , tj. qmgrname@host@port@channel.

dst
Povinné Určuje cílového agenta pro operaci kopírování. Zadejte tyto informace ve formátu: agentname@qmgrname , kde agentname je název cílového agenta a qmgrname je název správce front, ke kterému je tento agent přímo připojen.

idproperty idproperty
Volitelné, pokud jste nezadali outcome hodnotu defer. Určuje název vlastnosti, ke které má být přiřazen identifikátor přenosu. Identifikátory přenosu jsou generovány v okamžiku, kdy je odeslán požadavek na přenos, a můžete použít identifikátory přenosu ke sledování průběhu přenosu, diagnostice problémů s přenosem a zrušení přenosu.

Tuto vlastnost nelze určit, pokud jste zadali také vlastnost outcome s hodnotou ignore. Pokud jste však zadali také vlastnost outcome defer, musíte zadat hodnotu idproperty .

JOBNAME
Volitelné. Přiřadí název úlohy k požadavku na kopírování. Názvy úloh můžete použít k vytvoření logických skupin přenosů. Pomocí úlohy “fte: úloha uuid Ant” na stránce 2107 vygenerujte pseudojedinečné názvy úloh. Pokud nepoužijete atribut jobname , úloha standardně použije hodnotu vlastnosti com.ibm.wmqfte.ant.jobName , pokud je tato vlastnost nastavena. Pokud tuto vlastnost nenastavíte, k požadavku na kopírování nebude přidružen žádný název úlohy.

původní uživatel
Volitelné. Uvádí identifikátor původního uživatele, který se má přidružit k požadavku na kopírování. Pokud nepoužijete atribut originuser , standardně se použije ID uživatele, které se používá ke spuštění skriptu Ant .

výsledek
Volitelné. Určuje, zda úloha čeká na dokončení operace kopírování, než vrátí řízení skriptu Ant . Zadejte jednu z následujících voleb:

čekání na
Úloha před návratem čeká na dokončení operace kopírování. Je-li zadána hodnota outcome z await , je atribut idproperty nepovinný.

defer
Úloha se vrátí, jakmile je odeslán požadavek na kopírování, a předpokládá, že výsledek operace kopírování je zpracován později pomocí úloh “fte: úloha awaitVýsledek Ant” na stránce 2094 nebo “fte: úloha ignorevýsledek Ant” na stránce 2106 . Je-li zadána hodnota `outcome` z hodnoty `defer` , je vyžadován atribut `idproperty` .

Ignorovat
Pokud není výsledek operace kopírování důležitý, můžete zadat hodnotu `ignore`. Úloha se poté vrátí ihned po odeslání požadavku na kopírování, aniž by byly alokovány prostředky pro sledování výsledku přenosu. Je-li zadána hodnota `outcome` z `ignore` , nelze zadat atribut `idproperty` .

Pokud neuvedete atribut výsledek , úloha standardně použije hodnotu `await`.

priorita
Volitelné. Uvádí prioritu, která se má přidružit k požadavku na kopírování. Obecně mají požadavky na přenos s vyšší prioritou přednost před požadavky s nižší prioritou. Hodnota priority musí být v rozsahu 0-9 (včetně). Hodnota priority 0 je nejnižší priorita a hodnota 9 je nejvyšší priorita. Pokud neuvedete atribut `priority` , přenos bude standardně nastaven na prioritu 0.

rcproperty (vlastnost)
Volitelné. Určuje název vlastnosti, ke které má být přiřazen kód výsledku požadavku na kopírování. Výsledkový kód odráží celkový výsledek požadavku na kopírování.

Tuto vlastnost nelze zadat, pokud jste zadali také vlastnost `outcome` `ignore` nebo `defer`. Pokud však zadáte výsledek `await`, musíte zadat hodnotu `rcproperty` .

Časový limit transferRecovery
Volitelné. Nastavuje dobu v sekundách, během které se zdrojový agent stále pokouší obnovit pozastavený přenos souborů. Zadejte jednu z následujících voleb:

-1
Agent se nadále pokouší obnovit pozastavený přenos, dokud není přenos dokončen. Použití této volby je ekvivalentem výchozího chování agenta, když není vlastnost nastavena.

0
Agent zastaví přenos souborů, jakmile vstoupí do obnovy.

>0
Agent se nadále pokouší obnovit pozastavený přenos po dobu v sekundách nastavenou uvedenou kladnou celočíselnou hodnotou. Například:

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
              src="agent1@qm1" dst="agent2@qm2"
              rcproperty="copy.result" transferRecoveryTimeout="21600">

    <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/
file.bin"/>

</fte:filecopy>
```

označuje, že agent se stále pokouší obnovit přenos po dobu 6 hodin od okamžiku, kdy vstoupí do obnovy. Maximální hodnota tohoto atributu je 999999999.

Zadáním hodnoty časového limitu obnovy přenosu ji nastavíte na základě jednotlivých přenosů. Chcete-li nastavit globální hodnotu pro všechny přenosy v síti Managed File Transfer , můžete přidat vlastnost do pole Vlastnosti časového limitu obnovy přenosu. Další informace naleznete v tématu Volba časového limitu pro přenosy v obnově.

src
Povinné Určuje zdrojového agenta pro operaci kopírování. Zadejte tyto informace ve formátu: *agentname*@*qmgrname* , kde *agentname* je název zdrojového agenta a *qmgrname* je název správce front, ke kterému je tento agent přímo připojen.

Parametry určené jako vnořené prvky

fte: specifikace souboru
Povinné Musíte uvést alespoň jednu specifikaci souboru, která identifikuje soubory, které se mají kopírovat. V případě potřeby můžete zadat více než jednu specifikaci souboru. Další informace viz “fte: vnořený prvek filespec Ant” na stránce 2108.

fte: metadata
Můžete určit metadata, která se mají přidružit k operaci kopírování. Tato metadata jsou přenášena spolu s přenosem a jsou zaznamenána ve zprávách protokolu generovaných přenosem. K danému prvku přenosu lze přidružit pouze jeden blok metadat. Tento blok však může obsahovat mnoho částí metadat. Další informace viz téma fte: metadata .

fte: předtisková
Uvádí vyvolání programu, které se má provést na zdrojovém agentovi před spuštěním přenosu. K danému přenosu můžete přidružit pouze jeden prvek `fte:presrc` . Další informace viz téma vyvolání programu .

fte: předchozí
Uvádí vyvolání programu, které se má provést na cílovém agentovi před spuštěním přenosu. K danému přenosu můžete přidružit pouze jeden prvek `fte:predst` . Další informace viz téma vyvolání programu .

fte: postsrc
Uvádí vyvolání programu, které se má provést na zdrojovém agentovi po dokončení přenosu. K danému přenosu můžete přidružit pouze jeden prvek `fte:postsrc` . Další informace viz téma vyvolání programu .

fte: postdst (postdst)
Uvádí vyvolání programu, které se má provést na cílovém agentu po dokončení přenosu. K danému přenosu můžete přidružit pouze jeden prvek `fte:postdst` . Další informace viz téma vyvolání programu .

Pokud fte:presrc, fte:predst, fte:postsrc, fte:postdst a ukončení nevrátí stav úspěchu, pravidla jsou v uvedeném pořadí:

1. Spusťte uživatelské procedury spuštění zdroje. Pokud uživatelské procedury spuštění zdroje selžou, přenos se nezdaří a dále se nespustí nic.
2. Spusťte volání před zdrojem (je-li k dispozici). Pokud volání před zdrojem selže, přenos se nezdaří a nic dalšího se nespustí.
3. Spusťte uživatelské procedury spuštění cíle. Dojde-li k selhání ukončení spuštění cíle, přenos se nezdaří a dále se nespustí nic.
4. Spusťte volání před cílem (je-li přítomno). Pokud volání před cílem selže, přenos se nezdaří a nic dalšího se nespustí.
5. Proveďte přenosy souborů.
6. Spusťte uživatelské procedury konce cíle. Pro tyto uživatelské procedury neexistuje žádný stav selhání.
7. Pokud je přenos úspěšný (pokud jsou některé soubory úspěšně přeneseny, je to považováno za úspěšné), spusťte volání po cíli (je-li k dispozici). Pokud volání po cíli selže, přenos se nezdaří.
8. Spusťte koncové uživatelské procedury zdroje. Pro tyto uživatelské procedury neexistuje žádný stav selhání.
9. Je-li přenos úspěšný, spusťte volání post-source (je-li k dispozici). Pokud volání po zdroji selže, přenos se nezdaří.

Příklady

Tento příklad ukazuje základní přenos souborů mezi `agent1` a `agent2`. Příkaz ke spuštění přenosu souborů je odeslán správci front s názvem `qm0`, pomocí připojení v režimu transportu klienta. Výsledek operace přenosu souborů je přiřazen k vlastnosti s názvem `copy.result`.

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
              src="agent1@qm1" dst="agent2@qm2"
              rcproperty="copy.result">

    <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>

</fte:filecopy>
```

Tento příklad zobrazuje stejný přenos souborů, ale s přidáním metadat a spuštěním programu, který se má uskutečnit na zdrojovém agentovi po dokončení přenosu.

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
              src="agent1@qm"1 dst="agent2@qm2"
              rcproperty="copy.result">


    <fte:metadata>
            <fte:entry name="org.example.departId" value="ACCOUNTS"/>
            <fte:entry name="org.example.batchGroup" value="A1"/>
    </fte:metadata>

    <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>

    <fte:postsrc command="/home/fteuser2/scripts/post.sh" successrc="1" >
            <fte:arg value="/home/fteuser2/file.bin"/>
  </fte:postsrc>
</fte:filecopy>
```

Související pojmy

Volba časového limitu pro přenosy souborů při obnově

Související úlohy

Použití Apache Ant s MFT

fte: úloha přesunu souboru Ant

Úloha **`fte:filemove`** přesouvá soubory mezi agenty Managed File Transfer . Když byl soubor úspěšně přenesen ze zdrojového agenta do cílového agenta, je odstraněn ze zdrojového agenta.

Atributy

cmdqm

Volitelné. Správce front příkazů, na kterého má být odeslán požadavek. Zadejte tyto informace ve formátu *qmgrname*@*host*@*port*@*channel*, kde:

- *qmgrname* je název správce front.
- *host* je volitelný název hostitele systému, kde je spuštěn správce front.
- *port* je volitelné číslo portu, na kterém správce front naslouchá.
- *channel* je volitelný kanál SVRCONN, který se má použít.

Pokud vynecháte informace o *host*, *port*nebo *channel* pro správce front příkazů, použijí se informace o připojení uvedené v souboru `command.properties` .

Upozornění: Není-li uvedena žádná hodnota pro:

- Proměnná *host* , používá se režim vazeb.
- Proměnná *port* , použije se hodnota 1414.
- Proměnná *channel* , SYSTEM.DEF.SVRCONN .

Další informace viz Soubor MFT command.properties .

Nemůžete však přeskočit atributy uprostřed, například `qmgrname@host@@channel`. Můžete mít například `qmgrname@host`, `qmgrname@host@portnebo qmgrname@hostport@@channel`.

MFT rozdělí daný atribut pomocí oddělovače `@` . V závislosti na počtu nalezených tokenů bere první token jako *qmgrname*, druhý jako *hostitel*, třetí jako *port* a nakonec *kanál*.

Další informace viz Soubor MFT command.properties.

Pomocí vlastnosti **`com.ibm.wmqfte.propertySet`** můžete určit, který soubor `command.properties` se má použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost `com.ibm.wmqfte.ant.commandQueueManager` , pokud je tato vlastnost nastavena. Není-li vlastnost `com.ibm.wmqfte.ant.commandQueueManager` nastavena, dojde k pokusu o připojení k výchozímu správci front definovanému v souboru `command.properties` . Formát vlastnosti `com.ibm.wmqfte.ant.commandQueueManager` je stejný jako atribut cmdqm , tj. `qmgrname@host@port@channel`.

dst
Povinné Určuje cílového agenta pro operaci kopírování. Zadejte tyto informace ve formátu: *`agentname`*`@`*`qmgrname`* , kde *`agentname`* je název cílového agenta a *`qmgrname`* je název správce front, ke kterému je tento agent přímo připojen.

idproperty idproperty
Volitelné, pokud jste nezadali `outcome` hodnotu `defer`. Určuje název vlastnosti, ke které má být přiřazen identifikátor přenosu. Identifikátory přenosu jsou generovány v okamžiku, kdy je odeslán požadavek na přenos, a můžete použít identifikátory přenosu ke sledování průběhu přenosu, diagnostice problémů s přenosem a zrušení přenosu.

Tuto vlastnost nelze určit, pokud jste zadali také vlastnost `outcome` s hodnotou `ignore`. Pokud jste však zadali také vlastnost `outcome` `defer`, musíte zadat hodnotu `idproperty` .

JOBNAME
Volitelné. Přiřadí název úlohy k požadavku na přesun. Názvy úloh můžete použít k vytvoření logických skupin přenosů. Pomocí úlohy fte: uuid vygenerujte pseudojedinečné názvy úloh. Pokud nepoužijete atribut jobname , úloha standardně použije hodnotu vlastnosti `com.ibm.wmqfte.ant.jobName` , pokud je tato vlastnost nastavena. Pokud tuto vlastnost nenastavíte, k požadavku na přesun nebude přidružen žádný název úlohy.

původní uživatel
Volitelné. Uvádí identifikátor původního uživatele, který se má přidružit k požadavku na přesun. Pokud nepoužijete atribut originuser , standardně se použije ID uživatele, které se používá ke spuštění skriptu Ant .

výsledek
Volitelné. Určuje, zda úloha čeká na dokončení operace přesunu, než vrátí řízení skriptu Ant . Zadejte jednu z následujících voleb:

čekání na
Úloha před návratem čeká na dokončení operace přesunu. Je-li zadána hodnota `outcome` z `await` , je atribut `idproperty` nepovinný.

defer
Úloha se vrátí, jakmile je odeslán požadavek na přesun, a předpokládá, že výsledek operace přesunu je zpracován později pomocí úlohy “fte: úloha awaitVýsledek Ant” na stránce 2094 nebo “fte: úloha ignorevýsledek Ant” na stránce 2106 . Je-li zadána hodnota `outcome` z hodnoty `defer` , je vyžadován atribut `idproperty` .

Ignorovat
Pokud není výsledek operace přesunu důležitý, můžete zadat hodnotu `ignore`. Úloha se poté vrátí ihned po odeslání požadavku na přesun, aniž by byly alokovány prostředky pro sledování výsledku přenosu. Je-li zadána hodnota `outcome` z `ignore` , nelze zadat atribut `idproperty` .

Pokud neuvedete atribut výsledek , úloha standardně použije hodnotu `await`.

priorita
Volitelné. Uvádí prioritu, která se má přidružit k požadavku na přesun. Obecně mají požadavky na přenos s vyšší prioritou přednost před požadavky s nižší prioritou. Hodnota priority musí být v rozsahu 0-9 (včetně). Hodnota priority 0 je nejnižší priorita a hodnota 9 je nejvyšší priorita. Pokud neuvedete atribut `priority` , přenos bude standardně nastaven na prioritu 0.

rcproperty (vlastnost)
Volitelné. Určuje název vlastnosti, ke které má být přiřazen kód výsledku požadavku na přesun. Výsledkový kód odráží celkový výsledek požadavku na přesun.

Tuto vlastnost nelze zadat, pokud jste zadali také vlastnost `outcome` `ignore` nebo `defer`. Pokud jste však zadali výsledek `await`, musíte zadat hodnotu `rcproperty` .

Časový limit transferRecovery
Volitelné. Nastavuje dobu v sekundách, během které se zdrojový agent stále pokouší obnovit pozastavený přenos souborů. Zadejte jednu z následujících voleb:

-1
Agent se nadále pokouší obnovit pozastavený přenos, dokud není přenos dokončen. Použití této volby je ekvivalentem výchozího chování agenta, když není vlastnost nastavena.

0
Agent zastaví přenos souborů, jakmile vstoupí do obnovy.

>0
Agent se nadále pokouší obnovit pozastavený přenos po dobu v sekundách nastavenou uvedenou kladnou celočíselnou hodnotou. Například:

```
<fte:filemove cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
              src=agent1@qm1 dst="agent2@qm2"
              rcproperty="move.result" transferRecoveryTimeout="21600">

    <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/
file.bin"/>

</fte:filemove
```

označuje, že agent se stále pokouší obnovit přenos po dobu 6 hodin od okamžiku, kdy vstoupí do obnovy. Maximální hodnota tohoto atributu je 999999999.

Zadáním hodnoty časového limitu obnovy přenosu ji nastavíte na základě jednotlivých přenosů. Chcete-li nastavit globální hodnotu pro všechny přenosy v síti Managed File Transfer , můžete přidat vlastnost do pole Vlastnosti časového limitu obnovy přenosu. Další informace naleznete v tématu Volba časového limitu pro přenosy v obnově.

src
Povinné Určuje zdrojového agenta pro operaci přesunu. Zadejte tyto informace ve formátu: *agentname*@*qmgrname* , kde *agentname* je název zdrojového agenta a *qmgrname* je název správce front, ke kterému je tento agent přímo připojen.

Parametry určené jako vnořené prvky

fte: specifikace souboru
Povinné Musíte uvést alespoň jednu specifikaci souboru, která identifikuje soubory, které se mají přesunout. V případě potřeby můžete zadat více než jednu specifikaci souboru. Další informace viz “fte: vnořený prvek filespec Ant” na stránce 2108.

fte: metadata
Volitelné. Můžete určit metadata, která se mají přidružit k operaci přesunu souboru. Tato metadata jsou přenášena spolu s přenosem a jsou zaznamenána ve zprávách protokolu generovaných přenosem. K danému prvku přenosu lze přidružit pouze jeden blok metadat. Tento blok však může obsahovat mnoho částí metadat. Další informace viz téma fte: metadata .

fte: předtisková
Volitelné. Uvádí vyvolání programu, které se má provést na zdrojovém agentovi před spuštěním přenosu. K danému přenosu můžete přidružit pouze jeden prvek `fte:presrc` . Další informace viz téma vyvolání programu .

fte: předchozí
Volitelné. Uvádí vyvolání programu, které se má provést na cílovém agentovi před spuštěním přenosu. K danému přenosu můžete přidružit pouze jeden prvek `fte:predst` . Další informace viz téma vyvolání programu .

fte: postsrc
Volitelné. Uvádí vyvolání programu, které se má provést na zdrojovém agentovi po dokončení přenosu. K danému přenosu můžete přidružit pouze jeden prvek `fte:postsrc` . Další informace viz téma vyvolání programu .

fte: postdst (postdst)
Volitelné. Uvádí vyvolání programu, které se má provést na cílovém agentu po dokončení přenosu. K danému přenosu můžete přidružit pouze jeden prvek `fte:postdst` . Další informace viz téma vyvolání programu .

Pokud fte:presrc, fte:predst, fte:postsrc, fte:postdst a ukončení nevrátí stav úspěchu, pravidla jsou v uvedeném pořadí:

1. Spusťte uživatelské procedury spuštění zdroje. Pokud uživatelské procedury spuštění zdroje selžou, přenos se nezdaří a dále se nespustí nic.
2. Spusťte volání před zdrojem (je-li k dispozici). Pokud volání před zdrojem selže, přenos se nezdaří a nic dalšího se nespustí.
3. Spusťte uživatelské procedury spuštění cíle. Dojde-li k selhání ukončení spuštění cíle, přenos se nezdaří a dále se nespustí nic.
4. Spusťte volání před cílem (je-li přítomno). Pokud volání před cílem selže, přenos se nezdaří a nic dalšího se nespustí.
5. Proveďte přenosy souborů.
6. Spusťte uživatelské procedury konce cíle. Pro tyto uživatelské procedury neexistuje žádný stav selhání.
7. Pokud je přenos úspěšný (pokud jsou některé soubory úspěšně přeneseny, je přenos považován za úspěšný), spusťte volání po cíli (je-li k dispozici). Pokud volání po cíli selže, přenos se nezdaří.
8. Spusťte koncové uživatelské procedury zdroje. Pro tyto uživatelské procedury neexistuje žádný stav selhání.
9. Je-li přenos úspěšný, spusťte volání post-source (je-li k dispozici). Pokud volání po zdroji selže, přenos se nezdaří.

Příklady

Tento příklad ukazuje základní přesun souboru mezi `agent1` a `agent2`. Příkaz ke spuštění přesunu souboru je odeslán správci front s názvem `qm0`, pomocí připojení v režimu transportu klienta. Výsledek operace přenosu souborů je přiřazen k vlastnosti s názvem `move.result`.

```
<fte:filemove cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
              src="agent1@qm1" dst="agent2@qm2"
              rcproperty="move.result">

    <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>

</fte:filemove>
```

Související pojmy

Volba časového limitu pro přenosy souborů při obnově

Související úlohy

Použití Apache Ant s MFT

fte: úloha ignorevýsledek Ant

Ignorujte výsledek příkazu **fte:filecopy**, **fte:filemove**nebo **fte:call** . Když určíte úlohu **fte:filecopy**, **fte:filemove**nebo **fte:call** , která má mít výsledek `defer`, úloha Ant přidělí prostředky pro sledování tohoto výsledku. Pokud již o výsledek nemáte zájem, můžete tyto prostředky uvolnit pomocí úlohy **fte:ignoreoutcome** .

Atributy

ID

Povinné Identifikuje výsledek, který již není předmětem zájmu. Tento identifikátor obvykle zadáváte pomocí vlastnosti, kterou nastavíte pomocí atributu `idproperty` úlohy “fte: úloha filecopy Ant” na stránce 2098, “fte: úloha přesunu souboru Ant” na stránce 2102nebo “fte: úloha volání Ant” na stránce 2095 .

Příklad

Tento příklad ukazuje, jak můžete použít úlohu fte: ignorevýsledek k uvolnění prostředků přidělených ke sledování výsledku předchozí úlohy “fte: úloha filecopy Ant” na stránce 2098 .

```
<!-- issue a file copy request -->
<fte:filecopy cmdqm="qm1@localhost@1414@SYSTEM.DEF.SVRCONN"
                  src="agent1@qm1" dst="agent1@qm1"
                  idproperty="copy.id"
                  outcome="defer"/>

<!-- do some other things -->

<!-- decide that the result of the copy is not interesting -->
<fte:ignoreoutcome id="${copy.id}"/>
```

Související úlohy

Použití Apache Ant s MFT

fte: úloha příkazu ping Ant

Tato úloha produktu IBM MQ Managed File Transfer Ant odešle agentovi příkaz ping, aby vyvolal odezvu, a tak určí, zda je agent schopen zpracovat přenosy.

Poznámka: IBM WebSphere MQ File Transfer Edition (FTE) již není podporovaným produktem. Chcete-li migrovat z FTE na komponentu Managed File Transfer v produktu IBM MQ, viz Migrace Managed File Transfer.

Atributy

agent

Povinné Uvádí agenta, na kterého se má odeslat požadavek **fte:ping** . Hodnota je ve formátu: *agentname*@*qmgrname* , kde *agentname* je název agenta a *qmgrname* je název správce front, ke kterému je tento agent přímo připojen.

cmdqm

Volitelné. Správce front příkazů, na kterého má být odeslán požadavek. Zadejte tyto informace ve formátu *qmgrname*@*host*@*port*@*channel*, kde:

- *qmgrname* je název správce front.
- *host* je volitelný název hostitele systému, kde je spuštěn správce front.
- *port* je volitelné číslo portu, na kterém správce front naslouchá.
- *channel* je volitelný kanál SVRCONN, který se má použít.

Pokud vynecháte informace o *host*, *port*nebo *channel* pro správce front příkazů, použijí se informace o připojení uvedené v souboru `command.properties` .

Upozornění: Není-li uvedena žádná hodnota pro:

- Proměnná *host* , používá se režim vazeb.
- Proměnná *port* , použije se hodnota 1414.
- Proměnná *channel* , SYSTEM.DEF.SVRCONN .

Další informace viz Soubor MFT command.properties .

Nemůžete však přeskočit atributy uprostřed, například `qmgrname@host@@channel`. Můžete mít například `qmgrname@host`, `qmgrname@host@portnebo qmgrname@hostport@@channel`.

MFT rozdělí daný atribut pomocí oddělovače `@` . V závislosti na počtu nalezených tokenů bere první token jako *qmgrname*, druhý jako *hostitel*, třetí jako *port* a nakonec *kanál*.

Další informace viz Soubor MFT command.properties.

Pomocí vlastnosti **`com.ibm.wmqfte.propertySet`** můžete určit, který soubor `command.properties` se má použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost `com.ibm.wmqfte.ant.commandQueueManager` , pokud je tato vlastnost nastavena. Není-li vlastnost `com.ibm.wmqfte.ant.commandQueueManager` nastavena, dojde k pokusu o připojení k výchozímu správci front definovanému v souboru `command.properties` . Formát vlastnosti `com.ibm.wmqfte.ant.commandQueueManager` je stejný jako atribut cmdqm , tj. `qmgrname@host@port@channel`.

rcproperty
Povinné Pojmenuje vlastnost, do které se má uložit návratový kód operace **`ping`** .

časový limit
Volitelné. Maximální doba v sekundách, po kterou má úloha čekat na odezvu agenta. Minimální časový limit je nula sekund, avšak časový limit minus jedna může být také uveden tak, aby příkaz čekal navždy na odezvu agenta. Není-li pro parametr `timeout` uvedena žádná hodnota, předvolba je čekat až 5 sekund, než agent odpoví.

Příklad

Tento příklad odešle požadavek **`fte:ping`** na server `agent1` hostovaný `qm1`. Požadavek **`fte:ping`** čeká 15 sekund na odezvu agenta. Výsledek požadavku **`fte:ping`** je uložen ve vlastnosti s názvem `ping.rc`.

```
<fte:ping agent="agent1@qm1" rcproperty="ping.rc" timeout="15"/>
```

Návratové kódy

0
Příkaz byl úspěšně dokončen.

2
Vypršel časový limit příkazu.

Související úlohy
Použití Apache Ant s MFT

fte: úloha uuid Ant

Vygeneruje pseudonáhodný jedinečný identifikátor a přiřadí jej k dané vlastnosti. Tento identifikátor můžete například použít ke generování názvů úloh pro jiné operace přenosu souborů.

Atributy

délka
Povinné Číselná délka identifikátoru UUID, který se má generovat. Tato hodnota délky nezahrnuje délku žádné předpony určenou parametrem **prefix** .

vlastnost
Povinné Název vlastnosti, ke které má být přiřazen vygenerovaný klíč UUID.

Předpona
Volitelné. Předpona, která se má přidat k vygenerovanému identifikátoru UUID. Tato předpona není započítána jako součást délky UUID, jak je určeno parametrem **length** .

Příklad

Tento příklad definuje klíč UUID, který začíná písmeny ABC následovanými 16 pseudo-náhodnými hexadecimálními znaky. Klíč UUID je přiřazen k vlastnosti s názvem `uuid.property`.

```
<fte:uuid length="16" property="uuid.property" prefix="ABC"/>
```

Související úlohy

Použití Apache Ant s MFT

fte: vnořený prvek filespec Ant

Parametr **fte:filespec** se používá jako vnořený prvek v jiných úlohách. **fte:filespec** popisuje mapování mezi jedním nebo více zdrojovými soubory, adresáři z/OS nebo datovými sadamia místem určení. Obvykle se tento prvek používá při vyjádření sady souborů nebo adresářů z/OS nebo datových sad k přesunu nebo kopírování.

Vnořeno podle:

- Úloha fte: filecopy
- Úloha fte: filemove

Atributy specifikace zdroje

Musíte zadat jednu ze specifikací srcfilespec nebo srcqueue.

srcfilespec
Uvádí zdroj operace souboru. Hodnota tohoto atributu může obsahovat zástupný znak.

srcqueue
Určuje, že zdrojem přenosu je fronta. Přenos přesune data ze zpráv uložených ve frontě určené tímto atributem. Tento atribut nelze určit, pokud je úloha **fte:filespec** vnořená v úloze **fte:filecopy** .

Atribut srcqueue není podporován, pokud je zdrojový agent agentem mostu protokolů.

Atributy specifikace místa určení

Musíte uvést jeden z dstdir, dstds, dstfilespace, dstfile, dstqueue nebo dstpds.

dstdir
Určuje adresář jako místo určení pro operaci se souborem.

z/OS **dstds**
Určuje datovou sadu jako cíl pro operaci se souborem.

Tento atribut je podporován pouze v případě, že je cílový agent spuštěn na platformě z/OS .

dstfile
Určuje soubor jako cíl pro operaci se souborem.

dstfilespace
Uvádí souborový prostor jako cíl pro operaci se souborem.

Tento atribut platí pouze v případě, že cílový agent je webový agent IBM MQ 8.0 , který má přístup k soubokovému prostoru webové brány.

z/OS **dstpds**
Určuje rozdělenou datovou sadu jako cíl pro operaci se souborem.

Tento atribut je podporován pouze v případě, že je cílový agent spuštěn na platformě z/OS .

dstqueue

Uvádí frontu jako cíl pro operaci souboru se zprávou. Volitelně můžete do této specifikace zahrnout název správce front ve formátu QUEUE@QUEUEMANAGER. Pokud nezadáte název správce front, použije se správce front cílového agenta, pokud jste nenastavili vlastnost výstupního agenta enableClusterQueueInputna hodnotu true. Pokud je vlastnost enableClusterQueueInputOuput nastavena na hodnotu true, cílový agent použije standardní procedury IBM MQ k určení umístění fronty. Je třeba zadat platný název fronty, která existuje ve správci front.

Zadáte-li atribut dstqueue , nemůžete zadat atributy srcqueue , protože se tyto atributy vzájemně vylučují.

Atribut dstqueue není podporován, pokud je cílový agent agentem mostu protokolů.

Atributy volby zdroje

srcencoding
Volitelné. Kódování znakové sady použité souborem pro přenos.

Tento atribut můžete zadat pouze v případě, že je atribut conversion nastaven na hodnotu `text.` .

Pokud nezadáte atribut srcencoding , bude pro přenos textu použita znaková sada zdrojového systému.

srceol
Volitelné. Konec oddělovače řádků používaný přenášeným souborem. Platné hodnoty jsou:

- CRLF -Jako oddělovač řádků použijte znak CR následovaný znakem LF. Tato konvence je typická pro systémy Windows .
- LF -Jako konec oddělovače řádků použijte znak posunu řádku. Tato konvence je typická pro systémy UNIX .

Tento atribut můžete zadat pouze v případě, že je atribut conversion nastaven na hodnotu `text`. Pokud neuvedete atribut srceol , textové přenosy automaticky určí správnou hodnotu na základě operačního systému zdrojového agenta.

z/OS **srckeeptrailingspaces**
Volitelné. Určuje, zda jsou koncové mezery uchovávány ve zdrojových záznamech čtených z datové sady s pevným formátem délky jako součást přenosu v textovém režimu. Platné hodnoty jsou:

- `true` -koncové mezery jsou zachovány.
- `false` -koncové mezery jsou odstraněny.

Pokud nezadáte atribut srckeeptrailingspaces , bude zadána výchozí hodnota `false` .

Tento atribut můžete zadat pouze v případě, že zadáte také atribut srcfilespec a nastavíte atribut conversion na hodnotu `text.` .

srcmsgdelimbytes
Volitelné. Určuje jednu nebo více bajtových hodnot, které se mají vložit jako oddělovač při připojování více zpráv k binárnímu souboru. Každá hodnota musí být uvedena jako dvě hexadecimální číslice v rozsahu 00-FF s předponou x. Více bajtových hodnot je třeba oddělit čárkou. Například `srcmsgdelimbytes="x08,xA4"`. Atribut srcmsgdelimbytes můžete určit pouze v případě, že jste

zadali také atribut srcqueue . Atribut srcmsgdelimbytes nelze určit, pokud jste pro atribut conversion zadali také hodnotu `text` .

srcmsgdelimtext
Volitelné. Určuje posloupnost textu, který má být vložen jako oddělovač při připojování více zpráv k textovému souboru. Do oddělovače můžete zahrnout řídicí posloupnosti Java pro řetězcové literály. Například `srcmsgdelimtext="\u007d\n"`. Oddělovač textu je vložen za každou zprávu zdrojovým agentem. Oddělovač textu je zakódován do binárního formátu pomocí zdrojového kódování přenosu. Každá zpráva je načtena v binárním formátu, kódovaný oddělovač je připojen v binárním formátu ke zprávě a výsledek je přenesen v binárním formátu do cílového agenta. Pokud zdrojová kódová stránka agenta obsahuje stavy shift-in a shift-out, agent předpokládá, že každá zpráva je na konci zprávy ve stavu shift-out. Na cílovém agentovi se binární data převedou stejným způsobem jako přenos souboru do textového souboru. Atribut srcmsgdelimtext můžete zadat pouze v případě, že jste pro atribut conversion zadali také atribut srcqueue a hodnotu `text` .

srcmsgdelvyřazování
Volitelné. Určuje pozici, do které je vložen textový nebo binární oddělovač. Platné hodnoty jsou:

- `prefix` -oddělovače jsou vloženy do cílového souboru před data z každé zprávy.
- `postfix` -oddělovače jsou vloženy do cílového souboru po datech z každé zprávy.

Atribut srcmsgdelimposition můžete určit pouze v případě, že jste zadali také jeden z atributů srcmsgdelimbytes nebo srcmsgdelimtext .

srcmsggroups
Volitelné. Určuje, že zprávy jsou seskupeny podle ID skupiny IBM MQ . První úplná skupina se zapíše do cílového souboru. Není-li tento atribut uveden, všechny zprávy ve zdrojové frontě se zapíší do cílového souboru. Atribut srcmsggroups můžete zadat pouze v případě, že jste zadali také atribut srcqueue .

srcqueuetimeout
Volitelné. Uvádí dobu v sekundách, po kterou se má čekat na splnění jedné z následujících podmínek:

- Pro novou zprávu, která má být zapsána do fronty.
- Pokud byl uveden atribut srcmsggroups , pro úplnou skupinu, která se má zapsat do fronty.

Pokud není žádná z těchto podmínek splněna v době určené hodnotou srcqueuetimeout, zdrojový agent zastaví čtení z fronty a dokončí přenos. Pokud není zadán atribut srcqueuetimeout , zdrojový agent okamžitě zastaví čtení ze zdrojové fronty, pokud je zdrojová fronta prázdná, nebo v případě, že je zadán atribut srcmsggroups , pokud ve frontě není žádná úplná skupina. Atribut srcqueuetimeout můžete zadat pouze v případě, že jste zadali také atribut srcqueue .

Informace o nastavení hodnoty *srcqueuetimeout* naleznete v tématu Pokyny pro určení doby čekání při přenosu zprávy do souboru.

z/OS **srcrecdelimbytes**
Volitelné. Uvádí jednu nebo více bajtových hodnot, které se mají vložit jako oddělovač při připojování více záznamů ze zdrojového souboru orientovaného na záznamy do binárního souboru. Každou hodnotu musíte zadat jako dvě hexadecimální číslice v rozsahu 00-FF s předponou x. Více bajtových hodnot je třeba oddělit čárkou. Příklad:

```
srcrecdelimbytes="x08,xA4"
```

Atribut srcrecdelimbytes můžete určit pouze v případě, že je zdrojový soubor přenosu orientovaný na záznamy, například datová sada z/OS , a cílový soubor je normální, nezáznamově orientovaný soubor. Atribut srcrecdelimbytes nelze určit, pokud jste pro atribut conversion zadali také hodnotu `text` .

srcrecdelimpos
Volitelné. Určuje pozici, do které je vložen binární oddělovač. Platné hodnoty jsou:

- prefix-oddělovače jsou vkládány do cílového souboru před data z každého záznamu souboru orientovaného na zdrojový záznam.

- postfix-oddělovače se vkládají do cílového souboru po datech z každého záznamu souboru orientovaného na zdrojový záznam.

Atribut srcrecdelimpos můžete určit pouze v případě, že jste zadali také atribut srcrecdelimbytes .

Atributy volby cíle

dstencoding
Volitelné. Kódování znakové sady, které má být použito pro přenesený soubor.

Tento atribut můžete zadat pouze v případě, že je atribut conversion nastaven na hodnotu `text.` .

Pokud atribut dstencoding není zadán, bude pro přenosy textu použita znaková sada cílového systému.

dsteol
Volitelné. Konec oddělovače řádků, který má být použit pro přenesený soubor. Platné hodnoty jsou:

- CRLF -Jako oddělovač řádků použijte znak CR následovaný znakem LF. Tato konvence je typická pro systémy Windows .
- LF -Jako konec oddělovače řádků použijte znak posunu řádku. Tato konvence je typická pro systémy UNIX .

Tento atribut můžete zadat pouze v případě, že je atribut conversion nastaven na hodnotu `text.` .

Pokud neuvedete atribut dsteol , textové přenosy automaticky určí správnou hodnotu na základě operačního systému cílového agenta.

dstmsgdelimbytes
Volitelné. Určuje hexadecimální oddělovač, který má být použit při rozdělení binárního souboru do více zpráv. Všechny zprávy mají stejné ID skupiny IBM MQ ; poslední zpráva ve skupině má nastaven příznak IBM MQ LAST_MSG_IN_GROUP. Formát pro určení hexadecimálního bajtu jako oddělovače je `xNN`, kde N je znak v rozsahu 0-9 nebo a-f. Můžete určit posloupnost hexadecimálních bajtů jako oddělovač zadáním seznamu hexadecimálních bajtů oddělených čárkami, například: `x3e,x20,x20,xbf`.

Atribut dstmsgdelimbytes můžete zadat pouze v případě, že jste zadali také atribut dstqueue a přenos je v binárním režimu. Můžete zadat pouze jeden z atributů dstmsgsize, dstmsgdelimbytesa dstmsgdelimpattern .

dstmsgdelimpattern
Volitelné. Určuje regulární výraz Java , který má být použit při rozdělení textového souboru do více zpráv. Všechny zprávy mají stejné ID skupiny IBM MQ ; poslední zpráva ve skupině má nastaven příznak IBM MQ LAST_MSG_IN_GROUP. Formát pro určení regulárního výrazu jako oddělovače je regulární výraz uzavřený v závorkách (*`regular_expression`*) nebo uzavřený v uvozovkách "*`regular_expression`*". Další informace viz Regulární výrazy používané MFT.

Standardně je délka řetězce, se kterým se může regulární výraz shodovat, omezena cílovým agentem na pět znaků. Toto chování můžete změnit pomocí vlastnosti agenta **`maxDelimiterMatchLength`** . Další informace viz Rozšířené vlastnosti agenta MFT.

Atribut dstmsgdelimpattern můžete zadat pouze v případě, že jste zadali také atribut dstqueue a přenos je v textovém režimu. Můžete zadat pouze jeden z atributů dstmsgsize, dstmsgdelimbytesa dstmsgdelimpattern .

dstmsgdelvyřazování
Volitelné. Určuje pozici, ve které je očekáván textový nebo binární oddělovač. Platné hodnoty jsou:

- `prefix` -Oddělovače jsou očekávány na začátku každého řádku.
- `postfix` -Oddělovače jsou očekávány na konci každého řádku.

Atribut dstmsgdelimposition můžete určit pouze v případě, že jste zadali také atribut dstmsgdelimpattern .

dstmsgincludedelim
Volitelné. Uvádí, zda zahrnout oddělovač, který se používá k rozdělení souboru do více zpráv ve zprávách. Je-li uveden atribut dstmsgzahrnutedelim , oddělovač se zahrne na konec zprávy, která obsahuje data souboru předcházející oddělovači. Ve výchozím nastavení není oddělovač ve zprávách zahrnut. Atribut dstmsgincludedelim můžete zadat pouze v případě, že jste zadali také jeden z atributů dstmsgdelimpattern a dstmsgdelimbytes .

dstmsgpersist
Volitelné. Určuje, zda jsou zprávy zapsané do cílové fronty trvalé. Platné hodnoty jsou:

- `true` -Zapsat trvalé zprávy do cílové fronty. Toto je výchozí hodnota.
- `false` -Zapsat dočasné zprávy do cílové fronty.
- `qdef` -Hodnota perzistence je převzata z atributu DefPersistence cílové fronty.

Tento atribut můžete zadat pouze v případě, že je uveden také atribut dstqueue .

dstmsgprops
Volitelné. Určuje, zda první zpráva zapsaná přenosem do cílové fronty má nastaveny vlastnosti zprávy IBM MQ . Možné hodnoty jsou:

- `true` -Nastavení vlastností zprávy pro první zprávu vytvořenou přenosem.
- `false` -Nenastavovat vlastnosti zprávy pro první zprávu vytvořenou přenosem. Toto je výchozí hodnota.

Další informace naleznete v tématu MQ -vlastnosti zpráv nastavené produktem MFT ve zprávách zapisovaných do cílových front.

Tento atribut můžete zadat pouze v případě, že je uveden také atribut dstqueue .

dstmsgsize
Volitelné. Určuje, zda má být soubor rozdělen do více zpráv s pevnou délkou. Všechny zprávy mají stejné ID skupiny IBM MQ ; poslední zpráva ve skupině má nastaven příznak IBM MQ LAST_MSG_IN_GROUP. Velikost zpráv je určena hodnotou dstmsgsize. Formát dstmsgsize je *lengthunits*, kde *length* je kladná celočíselná hodnota a *units* je jedna z následujících hodnot:

- `B` -Bajty. Minimální povolená hodnota je dvojnásobek maximální hodnoty počtu bajtů na znak kódové stránky cílových zpráv.
- `K` -Kibibajty. Jedná se o ekvivalent 1024 bajtů.
- `M` -Mebibytes. Jedná se o ekvivalent 1024 kibibajtů.

Pokud je soubor přenášen v textovém režimu a je ve dvoubajtové znakové sadě nebo ve vícebajtové znakové sadě, je soubor rozdělen do zpráv na nejbližší hranici znaků k zadané velikosti zprávy.

Atribut dstmsgsize můžete zadat pouze v případě, že jste zadali také atribut dstqueue . Můžete zadat pouze jeden z atributů dstmsgsize, dstmsgdelimbytesa dstmsgdelimpattern .

dstunsupportedcodepage
Volitelné. Určuje akci, která má být provedena v případě, že správce cílové fronty určený atributem dstqueue nepodporuje kódovou stránku používanou při přenosu dat souboru do fronty jako textový přenos. Platné hodnoty pro tento atribut jsou následující:

- `binary` -pokračovat v přenosu, ale nepoužít převod kódové stránky na přenášená data. Zadání této hodnoty je ekvivalentní nastavení atributu převodu na hodnotu `text`.
- `fail` -nepokračujte v operaci přenosu. Soubor je zaznamenán jako soubor, jehož přenos se nezdařil. Toto nastavení je výchozí.

Atribut dstunsupportedcodepage můžete zadat pouze v případě, že jste pro atribut conversion zadali také atribut dstqueue a hodnotu `text` .

z/OS **dsttruncaterecords**
Volitelné. Určuje, že cílové záznamy delší než atribut datové sady LRECL jsou oříznuty. Je-li hodnota nastavena na true, záznamy se oříznou. Je-li hodnota nastavena na false, záznamy se zalomí. Výchozí

nastavení je false. Tento parametr je platný pouze pro přenosy v textovém režimu, kde cílem je datová sada.

Ostatní atributy

checksum
Volitelné. Určuje algoritmus použitý pro kontrolní součet přenesených souborů.

- `MD5` -použijte hašovací algoritmus MD5 .
- `NONE` -nepoužívat algoritmus kontrolního součtu.

Pokud neuvedete atribut checksum , použije se výchozí hodnota `MD5` .

konverze
Volitelné. Uvádí typ konverze, která se má použít na soubor při jeho přenosu. Možné hodnoty jsou:

- `binary` -nepoužít žádný převod.
- `text` -použít převod kódové stránky mezi zdrojovým a cílovým systémem. Použijte také převod oddělovačů řádků. Atributy `srcencoding`, `dstencoding`, `srceol` a `dsteol` ovlivňují použitý převod.

Pokud nezadáte atribut conversion , bude zadána výchozí hodnota `binary` .

overwrite
Volitelné. Určuje, zda může operace přepsat existující cílový soubor z/OS nebo datovou sadu .
Zadáte-li hodnotu `true`, budou přepsány všechny existující cílové soubory z/OS nebo datové sady . Zadáte-li hodnotu `false`, existence duplicitního souboru z/OS nebo datové sady v místě určení bude mít za následek selhání operace. Pokud atribut overwrite není zadán, bude zadána výchozí hodnota `false` .

rekurze
Volitelné. Určuje, zda se přenos souborů bude opakovat do podadresářů. Zadáte-li hodnotu `true`, přenos se bude rekurzovat do podadresářů. Zadáte-li hodnotu `false`, přenos se nebude opakovat do podadresářů. Pokud není uveden atribut recurse , je uvedena výchozí hodnota `false` .

Příklad

Tento příklad uvádí fte: filespec se zdrojovým souborem `file1.bin` a cílovým souborem `file2.bin`.

```
<fte:filespec srcfilespec="/home/fteuser/file1.bin" dstfile="/home/fteuser/file2.bin"/>
```

Související úlohy

Použití Apache Ant s MFT

fte: metadata Ant vnořené prvky

Metadata se používají k přenosu dalších informací definovaných uživatelem s operací přenosu souborů.

Další informace o tom, jak produkt Managed File Transfer používá metadata, naleznete v tématu “Metadata pro uživatelské procedury MFT” na stránce 2117 .

Vnořeno podle:

- Úloha fte: filecopy
- Úloha fte: filemove
- Úloha fte: call

Parametry určené jako vnořené prvky

fte: položka
Musíte zadat alespoň jednu položku uvnitř vnořeného prvku fte:metadata . Můžete zvolit zadání více než jedné položky. Položky přidružují název klíče k hodnotě. Klíče musí být jedinečné v bloku fte:metadata

Atributy položky

name
Povinné Název klíče, který patří k této položce. Tento název musí být jedinečný v rámci všech parametrů **entry** vnořených v prvku fte: metadata .

hodnota
Povinné Hodnota, která se má přiřadit k této položce.

Příklad

Tento příklad ukazuje definici fte:metadata , která obsahuje dvě položky.

```
<fte:metadata>
    <fte:entry name="org.foo.partColor" value="red"/>
    <fte:entry name="org.foo.partSize" value="medium"/>
</fte:metadata>
```

Související úlohy
Použití Apache Ant s MFT

Vnořené prvky vyvolání programu

Programy lze spustit pomocí jednoho z pěti vnořených prvků: `fte:presrc`, `fte:predst`, `fte:postdst`, `fte:postsrca` `fte:command`. Tyto vnořené prvky instruují agenta, aby volal externí program jako součást jeho zpracování. Než budete moci spustit program, musíte se ujistit, že je příkaz v umístění uvedeném ve vlastnosti commandPath v souboru `agent.properties` agenta, který spouští příkaz.

I když má každý prvek vyvolání programu jiný název, sdílí stejnou sadu atributů a stejnou sadu vnořených prvků. Programy lze spustit pomocí úloh Ant **fte:filecopy**, **fte:filemove**a **fte:command** .

Programy nelze vyvolat z agenta mostu Connect:Direct .

Úlohy Ant, které mohou vyvolávat programy:

- Úloha fte: filecopy vnořuje parametry vyvolání programu pomocí vnořených prvků `fte:predst`, `fte:postdst`, `fte:presrca` `fte:postsrc` .
- Úloha fte: filemove vnořuje parametry vyvolání programu pomocí vnořených prvků `fte:predst`, `fte:postdst`, `fte:presrca` `fte:postsrc` .
- Úloha fte: call vnořuje parametry vyvolání programu pomocí vnořeného prvku `fte:command` .

Atributy

příkaz
Povinné Pojmenuje program, který se má volat. Aby mohl agent spustit příkaz, musí být příkaz v umístění určeném vlastností commandPath v souboru `agent.properties` agenta. Další informace viz commandPath MFT vlastnost. Veškeré informace o cestě uvedené v atributu `command` jsou považovány za relativní vzhledem k umístění určenému vlastností commandPath . Je-li hodnota `type` `executable`, je očekáván spustitelný program, jinak je očekáván skript odpovídající typu volání.

počet opakování
Volitelné. Počet opakování volání programu, pokud program nevrátí návratový kód úspěchu. Program pojmenovaný atributem `command` je volán až do tohoto počtu. Hodnota přiřazená k tomuto atributu musí být nezáporná. Pokud neuvedete atribut `retrycount` , použije se výchozí hodnota nula.

retrywait-čekání
Volitelné. Doba čekání, v sekundách, před dalším pokusem o vyvolání programu. Pokud program pojmenovaný atributem `command` nevrátí návratový kód úspěchu a atribut `retrycount` uvádí nenulovou hodnotu, tento parametr určuje dobu čekání mezi opakovanými pokusy. Hodnota přiřazená k tomuto atributu musí být nezáporná. Pokud neuvedete atribut `retrywait` , použije se výchozí hodnota nula.

successrc (úspěšně)
Volitelné. Hodnota tohoto atributu se používá k určení, kdy se úspěšně spustí vyvolání programu. Návratový kód procesu pro příkaz je vyhodnocen pomocí tohoto výrazu. Hodnota se může skládat z jednoho nebo více výrazů kombinovaných se znakem svislé čáry (|), který označuje logickou hodnotu OR, nebo ampersand (&). znak pro označení logického operátoru AND. Každý výraz může být jedním z následujících typů výrazů:

- Číslo, které označuje test rovnosti mezi návratovým kódem procesu a číslem.
- Číslo s předponou ">" označující test větší než mezi číslem a návratovým kódem procesu.
- Číslo s předponou "<" , které označuje test menší než mezi číslem a návratovým kódem procesu.
- Číslo s předponou "!" pro označení testu, který se nerovná hodnotě mezi číslem a návratovým kódem procesu.

Například: >2&<7&!5|0|14 je interpretováno jako úspěšné následující návratové kódy: 0, 3, 4, 6, 14. Všechny ostatní návratové kódy jsou interpretovány jako neúspěšné. Pokud neuvedete atribut `successrc` , použije se výchozí hodnota nula. To znamená, že příkaz je považován za úspěšně spuštěný, pokud, a pouze pokud, vrátí kód nula.

typ
Volitelné. Hodnota tohoto atributu uvádí, jaký typ programu se volá. Zadejte jednu z následujících voleb:

Spustitelné
Úloha volá spustitelný program. Může mít další argumenty určené pomocí vnořeného prvku arg . Očekává se, že program bude přístupný na commandPath a kde je to možné, bude mít nastaveno oprávnění k provádění. Skripty UNIX lze volat tak dlouho, dokud určují program shellu (například první řádek skriptového souboru shellu je: `#!/bin/sh`). Výstup příkazu zapsaný do stderr nebo stdout je odeslán do protokolu Managed File Transfer pro volání. Avšak množství datového výstupu je omezeno konfigurací agenta. Výchozí hodnota je 10K bajtů dat, ale tuto výchozí hodnotu můžete přepsat pomocí vlastnosti agenta: maxCommandOutput.

antscript
Úloha spustí uvedený skript Ant pomocí příkazu **`fteAnt`** . Vlastnosti lze určit pomocí vnořeného prvku property . Cíle Ant lze určit pomocí vnořeného prvku target . Očekává se, že skript Ant bude přístupný na commandPath. Výstup Ant zapsaný do stderr nebo stdout je odeslán do protokolu Managed File Transfer pro volání. Avšak množství datového výstupu je omezeno konfigurací agenta. Výchozí hodnota je 10K bajtů dat, ale tuto výchozí hodnotu můžete přepsat pomocí vlastnosti agenta: maxCommandOutput.

z/OS **JCL**
Hodnota `jcl` je podporována pouze v systému z/OS a spouští určený skript z/OS JCL. JCL je odesláno jako úloha a vyžaduje, aby byl přítomen zakázkový list. Když je úloha úspěšně odeslána, výstup příkazu JCL, zapsaný do protokolu Managed File Transfer , obsahuje následující text: JOB *job_name*(*job_id*), kde:

- *job_name* je název úlohy identifikovaný zakázkový list v JCL.
- *ID_úlohy* je ID úlohy generované systémem z/OS .

Pokud úlohu nelze úspěšně odeslat, příkaz skriptu JCL selže a zapíše zprávu do protokolu označující příčinu selhání (například není přítomen žádný zakázkový list). Chcete-li zjistit, zda byla úloha úspěšně spuštěna nebo dokončena, použijte systémovou službu, například SDSF. Managed File Transfer neposkytuje informace, protože pouze zadává úlohu; systém pak určí, kdy se má úloha spustit a jak se má prezentovat výstup úlohy. Protože je skript JCL odeslán jako dávková úloha, není vhodné zadat `jcl` pro vnořený prvek `presrc` nebo `predst` , protože víte pouze, že

úloha byla úspěšně odeslána, a nikoli, zda byla úspěšně dokončena před spuštěním přenosu. Neexistují žádné vnořené prvky, které jsou platné s typem `jcl`.

Následující příklad ukazuje úlohu JCL:

```
//MYJOB JOB
//*
//MYJOB EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=H
//SYSUT1 DD DSN=FRED.DEMO.TXT,DISP=SHR
//SYSUT2 DD DSN=BOB.DEMO.TXT,DISP=(NEW,CATLG),
// RECFM=VB,LRECL=133,BLKSIZE=2048,
// SPACE=(TRK,(30,5),RLSE)
//SYSIN DD DUMMY
```

Parametry určené jako vnořené prvky

fte: arg
Platné pouze v případě, že hodnota atributu `type` je spustitelný. Pomocí vnořených prvků `fte:arg` určete argumenty programu, který je volán jako součást vyvolání programu. Argumenty programu jsou sestaveny z hodnot určených prvky `fte:arg` v pořadí, ve kterém jsou rozpoznány prvky `fte:arg` . Můžete se rozhodnout, že jako vnořené prvky vyvolání programu zadáte nula nebo více prvků `fte:arg` .

fte: zařízení
Platné pouze v případě, že hodnota atributu `type` je antscript. Použijte atributy `name` a `value` vnořených prvků `fte:property` k předání dvojic název-hodnota do skriptu Ant . Můžete se rozhodnout, že jako vnořené prvky vyvolání programu zadáte nula nebo více prvků `fte:property` .

fte: cíl
Platné pouze v případě, že hodnota atributu `type` je antscript. Uveďte cíl ve skriptu Ant , který se má volat. Můžete se rozhodnout, že jako vnořené prvky vyvolání programu zadáte nula nebo více prvků `fte:target` .

Atributy arg

hodnota
Povinné Hodnota argumentu, která má být předána volanému programu.

Atributy vlastností

Název
Povinné Název vlastnosti, která se má předat skriptu Ant .

hodnota
Povinné Hodnota, která se má přidružit k názvu vlastnosti předávané skriptu Ant .

Příklady

Tento příklad ukazuje vyvolání programu `fte:postsrc` , které je uvedeno jako součást úlohy fte: filecopy . Vyvolání programu je pro program s názvem `post.sh` a dodává se s jedním argumentem `/home/fteuser2/file.bin.` .

```
<fte:filecopy
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
 src="agent1@qm1" dst="agent2@qm2"
 rcproperty="copy.result">
<fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>
<fte:postsrc command="post.sh" successrc="1" >
    <fte:arg value="/home/fteuser2/file.bin"/>
</fte:postsrc>
</fte:filecopy>
```

Tento příklad ukazuje vyvolání programu `fte:command` , které je uvedeno jako součást úlohy fte: call . Vyvolání programu je pro spustitelný soubor s názvem `command.sh`, kterému nejsou předány žádné argumenty příkazového řádku. Pokud `command.sh` nevrátí návratový kód úspěchu 1, příkaz se zopakuje po 30 sekundách.

```
<fte:call
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
 agent="agent1@qm1"
 rcproperty="call.rc"
 origuser="bob"
 jobname="${job.id}">
<fte:command command="command.sh" successrc="1" retrycount="5" retrywait="30"/>
</fte:call>
```

Tento příklad ukazuje vyvolání programu `fte:command` , které je uvedeno jako součást úlohy fte: call . Vyvolání programu je pro cíl kopírování a komprese ve skriptu Ant s názvem `script.xml`, kterému jsou předány dvě vlastnosti.

```
<fte:call
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
 agent="agent1@qm1"
 rcproperty="call.rc"
 origuser="bob"
 jobname="${job.id}">
<fte:command command="script.xml" type="antscript">
  <property name="src" value="AGENT5@QM5"/>
  <property name="dst" value="AGENT3@QM3"/>
  <target name="copy"/>
  <target name="compress"/>
</fte:command>
</fte:call>
```

Související úlohy

Určení programů, které se mají spustit s MFT

Použití Apache Ant s MFT

Uživatelské procedury MFT pro odkaz na přizpůsobení

Referenční informace, které vám pomohou konfigurovat uživatelské procedury pro Managed File Transfer.

Související pojmy

MFT zdrojové a cílové uživatelské procedury

Metadata pro uživatelské procedury MFT

Existují tři různé typy metadat, které lze dodat do uživatelských procedur pro produkt Managed File Transfer: prostředí, přenos a metadata souboru. Tato metadata jsou prezentována jako mapy dvojic klíč-hodnota Java .

Metadata prostředí

Metadata prostředí jsou předána všem uživatelským proceduře a popisují běhové prostředí agenta, ze kterého je volána uživatelská procedura. Tato metadata jsou jen pro čtení a nelze je aktualizovat žádnou uživatelskou procedurou.

Tabulka 882. Metadata prostředí

Klíč	**Popis**
KLÍČ_ADRESÁŘE_KONFIGURACE_AGENTA	Název adresáře, který obsahuje informace o konfiguraci agenta.
AGENT_PRODUCT_DIRECTORY_KEY	Název adresáře, ve kterém byl nainstalován kód agenta.

Tabulka 882. Metadata prostředí (pokračování)	
Klíč	**Popis**
AGENT_VERSION_KEY	Číslo verze běhového prostředí agenta, které volá uživatelskou proceduru.

Názvy klíčů a názvy hodnot uvedené v tabulce 1 jsou konstanty, které jsou definovány v rozhraní EnvironmentMetaDataConstants .

Metadata přenosu

Metadata přenosu jsou předána všem uživatelským proceduře. Metadata se skládají z hodnot dodaných systémem a hodnot dodaných uživatelem. Změníte-li hodnoty dodané systémem, budou tyto změny ignorovány. Počáteční uživatelem zadané hodnoty pro uživatelskou proceduru spuštění přenosu zdroje jsou založeny na hodnotách, které zadáte při definování přenosu. Zdrojový agent může změnit hodnoty dodané uživatelem jako součást zpracování uživatelské procedury spuštění přenosu zdroje. Tato uživatelská procedura je volána před zahájením celého přenosu souborů. Tyto změny se používají v následných voláních jiných uživatelských procedur, které se vztahují k tomuto přenosu. Metadata přenosu se použijí na celý přenos.

Ačkoli všechny uživatelské procedury mohou číst hodnoty z metadat přenosu, pouze uživatelská procedura spuštění přenosu zdroje může změnit metadata přenosu.

Metadata přenosu nelze použít k šíření informací mezi různými přenosy souborů.

Systémem dodávaná metadata přenosu jsou podrobně popsána v tabulce 2:

Tabulka 883. Metadata přenosu	
Klíč	**Popis**
KLÍČ_CÍLOVÉHO_AGENTA	Název agenta, který je cílem přenosu.
Klíč JOB_NAME_KEY	Název úlohy přidružené k požadavku na přenos
MQMD_USER_KEY	Pole uživatele MQMD ze zprávy použité k odeslání požadavku na přenos
KLÍČ_HOSTITELE_ORIGINÁLU	Název hostitele zadaný jako původní název hostitele v požadavku na přenos
KLÍČ_UŽIVATELE_ORIGINÁLU	Jméno uživatele zadané jako ID původního uživatele v požadavku na přenos
ZDROJOVÝ_KLÍČ	Název agenta, který je zdrojem přenosu
KLÍČ_ID_TRANSFERU	Identifikátor přenosu

Názvy klíčů a názvy hodnot uvedené v tabulce 2 jsou konstanty, které jsou definovány v rozhraní TransferMetaDataConstants .

Metadata souboru

Metadata souboru jsou předána uživatelské proceduře zahájení přenosu zdroje jako součást specifikace souboru. Pro zdrojový a cílový soubor existují oddělená metadata souboru.

Metadata souboru nelze použít k šíření informací mezi různými přenosy souborů.

Tabulka 884. Metadata souboru

Klíč	Povolené hodnoty	Popis
Oddělovače CONVERT_LINE_SEPARATORS		Hodnota klíče použitá pro textové přenosy k určení, zda jsou posloupnosti oddělovače řádků CRLF (znak CR-LF) nebo LF (znak LF) ve zdrojových datech převedeny na posloupnost oddělovačů řádků v místě určení.
KLÍČ_ODDĚLOVAČE		Hodnota klíče použitá k definování oddělovače pro oddělení dat záznamu při přenosu dat orientovaných na záznamy do normálních souborů. Používá se také pro přenos zpráv do souborů a souborů do zpráv.
VYMEZIT_KLÍČ_POZICE	ODDĚLOVAČ_POZIC_HODNOTY_PŘEDPOKLAD_HODNOTY ODDĚLOVAČ_POZIC_POSTFIX_HODNOTA	Použijte s DELIMITER_KEY k definování pozice oddělovače; předpona nebo následná oprava.
KLÍČ typu DELIMITER_TYPE_KEY	Hodnota BINARY_DELIMITER_TYPE_BINARY_VALUE HODNOTY_TEXTOVÉHO_TYPU_VYMEZENÍ Hodnota DELIMITER_TYPE_SIZE_VALUE	Pomocí parametru DELIMITER_KEY můžete definovat typ oddělovače.
CÍLOVÝ_EXIST_KLÍČ	HODNOTA_CHYBY_KLÍČ_EXISTUJE_KLÍČ_HODNOTA_EXISTUJE_KLÍČ_PŘEPIS_HODNOTA	Určuje chování přenosu souborů, pokud cílový soubor existuje.
FILE_ALIAS_KEY		Hodnota klíče použitá k definování aliasu pro přenášený soubor.
KLÍČ_METODY_KONTROLNÍHO součtu	FILE_CHECKSUM_METHOD_NONE_VALUE FILE_CHECKSUM_METHOD_MD5_VALUE	Určuje metodu kontrolního součtu, která se má použít při přenosu souboru.
FILE_CONVERSION_KEY	HODNOCENÍ_TEXT_KONVERZE_SOUBORU_KONVERZE_BINÁRNÍ_HODNOTA	Určuje typ převodu použitý na obsah souboru.
FILE_ENCODING_KEY		Určuje kódování použité pro textový soubor.
FILE_END_OF_LINE_KEY	KONCOVÁ_SOUBOR_ŘÁDEK_HODNOTA_LF_HODNOTA_SOUBORU_KONCE_ŘÁDEK_ŘÁDEK_HODNOTY	Určuje posloupnost znaků, která označuje konec řádku: < LF > nebo < CR> < LF>.
FILE_SPACE_ALIAS		Určuje alias souboru v souborovém prostoru. **Poznámka:** Tato metadata lze použít pouze v případě, že parametr FILE_TYPE_KEY má hodnotu FILE_TYPE_FILE_SPACE_VALUE.
FILE_SPACE_NAME		Určuje název souborového prostoru. **Poznámka:** Tato metadata lze použít pouze v případě, že parametr FILE_TYPE_KEY má hodnotu FILE_TYPE_FILE_SPACE_VALUE.
FILE_TYPE_KEY	TYP_SOUBORU_SOUBORU_HODNOTA TYP_SOUBORU_SOUBORU_HODNOTA TYP_SOUBORU_DATASET_HODNOTA TYP_SOUBORU_SOUBORU_PDS_HODNOTA TYP_SOUBORU_VE_SOUBORU_SOUBORU_HODNOTA	Určuje cílový soubor, frontu nebo specifikaci souborového prostoru.

Tabulka 884. Metadata souboru (pokračování)

Klíč	Povolené hodnoty	Popis
SKUPINOVÝ_ID_KLÍČ		Hodnota klíče použitá pro přenos zpráv do souborů k určení skupiny zpráv, které se mají číst ze zdrojové fronty. Tento atribut je platný pouze v případě, že hodnota parametru USE_GROUPS_KEY je USE_GROUPS_TRUE_VALUE.
KLÍČ_ZPRÁVY_VYMEZENÍ_IN_ZPRÁVY	ZAHRNOUT_ODDĚLOVITER_IN_MESSAGE_TRUE_VALUE ZAHRNOUT_ODDĚLOVAČ_IN_ZPRÁVA_FALSE_VALUE	Hodnota klíče použitá pro přenosy souborů do zpráv k určení, zda zahrnout oddělovače, které byly použity k rozdělení souboru do více zpráv na konci zpráv. Tento atribut je platný pouze v případě, že hodnota parametru DELIMITER_TYPE_KEY je DELIMITER_TYPE_BINARY_VALUE DELIMITER_TYPE_TEXT_VALUE.
VLOŽIT_ODDĚLOVAČ_ŘÁDEK_ŘÁDKU		Hodnota klíče použitá pro textové přenosy ze souborů orientovaných na záznamy k určení, zda jsou oddělovače řádků vloženy do dat po každém záznamu.
KEEP_TRAILING_SPACES_KEY	KEEP_TRAILING_SPACES_TRUE_VALUE KEEP_TRAILING_SPACES_FALSE_VALUE	Hodnota klíče použitá k určení, zda jsou koncové mezery odebrány ze záznamů čtených z datových sad formátu pevné délky.
NEW_RECORD_ON_LINE_SEPARATOR_KEY		Hodnota klíče použitá pro textové přenosy do záznamově orientovaných souborů, která uvádí, zda se oddělovače řádků v datech zahrnou do dat záznamu nebo způsobí nový záznam (a nezapisují se).
TRVALÝ_KLÍČ	PERSISTENT_TRUE_VALUE Hodnota PERSISTENT_FALSE_VALUE Hodnota PERSISTENT_QDEF_VALUE	Hodnota klíče použitá pro přenos souborů do zpráv k určení, zda jsou zprávy trvalé.
SET_MQ_PROPS_KEY	SET_MQ_PROPS_TRUE_VALUE SET_MQ_PROPS_FALSE_VALUE	Hodnota klíče použitá pro přenosy souborů do zpráv k určení, zda jsou vlastnosti zprávy IBM MQ nastaveny na první zprávu v souboru a na všechny zprávy zapsané do fronty, když dojde k chybě.
NEROZPOZNÁ_KLÍČ_STRÁNKY_KÓDU	NEROZPOZNÁNA_KÓD_KÓD_STRÁNKY_SELHÁNÍ_HODNOTA NEROZPOZNÁNA_KÓD_KÓD_STRÁNKY_BINÁRNÍ_HODNOTA	Hodnota klíče použitá pro přenos souborů do zpráv k určení, zda se přenos v textovém režimu nezdaří nebo zda se provede převod, pokud není kódová stránka dat rozpoznána správcem cílové fronty.
KLÍČ_SKUPINY_POUŽITÍ	USE_GROUPS_TRUE_VALUE USE_GROUPS_FALSE_VALUE	Hodnota klíče použitá pro přenos zpráv do souborů k určení, zda se má přenést pouze úplná skupina zpráv ze zdrojové fronty.

Tabulka 884. Metadata souboru (pokračování)		
Klíč	**Povolené hodnoty**	**Popis**
WAIT_TIME_KEY		Hodnota klíče použitá pro přenos zpráv do souborů k určení doby v sekundách, po kterou má zdrojový agent čekat na jeden z následujících případů: • Zpráva, která se objeví ve zdrojové frontě, pokud je fronta prázdná nebo se stala prázdnou, pokud je hodnota USE_GROUPS_KEY FALSE. • Úplná skupina, která se objeví ve zdrojové frontě, pokud je hodnota USE_GROUPS_KEY TRUE.

Názvy klíčů a názvy hodnot uvedené v tabulce 3 jsou konstanty, které jsou definovány v rozhraní FileMetaDataConstants .

Související pojmy

“Java rozhraní pro uživatelské procedury MFT” na stránce 2127
Referenční informace o rozhraních Java pro uživatelské procedury naleznete v tématech v této části.

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“Uživatelské procedury monitoru prostředků MFT” na stránce 2121
Uživatelské procedury monitoru prostředků vám umožňují konfigurovat vlastní kód pro spuštění, když je splněna podmínka spouštěče monitoru, před spuštěním přidružené úlohy.

“MFT Vlastnosti agenta pro uživatelské procedury” na stránce 2125
Kromě standardních vlastností v souboru `agent.properties` existuje několik rozšířených vlastností speciálně pro uživatelské procedury. Tyto vlastnosti nejsou standardně zahrnuty, takže pokud je chcete použít, musíte ručně upravit soubor `agent.properties` . Pokud provedete změnu v souboru `agent.properties` , zatímco je agent spuštěn, zastavte a restartujte agenta, aby se změny projevily.

Uživatelské procedury monitoru prostředků MFT

Uživatelské procedury monitoru prostředků vám umožňují konfigurovat vlastní kód pro spuštění, když je splněna podmínka spouštěče monitoru, před spuštěním přidružené úlohy.

Nedoporučuje se vyvolávat nové přenosy přímo z kódu uživatelské procedury. V některých případech to způsobí, že soubory budou přeneseny vícekrát, protože uživatelské procedury nejsou schopné obnovy agenta.

Uživatelské procedury monitoru prostředků používají existující infrastrukturu pro uživatelské procedury. Uživatelské procedury monitoru jsou volány po spuštění monitoru, ale před spuštěním odpovídající úlohy úlohou monitoru. To umožňuje uživatelské proceduře upravit úlohu, která má být spuštěna, a rozhodnout, zda má úloha pokračovat či nikoli. Úlohu monitorování můžete upravit tak, že aktualizujete metadata monitoru, která se pak použijí pro substituci proměnných v dokumentu úlohy vytvořeném vytvořením původního monitoru. Volitelně může uživatelská procedura monitoru nahradit nebo aktualizovat řetězec XML definice úlohy předaný jako parametr. Uživatelská procedura monitoru může pro úlohu vrátit výsledný kód buď 'pokračovat', nebo 'zrušit'. Pokud je vráceno zrušení, úloha se nespustí a monitor se znovu nespustí, dokud monitorovaný prostředek nebude odpovídat podmínkám spouštěče. Pokud se prostředek nezměnil, spouštěč se nespustí. Stejně jako u ostatních uživatelských procedur můžete monitorovat řetězce uživatelských procedur společně. Pokud jedna z uživatelských procedur vrátí kód výsledku zrušení, celkový výsledek se zruší a úloha se nespustí.

- Mapa metadat prostředí (stejná jako ostatní uživatelské procedury)
- Mapa metadat monitoru včetně neměnných metadat systému a měnitelných metadat uživatele. Neměnná systémová metadata jsou následující:
  - FILENAME-název souboru, který splnil podmínku spouštěče
  - FILEPATH-cesta k souboru, který splnil podmínku spouštěče
  - FILESIZE (v bajtech-tato metadata nemusí být přítomna)-velikost souboru, který splňuje podmínku spouštěče
  - LASTMODIFIEDDATE (Local)-datum poslední změny souboru, který splnil podmínku spouštěče. Tento datum se vyjadřuje jako místní datum časového pásma, ve kterém je agent spuštěný, a je naformátovaný na datum dle normy ISO 8601.
  - LASTMODIFIEDTIME (Local)-čas v lokálním formátu, kdy byl soubor, který splnil podmínku spouštěče, naposledy změněn. Tento čas se vyjadřuje jako místní čas časového pásma, ve kterém je agent spuštěný, a je naformátovaný na čas dle normy ISO 8601.
  - LASTMODIFIEDDATEUTC-date v univerzálním formátu, že soubor, který splnil podmínku spouštěče, byl naposledy změněn. Tento datum se vyjadřuje jako místní datum převedený do časového pásma UTC a je naformátovaný na datum podle normy ISO 8601.
  - LASTMODIFIEDTIMEUTC-time v univerzálním formátu, že soubor, který splnil podmínku spouštěče, byl naposledy změněn. Tento čas se vyjadřuje jako místní čas převedený do časového pásma UTC a je naformátovaný na čas podle normy ISO 8601.
  - AGENTNAME-název agenta monitorování
- Řetězec XML představující úlohu, která má být spuštěna jako výsledek spouštěče monitoru.

Uživatelské procedury monitoru vracejí následující data:

- Indikátor, který určuje, zda má pokračovat dále (pokračovat nebo zrušit)
- Řetězec, který má být vložen do zprávy protokolu vyhovující spouštěči.

V důsledku spuštění kódu uživatelské procedury monitoru mohla být aktualizována také metadata monitoru a řetězec XML definice úlohy, které byly původně předány jako parametry.

Hodnota vlastnosti agenta monitorExitTřídy (v souboru `agent.properties`) určuje, které třídy uživatelských procedur monitoru se mají načíst, přičemž jednotlivé třídy uživatelských procedur jsou odděleny čárkou. Příklad:

```
monitorExitClasses=testExits.TestExit1,testExits.testExit2
```

Rozhraní uživatelské procedury monitoru je:

```
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing.  This interface defines a method that will be
 * invoked immediately prior to starting a task as the result of a monitor trigger
 */
public interface MonitorExit {

    /**
     * Invoked immediately prior to starting a task as the result of a monitor
     * trigger.
     *
     * @param environmentMetaData
     *            meta data about the environment in which the implementation
     *            of this method is running.  This information can only be read,
     *            it cannot be updated by the implementation.  The constant
     *            defined in <code>EnvironmentMetaDataConstants</code> class can
```

```
     *            be used to access the data held by this map.
     *
     * @param monitorMetaData
     *            meta data to associate with the monitor. The meta data passed
     *            to this method can be altered, and the changes will be
     *            reflected in subsequent exit routine invocations. This map
     *            also contains keys with IBM reserved names.  These entries are
     *            defined in the <code>MonitorMetaDataConstants</code> class and
     *            have special semantics. The the values of the IBM reserved names
     *            cannot be modified by the exit
     *
     * @param taskDetails
     *            An XML String representing the task to be executed as a result of
     *            the monitor triggering. This XML string may be modified by the
     *            exit
     *
     * @return    a monitor exit result object which is used to determine if the
     *            task should proceed, or be cancelled.
     */
    MonitorExitResult onMonitor(Map<String, String> environmentMetaData,
                                Map<String, String> monitorMetaData,
                                Reference<String> taskDetails);
}
```

Konstanty pro hodnoty IBM-reserved v metadatech monitoru jsou následující:

```
package com.ibm.wmqfte.exitroutine.api;

/**
* Constants for IBM reserved values placed into the monitor meta data
* maps used by the monitor exit routines.
*/
public interface MonitorMetaDataConstants {

/**
* The value associated with this key is the name of the trigger
* file associated with the monitor. Any modification performed
* to this property by user exit routines will be ignored.
*/
final String FILE_NAME_KEY = "FILENAME";

/**
* The value associated with this key is the path to the trigger
* file associated with the monitor. Any modification performed
* to this property by user exit routines will be ignored.
*/
final String FILE_PATH_KEY = "FILEPATH";

/**
* The value associated with this key is the size of the trigger
* file associated with the monitor. This will not be present in
* the cases where the size cannot be determined. Any modification
* performed to this property by user exit routines will be ignored.
*/
final String FILE_SIZE_KEY = "FILESIZE";

/**
* The value associated with this key is the local date on which
* the trigger file associated with the monitor was last modified.
* Any modification performed to this property by user exit routines
* will be ignored.
*/
final String LAST_MODIFIED_DATE_KEY = "LASTMODIFIEDDATE";

/**
* The value associated with this key is the local time at which
* the trigger file associated with the monitor was last modified.
* Any modification performed to this property by user exit routines
* will be ignored.
*/
final String LAST_MODIFIED_TIME_KEY = "LASTMODIFIEDTIME";

/**
* The value associated with this key is the UTC date on which
* the trigger file associated with the monitor was last modified.
* Any modification performed to this property by user exit routines
* will be ignored.
*/
```

```
final String LAST_MODIFIED_DATE_KEY_UTC = "LASTMODIFIEDDATEUTC";

/**
* The value associated with this key is the UTC time at which
* the trigger file associated with the monitor was last modified.
* Any modification performed to this property by user exit routines
* will be ignored.
*/
final String LAST_MODIFIED_TIME_KEY_UTC = "LASTMODIFIEDTIMEUTC";

/**
* The value associated with this key is the name of the agent on which
* the monitor is running. Any modification performed to this property by
* user exit routines will be ignored.
*/
final String MONITOR_AGENT_KEY = "AGENTNAME";

}
```

Příklad uživatelské procedury monitoru

Tato ukázková třída implementuje rozhraní MonitorExit . Tento příklad přidá do metadat monitoru vlastní substituční proměnnou s názvem *REDIRECTEDAGENT* , která bude naplněna hodnotou LONDON , pokud je hodina dne lichá, a hodnotou PARIS pro sudé hodiny. Výsledný kód ukončení monitoru je nastaven tak, aby vždy vracel hodnotu proceed.

```
package com.ibm.wmqfte.monitor;

import java.util.Calendar;
import java.util.Map;

import com.ibm.wmqfte.exitroutine.api.MonitorExit;
import com.ibm.wmqfte.exitroutine.api.MonitorExitResult;
import com.ibm.wmqfte.exitroutine.api.Reference;

/**
 * Example resource monitor user exit that changes the monitor mutable
 * metadata value between 'LONDON' and 'PARIS' depending on the hour of the day.
 *
 */
public class TestMonitorExit implements MonitorExit {

    // custom variable that will substitute destination agent
    final static String REDIRECTED_AGENT = "REDIRECTEDAGENT";

    public MonitorExitResult onMonitor(
Map<String, String> environmentMetaData,
            Map<String, String> monitorMetaData,
Reference<String> taskDetails) {

        // always succeed
        final MonitorExitResult result = MonitorExitResult.PROCEED_RESULT;

        final int hour = Calendar.getInstance().get(Calendar.HOUR_OF_DAY);

        if (hour%2 == 1) {
            monitorMetaData.put(REDIRECTED_AGENT, "LONDON");
        } else {
            monitorMetaData.put(REDIRECTED_AGENT, "PARIS");
        }

        return result;
    }
}
```

Odpovídající úloha pro monitor, který používá substituční proměnnou *REDIRECTEDAGENT* , může vypadat přibližně takto:

```
<?xml version="1.0" encoding="UTF-8"?>
<request version="4.00"
         xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
         xsi:noNamespaceSchemaLocation="FileTransfer.xsd">
  <managedTransfer>
    <originator>
        <hostName>reportserver.com</hostName>
```

```
        <userID>USER1</userID>
    </originator>
    <sourceAgent agent="AGENT1"
                 QMgr="QM1"/>
    <destinationAgent agent="${REDIRECTEDAGENT}"
                      QMgr="QM2"/>
    <transferSet>
      <item mode="binary" checksumMethod="MD5">
        <source recursive="false" disposition="delete">
          <file>c:\sourcefiles\reports.doc</file>
        </source>
        <destination type="file" exist="overwrite">
          <file>c:\destinationfiles\reports.doc</file>
        </destination>
      </item>
    </transferSet>
  </managedTransfer>
</request>
```

Před spuštěním tohoto přenosu se hodnota atributu `agent` prvku <destinationAgent> nahradí hodnotou `LONDON` nebo `PARIS`.

Musíte zadat substituční proměnnou ve třídě uživatelské procedury monitoru a XML definice úlohy velkými písmeny.

Související pojmy

Přizpůsobení produktu MFT pomocí uživatelských procedur

“Metadata pro uživatelské procedury MFT” na stránce 2117

Existují tři různé typy metadat, které lze dodat do uživatelských procedur pro produkt Managed File Transfer: prostředí, přenos a metadata souboru. Tato metadata jsou prezentována jako mapy dvojic klíč-hodnota Java .

“Java rozhraní pro uživatelské procedury MFT” na stránce 2127

Referenční informace o rozhraních Java pro uživatelské procedury naleznete v tématech v této části.

MFT zdrojové a cílové uživatelské procedury

Související odkazy

“MFT Vlastnosti agenta pro uživatelské procedury” na stránce 2125

Kromě standardních vlastností v souboru `agent.properties` existuje několik rozšířených vlastností speciálně pro uživatelské procedury. Tyto vlastnosti nejsou standardně zahrnuty, takže pokud je chcete použít, musíte ručně upravit soubor `agent.properties` . Pokud provedete změnu v souboru `agent.properties` , zatímco je agent spuštěn, zastavte a restartujte agenta, aby se změny projevily.

MFT Vlastnosti agenta pro uživatelské procedury

Kromě standardních vlastností v souboru `agent.properties` existuje několik rozšířených vlastností speciálně pro uživatelské procedury. Tyto vlastnosti nejsou standardně zahrnuty, takže pokud je chcete použít, musíte ručně upravit soubor `agent.properties` . Pokud provedete změnu v souboru `agent.properties` , zatímco je agent spuštěn, zastavte a restartujte agenta, aby se změny projevily.

Proměnné prostředí lze použít v některých vlastnostech Managed File Transfer , které představují umístění souborů nebo adresářů. To umožňuje, aby se umístění souborů nebo adresářů používaných při spouštění částí produktu lišila v závislosti na změnách prostředí, například na tom, který uživatel spouští proces. Další informace viz Proměnné prostředí ve vlastnostech MFT.

Vlastnosti uživatelské procedury

Uživatelské procedury jsou volány v pořadí uvedeném v následující tabulce. Další informace o souboru `agent.properties` naleznete v tématu Rozšířené vlastnosti agenta: Uživatelská procedura.

Tabulka 885. Vlastnosti agenta pro uživatelské procedury

Název vlastnosti	Popis
sourceTransferEndExitTřídy	Určuje seznam tříd, které implementují uživatelskou proceduru ukončení přenosu zdroje, oddělených čárkami.

Tabulka 885. Vlastnosti agenta pro uživatelské procedury (pokračování)	
Název vlastnosti	**Popis**
sourceTransferStartExitTřídy	Určuje seznam tříd oddělených čárkami, které implementují uživatelskou proceduru spuštění přenosu zdroje.
destinationTransferStartExitTřídy	Určuje seznam tříd oddělených čárkami, které implementují uživatelskou proceduru spuštění přenosu cíle.
destinationTransferEndExitTřídy	Určuje seznam tříd oddělených čárkami, které implementují cílovou uživatelskou proceduru přenosu.
exitClassCesta	Určuje seznam adresářů specifických pro konkrétní platformu, oddělených znaky, které slouží jako cesta ke třídám pro uživatelské procedury. Adresář uživatelských procedur agenta se prohledává před všemi položkami v této cestě ke třídám. Závorky, čárky (,) a zpětná lomítka (\) jsou speciální znaky v příkazech MFT a musí být uvozeny zpětným lomítkem (\). Windows Cesty k souborům v systému Windows lze zadat buď pomocí dvojitých zpětných lomítek (\\) jako oddělovače, nebo pomocí jednoduchých dopředných lomítek (/). Například: `exitClassPath=C:/mycomp/mqft/exits/encryptFileExit.jar; C:/mycomp/mqft/exits/fileFilter.jar.` Hodnota této vlastnosti může obsahovat proměnné prostředí.
exitNativeLibraryPath	Určuje seznam adresářů, které se chovají jako cesta k nativní knihovně pro uživatelské procedury, oddělených znaky pro konkrétní platformu. Hodnota této vlastnosti může obsahovat proměnné prostředí.
monitorExitClasses	Určuje seznam tříd oddělený čárkami, který implementuje uživatelskou proceduru monitoru. Další informace viz Uživatelské procedury monitoru prostředků produktu MFT.
protocolBridgeCredentialExitClasses	Určuje seznam tříd oddělený čárkami, který implementuje uživatelskou proceduru pověření mostu protokolů. Další informace viz Mapování pověření pro souborový server pomocí tříd uživatelské procedury.
protocolBridgePropertiesExitClasses	Určuje seznam tříd oddělený čárkami, který implementuje uživatelskou proceduru vlastností serveru mostu protokolů. Další informace viz ProtocolBridgePropertiesExit2: Vyhledání vlastností souborového serveru protokolů.
IOExitClasses	Určuje seznam tříd oddělený čárkami, který implementuje uživatelskou proceduru I/O. Vypište pouze ty třídy, které implementují rozhraní IOExit, tj. neuvádějte třídy, které implementují ostatní rozhraní uživatelské procedury I/O, například IOExitResourcePath a IOExitChannel. Další informace viz Použití uživatelských procedur I/O přenosu MFT.

Pořadí vyvolání ukončení

Uživatelské procedury zdroje a cíle jsou vyvolány v následujícím pořadí:

1. SourceTransferStartExit
2. DestinationTransferStartExit
3. DestinationTransferEndExit
4. SourceTransferEndExit

Zřetězení zdrojových a cílových uživatelských procedur

Zadáte-li více uživatelských procedur, první uživatelská procedura v seznamu se vyvolá jako první, následovaná druhou uživatelskou procedurou atd. Veškeré změny provedené první uživatelskou procedurou jsou předány jako vstup uživatelské proceduře, která je následně vyvolána atd. Pokud například existují dva počáteční uživatelské procedury pro přenos zdroje, budou veškeré změny provedené v metadatech přenosu první uživatelskou procedurou zadány druhé uživatelské proceduře.

Každá uživatelská procedura vrátí svůj vlastní výsledek. Pokud všechny uživatelské procedury daného typu vrátí POKRAČOVAT jako výsledný kód přenosu, celkový výsledek je POKRAČOVAT. Pokud jeden nebo více uživatelských procedur vrátí hodnotu CANCEL_TRANSFER, celkový výsledek je CANCEL_TRANSFER. Všechny výsledné kódy a řetězce vrácené výstupními výstupy jsou výstupem do protokolu přenosu.

Pokud je celkový výsledek ze zdrojové počáteční uživatelské procedury přenosu POKRAČOVAT, přenos pokračuje pomocí změn provedených uživatelskými procedurami. Pokud je celkový výsledek CANCEL_TRANSFER, budou vyvolány uživatelské procedury ukončení zdrojového přenosu a přenos bude zrušen. Stav dokončení v protokolu přenosu je "zrušeno".

Pokud je celkový výsledek z uživatelských procedur zahájení přenosu cíle POKRAČOVAT, přenos pokračuje pomocí změn provedených z uživatelských procedur. Pokud je celkový výsledek CANCEL_TRANSFER, jsou vyvolány koncové uživatelské procedury cílového přenosu, pak jsou vyvolány koncové uživatelské procedury zdrojového přenosu. Nakonec je přenos zrušen. Stav dokončení v protokolu přenosu je "zrušeno".

Pokud musí zdrojová nebo cílová uživatelská procedura předat informace následujícím uživatelským procedurám buď v řetězci, nebo v pořadí provedení, musí to být provedeno aktualizací metadat přenosu. Použití metadat přenosu je specifické pro implementaci ukončení. Pokud například uživatelská procedura nastaví výsledek návratu na hodnotu CANCEL_TRANSFER a potřebuje komunikovat s následujícími uživatelskými procedurami, že byl přenos zrušen, musí to být provedeno nastavením hodnoty metadat přenosu způsobem, který ostatní uživatelské procedury pochopí.

Příklad

```
sourceTransferStartExitClasses=com.ibm.wmqfte.test.MFTTestSourceTransferStartExit
sourceTransferEndExitClasses=com.ibm.wmqfte.test.MFTTestSourceTransferEndExit
destinationTransferStartExitClasses=com.ibm.wmqfte.test.MFTTestDestinationTransferStartExit
destinationTransferEndExitClasses=com.ibm.wmqfte.test.MFTTestDestinationTransferEndExit
exitClassPath=C:/mycomp/mqft/exits/encryptFileExit.jar;C:/mycomp/mqft/exits/fileFilter.jar
```

Související pojmy

Přizpůsobení produktu MFT pomocí uživatelských procedur

“Metadata pro uživatelské procedury MFT” na stránce 2117
Existují tři různé typy metadat, které lze dodat do uživatelských procedur pro produkt Managed File Transfer: prostředí, přenos a metadata souboru. Tato metadata jsou prezentována jako mapy dvojic klíč-hodnota Java .

“Java rozhraní pro uživatelské procedury MFT” na stránce 2127
Referenční informace o rozhraních Java pro uživatelské procedury naleznete v tématech v této části.

Související odkazy

“Uživatelské procedury monitoru prostředků MFT” na stránce 2121
Uživatelské procedury monitoru prostředků vám umožňují konfigurovat vlastní kód pro spuštění, když je splněna podmínka spouštěče monitoru, před spuštěním přidružené úlohy.

Proměnné prostředí ve vlastnostech MFT

Soubor MFT `agent.properties`

Java rozhraní pro uživatelské procedury MFT

Referenční informace o rozhraních Java pro uživatelské procedury naleznete v tématech v této části.

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“DestinationTransferStartExitrozhraní .java” na stránce 2131

“DestinationTransferEndExit.java” na stránce 2130

“Rozhraní IOExit.java” na stránce 2133

“Rozhraní IOExitChannel.java” na stránce 2135

CDCredentialExitrozhraní .java

CDCredentialExit.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that are invoked as part of
 * user exit routine processing.  This interface defines methods that are
 * invoked by a Connect:Direct bridge agent to map the IBM MQ user ID of the transfer to credentials
 * that are used to access the Connect:Direct node.
 * There will be one instance of each implementation class per Connect:Direct bridge agent. The methods
 * can be called from different threads so the methods must be synchronized.
 */
public interface CDCredentialExit {

    /**
     * Invoked once when a Connect:Direct bridge agent is started. It is intended to initialize
     * any resources that are required by the exit
     *
     * @param bridgeProperties
     *            The values of properties defined for the Connect:Direct bridge.
     *            These values can only be read, they cannot be updated by
     *            the implementation.
     *
     * @return    true if the initialisation is successful and false if unsuccessful
     *            If false is returned from an exit the Connect:Direct bridge agent does not
     *            start.
     */
     public boolean initialize(final Map<String, String> bridgeProperties);

     /**
      * Invoked once per transfer to map the IBM MQ user ID in the transfer message to the
      * credentials to be used to access the Connect:Direct node.
       *
       * @param mqUserId The IBM MQ user ID from which to map to the credentials to be used
      *                 to access the Connect:Direct node
```

```
* @param snode    The name of the Connect:Direct SNODE specified as the cdNode in the
*                 file path. This is used to map the correct user ID and password for the
 *                 SNODE.
* @return         A credential exit result object that contains the result of the map and
 *                 the credentials to use to access the Connect:Direct node
 */

public CDCredentialExitResult mapMQUserId(final String mqUserId, final String snode);

 /**
 * Invoked once when a Connect:Direct bridge agent is shutdown. This method releases
 * any resources that were allocated by the exit
 *
  * @param bridgeProperties
 *            The values of properties defined for the Connect:Direct bridge.
 *            These values can only be read, they cannot be updated by
  *            the implementation.
  *
  * @return
*/
public void shutdown(final Map<String, String> bridgeProperties);      }
```

CredentialExitResult.java

CredentialExitResult.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */

package com.ibm.wmqfte.exitroutine.api;

/**
 * The result of invoking a Credential mapMQUserId exit method.  It is composed of a result
 * code, which determines whether the mapping of the user id was successful, and an optional
 * Credentials object if the mapping is successful.
 */
public class CredentialExitResult {

    private final CredentialExitResultCode resultCode;
    private final Credentials credentials;

    /**
     * Constructor.  Creates a credential exit result object with a specified result
     * code and optionally credentials.
     *
     * @param resultCode
     *            The result code to associate with the exit result being created.
     *
     * @param credentials
     *            The credentials to associate with the exit result being created.
     *            A value of <code>null</code> can be specified to indicate no
     *            credentials. If the resultCode is USER_SUCCESSFULLY_MAPPED the
     *            credentials must be set to a non-null value,
     */
    public CredentialExitResult(CredentialExitResultCode resultCode, Credentials credentials) {
        this.resultCode = resultCode;
        this.credentials = credentials;
    }


    /**
     * Returns the result code associated with this credential exit result
     *
     * @return    the result code associated with this exit result.
     */
    public CredentialExitResultCode getResultCode() {
        return resultCode;
```

```
    }

    /**
     * Returns the credentials associated with this credential exit result
     *
     * @return    the explanation associated with this credential exit result.
     */
    public Credentials getCredentials() {
        return credentials;
    }
}
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“SourceTransferStartExitrozhraní .java” na stránce 2156

“DestinationTransferStartExitrozhraní .java” na stránce 2131

“DestinationTransferEndExit.java” na stránce 2130

“Rozhraní MonitorExit.java” na stránce 2149

“ProtocolBridgeCredentialExit.java” na stránce 2150

DestinationTransferEndExit.java

DestinationTransferEndExit.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing.  This interface defines a method that will be
 * invoked immediately after completing a transfer on the agent acting as the
 * destination of the transfer.
 */
public interface DestinationTransferEndExit {

    /**
     * Invoked immediately after the completion of a transfer on the agent acting as
     * the destination of the transfer.
     *
     * @param transferExitResult
     *            a result object reflecting whether or not the transfer completed
     *            successfully.
     *
     * @param sourceAgentName
     *            the name of the agent acting as the source of the transfer.
     *
     * @param destinationAgentName
     *            the name of the agent acting as the destination of the
     *            transfer.     This is the name of the agent that the
     *            implementation of this method will be invoked from.
     *
     * @param environmentMetaData
     *            meta data about the environment in which the implementation
     *            of this method is running.  This information can only be read,
     *            it cannot be updated by the implementation.  The constants
     *            defined in <code>EnvironmentMetaDataConstants</code> class can
     *            be used to access the data held by this map.
     *
     * @param transferMetaData
```

```
     *            meta data to associate with the transfer.  The information can
     *            only be read, it cannot be updated by the implementation.  This
     *            map may also contain keys with IBM reserved names.  These
     *            entries are defined in the <code>TransferMetaDataConstants</code>
     *            class and have special semantics.
     *
     * @param fileResults
     *            a list of file transfer result objects that describe the source
     *            file name, destination file name and result of each file transfer
     *            operation attempted.
     *
     * @return    an optional description to enter into the log message describing
     *            transfer completion.  A value of <code>null</code> can be used
     *            when no description is required.
     */
    String onDestinationTransferEnd(TransferExitResult transferExitResult,
                    String sourceAgentName,
                    String destinationAgentName,
                    Map<String, String>environmentMetaData,
                    Map<String, String>transferMetaData,
                    List<FileTransferResult>fileResults);
}
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“SourceTransferStartExitrozhraní .java” na stránce 2156

“SourceTransferEndExit.java” na stránce 2155

“DestinationTransferStartExitrozhraní .java” na stránce 2131

“Rozhraní MonitorExit.java” na stránce 2149

“ProtocolBridgeCredentialExit.java” na stránce 2150

DestinationTransferStartExitrozhraní .java

DestinationTransferStartExit.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing.  This interface defines a method that will be
 * invoked immediately prior to starting a transfer on the agent acting as the
 * destination of the transfer.
 */
public interface DestinationTransferStartExit {

    /**
     * Invoked immediately prior to starting a transfer on the agent acting as
     * the destination of the transfer.
     *
     * @param sourceAgentName
     *            the name of the agent acting as the source of the transfer.
     *
     * @param destinationAgentName
     *            the name of the agent acting as the destination of the
     *            transfer.     This is the name of the agent that the
     *            implementation of this method will be invoked from.
     *
     * @param environmentMetaData
```

```
     *            meta data about the environment in which the implementation
     *            of this method is running.  This information can only be read,
     *            it cannot be updated by the implementation.  The constants
     *            defined in <code>EnvironmentMetaDataConstants</code> class can
     *            be used to access the data held by this map.
     *
     * @param transferMetaData
     *            meta data to associate with the transfer.  The information can
     *            only be read, it cannot be updated by the implementation.  This
     *            map may also contain keys with IBM reserved names.  These
     *            entries are defined in the <code>TransferMetaDataConstants</code>
     *            class and have special semantics.
     *
     * @param fileSpecs
     *            a list of file specifications that govern the file data to
     *            transfer.  The implementation of this method can modify the
     *            entries in this list and the changes will be reflected in the
     *            files transferred. However, new entries may not be added and
     *            existing entries may not be removed.
     *
     * @return    a transfer exit result object which is used to determine if the
     *            transfer should proceed, or be cancelled.
     */
    TransferExitResult onDestinationTransferStart(String sourceAgentName,
                                                  String destinationAgentName,
                                            Map<String, String> environmentMetaData,
                                            Map<String, String> transferMetaData,
                                            List<Reference<String>> fileSpecs);
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“SourceTransferStartExitrozhraní .java” na stránce 2156

“SourceTransferEndExit.java” na stránce 2155

“DestinationTransferEndExit.java” na stránce 2130

“Rozhraní MonitorExit.java” na stránce 2149

“ProtocolBridgeCredentialExit.java” na stránce 2150

Rozhraní FileTransferResult.java

FileTransferResult.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */

package com.ibm.wmqfte.exitroutine.api;

/**
 * Result information about a file transfer.
 */
public interface FileTransferResult {

    /** An enumeration for the <code>getCorrelatorType()</code> method. */
    public enum CorrelationInformationType {
        /** No correlation information is available for this result */
        NONE,
        /**
         * The correlation information relates to work done in
         * IBM Sterling File Gateway.
         */
        SFG
```

```
    }

    /**
     * Returns the source file specification, from which the file was transferred.
     *
     * @return    the source file specification, from which the file was
     *            transferred.
     */
    String getSourceFileSpecification();

    /**
     * Returns the destination file specification, to which the file was transferred.
     *
     * @return    the destination file specification, to which the file was
     *            transferred.  A value of <code>null</code> may be returned
     *            if the transfer did not complete successfully.
     */
    String getDestinationFileSpecification();

    /**
     * Returns the result of the file transfer operation.
     *
     * @return    the result of the file transfer operation.
     */
    FileExitResult getExitResult();

    /**
     * @return an enumerated value that identifies the product to which this correlating
     *         information relates.
     */
    CorrelationInformationType getCorrelatorType();

    /**
     * @return the first string component of the correlating identifier that relates
     *         this transfer result to work done in another product.  A value of null
     *         may be returned either because the other product does not utilize a
     *         string based correlation information or because there is no correlation
     *         information.
     */
    String getString1Correlator();

    /**
     * @return the first long component of the correlating identifier that relates
     *         this transfer result to work done in another product.  A value of zero
     *         is returned when there is no correlation information or the other
     *         product does not utilize long based correlation information or because
     *         the value really is zero!
     */
    long getLong1Correlator();
}
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“SourceTransferStartExitrozhraní .java” na stránce 2156

“DestinationTransferStartExitrozhraní .java” na stránce 2131

“DestinationTransferEndExit.java” na stránce 2130

“Rozhraní MonitorExit.java” na stránce 2149

“ProtocolBridgeCredentialExit.java” na stránce 2150

Rozhraní IOExit.java

IOExit.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2011, 2024. All Rights Reserved.
```

```
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.util.Map;

import com.ibm.wmqfte.exitroutine.api.IOExitRecordResourcePath.RecordFormat;

/**
 * An interface that is implemented by classes that you want to be invoked as
 * part of user exit routine processing. This interface defines methods that
 * will be invoked during transfers to perform the underlying file system I/O
 * work for WMQFTE transfers.
 * <p>
 * The {@link #initialize(Map)} method will be called once when the exit is
 * first installed. The WMQFTE agent properties are passed to this method, thus
 * enabling the exit to understand its environment.
 * <p>
 * The {@link #isSupported(String)} method will be invoked during WMQFTE
 * transfers to determine whether the user exit should be used. If the
 * {@link #isSupported(String)} method returns a value of {@code true}, the
 * {@link #newPath(String)} method will be invoked for the paths specified for
 * the transfer request. The returned {@link IOExitPath} instance from a
 * {@link #newPath(String)} method invocation will then be used by the WMQFTE
 * transfer to obtain information about the resource and to transfer data to or
 * from the resource.
 * <p>
 * To obtain transfer context for an I/O exit, a {@link SourceTransferStartExit}
 * or {@link DestinationTransferStartExit} as appropriate, should be installed
 * to enable information to be seen by this exit. The
 * {@link SourceTransferStartExit} or {@link DestinationTransferStartExit} are
 * passed the transfer's environment, metadata, and a list of file
 * specifications for the transfer. The paths for the file specifications are
 * the paths passed to the I/O exit's {@link #newPath(String)} method.
 * <p>
 * Note also that the {@link #isSupported(String)} and {@link #newPath(String)}
 * methods might be called at other times by a WMQFTE agent and not just during
 * transfers. For example, at transfer setup time the I/O system is queried to
 * resolve the full resource paths for transfer.
 */
public interface IOExit {

    /**
     * Invoked once when the I/O exit is first required for use. It is intended
     * to initialize any resources that are required by the exit.
     *
     * @param agentProperties
     *            The values of properties defined for the WMQFTE agent. These
     *            values can only be read, they cannot be updated by the
     *            implementation.
     * @return {@code true} if the initialization is successful and {@code
     *         false} if unsuccessful. If {@code false} is returned from an
     *         exit, the exit will not be used.
     */
    boolean initialize(final Map<String, String> agentProperties);

    /**
     * Indicates whether this I/O user exit supports the specified path.
     * <p>
     * This method is used by WMQFTE to determine whether the I/O user exit
     * should be used within a transfer. If no I/O user exit returns true for
     * this method, the default WMQFTE file I/O function will be used.
     *
     * @param path
     *            The path to the required I/O resource.
     * @return {@code true} if the specified path is supported by the I/O exit,
     *         {@code false} otherwise
     */
    boolean isSupported(String path);

    /**
     * Obtains a new {@link IOExitPath} instance for the specified I/O resource
     * path.
     * <p>
     * This method will be invoked by WMQFTE only if the
     * {@link #isSupported(String)} method has been called for the path and
     * returned {@code true}.
     *
```

```
     * @param path
     *            The path to the required I/O resource.
     * @return A {@link IOExitPath} instance for the specified path.
     * @throws IOException
     *             If the path cannot be created for any reason.
     */
    IOExitPath newPath(String path) throws IOException;

    /**
     * Obtains a new {@link IOExitPath} instance for the specified I/O resource
     * path and passes record format and length information required by the
     * WMQFTE transfer.
     * <p>
     * Typically this method will be called for the following cases:
     * <ul>
     * <li>A path where a call to {@link #newPath(String)} has previously
     * returned a {@link IOExitRecordResourcePath} instance and WMQFTE is
     * re-establishing a new {@link IOExitPath} instance for the path, from an
     * internally-serialized state. The passed recordFormat and recordLength
     * will be the same as those for the original
     * {@link IOExitRecordResourcePath} instance.</li>
     * <li>A transfer destination path where the source of the transfer is
     * record oriented. The passed recordFormat and recordLength will be the
     * same as those for the source.</li>
     * </ul>
     * The implementation can act on the record format and length information as
     * deemed appropriate. For example, for a destination agent if the
     * destination does not already exist and the source of the transfer is
     * record oriented, the passed recordFormat and recordLength information
     * could be used to create an appropriate record-oriented destination path.
     * If the destination path already exists, the passed recordFormat and
     * recordLength information could be used to perform a compatibility check
     * and throw an {@link IOException} if the path is not compatible. A
     * compatibility check could ensure that a record oriented path's record
     * format is the same as the passed record format or that the record length
     * is greater or equal to the passed record length.
     * <p>
     * This method will be invoked by WMQFTE only if the
     * {@link #isSupported(String)} method has been called for the path and
     * returned {@code true}.
     *
     * @param path
     *            The path to the required I/O resource.
     * @param recordFormat
     *            The advised record format.
     * @param recordLength
     *            The advised record length.
     * @return A {@link IOExitPath} instance for the specified path.
     * @throws IOException
     *             If the path cannot be created for any reason. For example,
     *             the passed record format or length is incompatible with the
     *             path's actual record format or length.
     */
    IOExitPath newPath(String path, RecordFormat recordFormat, int recordLength)
            throws IOException;
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

Rozhraní IOExitChannel.java

IOExitChannel.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
```

```
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.nio.ByteBuffer;

/**
 * Represents a channel that enables data to be read from or written to an
 * {@link IOExitResourcePath} resource.
 */
public interface IOExitChannel {

    /**
     * Obtains the data size for the associated {@link IOExitResourcePath} in
     * bytes.
     *
     * @return The data size in bytes.
     * @throws IOException
     *             If a problem occurs while attempting obtain the size.
     */
    long size() throws IOException;

    /**
     * Closes the channel, flushing any buffered write data to the resource and
     * releasing any locks.
     *
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while closing the resource.
     *             This means that WMQFTE can attempt to recover the transfer.
     * @throws IOException
     *             If some other I/O problem occurs. For example, the channel might
     *             already be closed.
     */
    void close() throws RecoverableIOException, IOException;

    /**
     * Reads data from this channel into the given buffer, starting at this
     * channel's current position, and updates the current position by the
     * amount of data read.
     * <p>
     * Data is copied into the buffer starting at its current position and up to
     * its limit. On return, the buffer's position is updated to reflect the
     * number of bytes read.
     *
     * @param buffer
     *            The buffer that the data is to be copied into.
     * @return The number of bytes read, which might be zero, or -1 if the end of
     *         data has been reached.
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while reading the data. For a
     *             WMQFTE transfer this means that it will attempt to recover.
     * @throws IOException
     *             If some other I/O problem occurs. For a WMQFTE transfer this
     *             means that it will be failed.
     */
    int read(ByteBuffer buffer) throws RecoverableIOException, IOException;

    /**
     * Writes data to this channel from the given buffer, starting at this
     * channel's current position, and updates the current position by the
     * amount of data written. The channel's resource is grown to accommodate
     * the data, if necessary.
     * <p>
     * Data is copied from the buffer starting at its current position and up to
     * its limit. On return, the buffer's position is updated to reflect the
     * number of bytes written.
     *
     * @param buffer
     *            The buffer containing the data to be written.
     * @return The number of bytes written, which might be zero.
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while writing the data. For a
     *             WMQFTE transfer this means that it will attempt to recover.
     * @throws IOException
     *             If some other I/O problem occurs. For a WMQFTE transfer this
     *             means that it will be failed.
     */
    int write(ByteBuffer buffer) throws RecoverableIOException, IOException;

    /**
     * Forces any updates to this channel's resource to be written to its
     * storage device.
```

```
     * <p>
     * This method is required to force changes to both the resource's content
     * and any associated metadata to be written to storage.
     *
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while performing the force.
     *             For a WMQFTE transfer this means that it will attempt to
     *             recover.
     * @throws IOException
     *             If some other I/O problem occurs. For a WMQFTE transfer this
     *             means that it will be failed.
     */
    void force() throws RecoverableIOException, IOException;

    /**
     * Attempts to lock the entire resource associated with the channel for
     * shared or exclusive access.
     * <p>
     * The intention is for this method not to block if the lock is currently
     * unavailable.
     *
     * @param shared
     *            {@code true} if a shared lock is required, {@code false} if an
     *            exclusive lock is required.
     * @return A {@link IOExitLock} instance representing the newly acquired
     *         lock or null if the lock cannot be obtained.
     * @throws IOException
     *             If a problem occurs while attempting to acquire the lock.
     */
    IOExitLock tryLock(boolean shared) throws IOException;
}
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

Rozhraní IOExitLock.java

IOExitLock.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *   © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a lock on a resource for either shared or exclusive access.
 * {@link IOExitLock} instances are returned from
 * {@link IOExitChannel#tryLock(boolean)} calls and WMQFTE will request the
 * release of the lock at the appropriate time during a transfer. Additionally, when
 * a {@link IOExitChannel#close()} method is called it will be the
 * responsibility of the channel to release any associated locks.
 */
public interface IOExitLock {

    /**
     * Releases the lock.
     * <p>
     * After this method has been successfully called the lock is to be deemed as invalid.
     *
     * @throws IOException
     *             If the channel associated with the lock is not open or
     *             another problem occurs while attempting to release the lock.
     */
```

```
    void release() throws IOException;

    /**
     * Indicates whether this lock is valid.
     * <p>
     * A lock is considered valid until its @ {@link #release()} method is
     * called or the associated {@link IOExitChannel} is closed.
     *
     * @return {@code true} if this lock is valid, {@code false} otherwise.
     */
    boolean isValid();

    /**
     * @return {@code true} if this lock is for shared access, {@code false} if
     *         this lock is for exclusive access.
     */
    boolean isShared();
}
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

Rozhraní IOExitPath.java

IOExitPath.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * Represents an abstract path that can be inspected and queried by WMQFTE for
 * transfer purposes.
 * <p>
 * There are two types of path supported:
 * <ul>
 * <li>{@link IOExitResourcePath} - Represents a path that denotes a data
 * resource. For example, a file, directory, or group of database records.</li>
 * <li>{@link IOExitWildcardPath} - Represents a wildcard path that can be
 * expanded to multiple {@link IOExitResourcePath} instances.</li>
 * </ul>
 */
public abstract interface IOExitPath {

    /**
     * Obtains the abstract path as a {@link String}.
     *
     * @return The abstract path as a {@link String}.
     */
    String getPath();

    /**
     * Obtains the name portion of this abstract path as a {@link String}.
     * <p>
     * For example, a UNIX-style file system implementation evaluates the
     * path {@code /home/fteuser/file1.txt} as having a name of {@code
     * file1.txt}.
     *
     * @return the name portion of this abstract path as a {@link String}.
     */
    String getName();

    /**
     * Obtains the parent path for this abstract path as a {@link String}.
```

```
     * <p>
     * For example, a UNIX-style file system implementation evaluates the
     * path {@code /home/fteuser/file1.txt} as having a parent path of {@code
     * /home/fteuser}.
     *
     * @return The parent portion of the path as a {@link String}.
     */
    String getParent();

    /**
     * Obtains the abstract paths that match this abstract path.
     * <p>
     * If this abstract path denotes a directory resource, a list of paths
     * for all resources within the directory are returned.
     * <p>
     * If this abstract path denotes a wildcard, a list of all paths
     * matching the wildcard are returned.
     * <p>
     * Otherwise null is returned, because this abstract path probably denotes a
     * single file resource.
     *
     * @return An array of {@IOExitResourcePath}s that
     *         match this path, or null if this method is not applicable.
     */
    IOExitResourcePath[] listPaths();
}
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

IOExitPropertiesrozhraní .java

IOExitProperties.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * Properties that determine how WMQFTE treats an {@link IOExitPath} for certain
 * aspects of I/O. For example, whether to use intermediate files.
 */
public class IOExitProperties {

    private boolean rereadSourceOnRestart = true;
    private boolean rechecksumSourceOnRestart = true;
    private boolean rechecksumDestinationOnRestart = true;
    private boolean useIntermediateFileAtDestination = true;
    private boolean requiresSingleThreadedChannelIO = false;

    /**
     * Determines whether the I/O exit implementation expects the resource to be
     * re-read from the start if a transfer is restarted.
     *
     * @return {@code true} if, on restart, the I/O exit expects the source
     *         resource to be opened at the beginning and re-read from the
     *         beginning (the {@link IOExitPath#openForRead(long)} method is
     *         always invoked with 0L as an argument). {@code false} if, on
     *         restart, the I/O exit expects the source to be opened at the
     *         offset that the source agent intends to start reading from (the
     *         {@link IOExitPath#openForRead(long)} method can be invoked with a
     *         non-zero value as its argument).
     */
    public boolean getRereadSourceOnRestart() {
```

```
    return rereadSourceOnRestart;
}

/**
 * Sets the value to determine whether the I/O exit implementation expects
 * the resource to be re-read from the beginning if a transfer is restarted.
 * <p>
 * The default is {@code true}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param rereadSourceOnRestart
 *            {@code true} if, on restart, the I/O exit expects the source
 *            resource to be opened at the beginning and re-read from the
 *            beginning (the {@link IOExitPath#openForRead(long)} method
 *            is always invoked with 0L as an argument). {@code false}
 *            if, on restart, the I/O exit expects the source to be opened
 *            at the offset that the source agent intends to start reading
 *            from (the {@link IOExitPath#openForRead(long)} method can be
 *            invoked with a non-zero value as its argument).
 */
public void setRereadSourceOnRestart(boolean rereadSourceOnRestart) {
    this.rereadSourceOnRestart = rereadSourceOnRestart;
}

/**
 * Determines whether the I/O exit implementation requires the source
 * resource to be re-checksummed if the transfer is restarted.
 * Re-checksumming takes place only if the
 * {@link #getRereadSourceOnRestart()} method returns {@code true}.
 *
 * @return {@code true} if, on restart, the I/O exit expects the already-
 *         transferred portion of the source to be re-checksummed for
 *         inconsistencies. Use this option in environments
 *         where the source could be changed during a restart. {@code
 *         false} if, on restart, the I/O exit does not require the
 *         already-transferred portion of the source to be re-checksummed.
 */
public boolean getRechecksumSourceOnRestart() {
    return rechecksumSourceOnRestart;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * the source resource to be re-checksummed if the transfer is restarted.
 * Re-checksumming takes place only if the
 * {@link #getRereadSourceOnRestart()} method returns {@code true}.
 * <p>
 * The default is {@code true}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param rechecksumSourceOnRestart
 *            {@code true} if, on restart, the I/O exit expects the already
 *            transferred portion of the source to be re-checksummed
 *            for inconsistencies. Use this option in environments
 *            where the source could be changed during a restart.
 *            {@code false} if, on restart, the I/O exit does not
 *            require the already-transferred portion of the source to be
 *            re-checksummed.
 */
public void setRechecksumSourceOnRestart(boolean rechecksumSourceOnRestart) {
    this.rechecksumSourceOnRestart = rechecksumSourceOnRestart;
}

/**
 * Determines whether the I/O exit implementation requires the destination
 * resource to be re-checksummed if the transfer is restarted.
 *
 * @return {@code true} if, on restart, the I/O exit expects the already
 *         transferred portion of the destination to be re-checksummed to
 *         check for inconsistencies. This option should be used in
 *         environments where the destination could have been changed while
 *         a restart is occurring. {@code false} if, on restart, the I/O exit
 *         does not require the already transferred portion of the
 *         destination to be re-checksummed.
 */
public boolean getRechecksumDestinationOnRestart() {
    return rechecksumDestinationOnRestart;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * the destination resource to be re-checksummed if the transfer is
```

```
     * restarted.
     * <p>
     * The default is {@code true}. The I/O exit should call this method when
     * required to change this value.
     *
     * @param rechecksumDestinationOnRestart
     *            {@code true} if, on restart, the I/O exit expects the already-
     *            transferred portion of the destination to be re-checksummed
     *            for inconsistencies. Use this option in environments
     *            where the destination could have been changed during a
     *            restart. {@code false} if, on restart, the I/O exit does not
     *            require the already-transferred portion of the destination
     *            to be re-checksummed.
     */
    public void setRechecksumDestinationOnRestart(
            boolean rechecksumDestinationOnRestart) {
        this.rechecksumDestinationOnRestart = rechecksumDestinationOnRestart;
    }

    /**
     * Determines whether the I/O exit implementation requires the use of an
     * intermediate file when writing the data at the destination. The
     * intermediate file mechanism is typically used to prevent an incomplete
     * destination resource from being processed.
     *
     * @return {@code true} if data should be written to an intermediate file at
     *         the destination and then renamed (to the requested destination
     *         path name as specified in the transfer request) after the transfer is
     *         complete. {@code false} if data should be written directly to the
     *         requested destination path name without the use of an
     *         intermediate file.
     */
    public boolean getUseIntermediateFileAtDestination() {
        return useIntermediateFileAtDestination;
    }

    /**
     * Sets the value to determine whether the I/O exit implementation requires
     * the use of an intermediate file when writing the data at the destination.
     * The intermediate file mechanism is typically used to prevent an
     * incomplete destination resource from being processed.
     *
     * <p>
     * The default is {@code true}. The I/O exit should call this method when
     * required to change this value.
     *
     * @param useIntermediateFileAtDestination
     *            {@code true} if data should be written to an intermediate file
     *            at the destination and then renamed (to the requested
     *            destination path name as specified in the transfer request) after
     *            the transfer is complete. {@code false} if data should be written
     *            directly to the requested destination path name without the
     *            use of an intermediate file
     */
    public void setUseIntermediateFileAtDestination(
            boolean useIntermediateFileAtDestination) {
        this.useIntermediateFileAtDestination = useIntermediateFileAtDestination;
    }

    /**
     * Determines whether the I/O exit implementation requires
     * {@link IOExitChannel} instances to be accessed by a single thread only.
     *
     * @return {@code true} if {@link IOExitChannel} instances are to be
     *         accessed by a single thread only.
     */
    public boolean requiresSingleThreadedChannelIO() {
        return requiresSingleThreadedChannelIO;
    }

    /**
     * Sets the value to determine whether the I/O exit implementation requires
     * channel operations for a particular instance to be accessed by a
     * single thread only.
     * <p>
     * For certain I/O implementations it is necessary that resource path
     * operations such as open, read, write, and close are invoked only from a
     * single execution {@link Thread}. When set {@code true}, WMQFTE ensures
     * that the following are invoked on a single thread:
     * <ul>
     * <li>{@link IOExitResourcePath#openForRead(long) method and all methods of
     * the returned {@link IOExitChannel} instance.</li>
```

```
     * <li>{@link IOExitResourcePath#openForWrite(boolean)) method and all
     * methods of the returned {@link IOExitChannel} instance.</li>
     * </ul>
     * <p>
     * This has a slight performance impact, hence enable single-threaded channel
     * I/O only when absolutely necessary.
     * <p>
     * The default is {@code false}. The I/O exit should call this method when
     * required to change this value.
     *
     * @param requiresSingleThreadedChannelIO
     *            {@code true} if {@link IOExitChannel} instances are to be
     *            accessed by a single thread only.
     */
    public void setRequiresSingleThreadedChannelIO(boolean requiresSingleThreadedChannelIO) {
        this.requiresSingleThreadedChannelIO = requiresSingleThreadedChannelIO;
    }
}
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

Rozhraní IOExitRecordChannel.java

IOExitRecordChannel.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.nio.ByteBuffer;

/**
 * Represents a channel that enables records of data to be read from or written
 * to an {@link IOExitRecordResourcePath} resource.
 * <p>
 * This is an extension of the {@link IOExitChannel} interface such that the
 * {@link #read(java.nio.ByteBuffer)} and {@link #write(java.nio.ByteBuffer)}
 * methods are expected to deal in whole records of data only. That is, the
 * {@link java.nio.ByteBuffer} returned from the read method and passed to the
 * write method is assumed to contain one or more complete records.
 */
public interface IOExitRecordChannel extends IOExitChannel {

    /**
     * Reads records from this channel into the given buffer, starting at this
     * channel's current position, and updates the current position by the
     * amount of data read.
     * <p>
     * Record data is copied into the buffer starting at its current position
     * and up to its limit. On return, the buffer's position is updated to
     * reflect the number of bytes read.
     * <p>
     * Only whole records are copied into the buffer.
     * <p>
     * For a fixed-record-format resource, this might be multiple records. The
     * amount of data in the return buffer does not necessarily need to be a
     * multiple of the record length, but the last record is still to be treated
     * as a complete record and padded as required by the caller.
     * <p>
     * For a variable-format resource, this is a single whole record of a size
     * corresponding to the amount of return data or multiple whole records with
     * all except the last being treated as records of maximum size.
```

```
     *
     * @param buffer
     *            The buffer that the record data is to be copied into.
     * @return The number of bytes read, which might be zero, or -1 if the end of
     *         data has been reached.
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while reading the data. For a
     *             WMQFTE transfer this means that it will attempt to recover.
     * @throws IOException
     *             If some other I/O problem occurs, for example, if the passed
     *             buffer is insufficient to contain at least one complete
     *             record). For a WMQFTE transfer this means that it will be
     *             failed.
     */
    int read(ByteBuffer buffer) throws RecoverableIOException, IOException;

    /**
     * Writes records to this channel from the given buffer, starting at this
     * channel's current position, and updates the current position by the
     * amount of data written. The channel's resource is grown to accommodate
     * the data, if necessary.
     * <p>
     * Record data is copied from the buffer starting at its current position
     * and up to its limit. On return, the buffer's position is updated to
     * reflect the number of bytes written.
     * <p>
     * The buffer is expected to contain only whole records.
     * <p>
     * For a fixed-record-format resource, this might be multiple records and if
     * there is insufficient data in the buffer for a complete record, the
     * record is to be padded as required to complete the record.
     * <p>
     * For a variable-record format resource the buffer is normally expected to
     * contain a single record of length corresponding to the amount of data
     * within the buffer. However, if the amount of data within the buffer
     * exceeds the maximum record length, the implementation can either:
     * <ol>
     * <li>throw an {@link IOException} indicating that it cannot handle the
     * situation.</li>
     * <li>Consume a record's worth of data from the buffer, leaving the remaining
     * data within the buffer.</li>
     * <li>Consume all the buffer data and just write what it can to the current
     * record. This effectively truncates the data.</li>
     * <li>Consume all the buffer data and write to multiple records.</li>
     * </ol>
     *
     * @param buffer
     *            The buffer containing the data to be written.
     * @return The number of bytes written, which might be zero.
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while writing the data. For a
     *             WMQFTE transfer this means that it will attempt to recover.
     * @throws IOException
     *             If some other I/O problem occurs. For a WMQFTE transfer this
     *             means that it will be failed.
     */
    int write(ByteBuffer buffer) throws RecoverableIOException, IOException;

}
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

z/OS *Rozhraní IOExitRecordResourcePath.java*

IOExitRecordResourcePath.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *   © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
```

```
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a path that denotes a record-oriented data resource (for example,
 * a z/OS data set). It allows the data to be located, the record format to be
 * understood, and {@link IOExitRecordChannel} instances to be created for read
 * or write operations.
 */
public interface IOExitRecordResourcePath extends IOExitResourcePath {

    /**
     * Record formats for record-oriented resources.
     */
    public enum RecordFormat {
        FIXED, VARIABLE
    }

    /**
     * Obtains the record length for records that are maintained by the resource
     * denoted by this abstract path.
     * <p>
     * For a resource with fixed-length records, the data for each record read
     * and written is assumed to be this length.
     * <p>
     * For a resource with variable-length records, this is the maximum length
     * for a record's data.
     * <p>
     * This method should return a value greater than zero, otherwise it can
     * result in the failure of a WMQFTE transfer that involves this abstract
     * path.
     *
     * @return The record length, in bytes, for records maintained by the
     *         resource.
     */
    int getRecordLength();

    /**
     * Obtains record format, as a {@link RecordFormat} instance, for records
     * that are maintained by the resource denoted by this abstract path.
     *
     * @return A {@link RecordFormat} instance for the record format for records
     *         that are maintained by the resource denoted by this abstract
     *         path.
     */
    RecordFormat getRecordFormat();

    /**
     * Opens a {@link IOExitRecordChannel} instance for reading data from the
     * resource denoted by this abstract path. The current data byte position
     * for the resource is expected to be the passed position value, such that
     * when {@link IOExitRecordChannel#read(java.nio.ByteBuffer)} is called,
     * data starting from that position is read.
     * <p>
     * Note that the data byte read position will be on a record boundary.
     *
     * @param position
     *            The required data byte read position.
     * @return A new {@link IOExitRecordChannel} instance allowing data to be
     *         read from the resource denoted by this abstract path.
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while attempting to open the
     *             resource for reading. This means that WMQFTE can attempt to
     *             recover the transfer.
     * @throws IOException
     *             If some other I/O problem occurs.
     */
    IOExitRecordChannel openForRead(long position)
            throws RecoverableIOException, IOException;

    /**
     * Opens a {@link IOExitRecordChannel} instance for writing data to the
     * resource denoted by this abstract path. Writing of data, using the
     * {@link IOExitRecordChannel#write(java.nio.ByteBuffer)} method, starts at
     * either the beginning of the resource or end of the current data for the
     * resource, depending on the specified append parameter.
```

```
     *
     * @param append
     *            When {@code true} indicates that data written to the resource
     *            should be appended to the end of the current data. When
     *            {@code false} indicates that writing of data is to start at
     *            the beginning of the resource; any existing data is lost.
     * @return A new {@link IOExitRecordChannel} instance allowing data to be
     *         written to the resource denoted by this abstract path.
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while attempting to open the
     *             resource for writing. This means that WMQFTE can attempt to
     *             recover the transfer.
     * @throws IOException
     *             If some other I/O problem occurs.
     */
    IOExitRecordChannel openForWrite(boolean append)
            throws RecoverableIOException, IOException;
}
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

Rozhraní IOExitResourcePath.java

IOExitResourcePath.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *   © Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a path that denotes a data resource (for example, a file,
 * directory, or group of database records). It allows the data to be located
 * and {@link IOExitChannel} instances to be created for read or write
 * operations.
 * <p>
 * There are two types of data resources as follows:
 * <ul>
 * <li>Directory - a container for other data resources. The
 * {@link #isDirectory()} method returns {@code true} for these.</li>
 * <li>File - a data container. This allows data to be read from or written to
 * it. The {@link #isFile()} method returns {@code true} for these.</li>
 * </ul>
 */
public interface IOExitResourcePath extends IOExitPath {

    /**
     * Creates a new {@link IOExitResourcePath} instance for a child path of the
     * resource denoted by this abstract path.
     * <p>
     * For example, with a UNIX-style path, {@code
     * IOExitResourcePath("/home/fteuser/test").newPath("subtest")} could be
     * equivalent to: {@code IOExitResourcePath("/home/fteuser/test/subtest")}
     *
     * @param child
     *            The child path name.
     * @return A new {@link IOExitResourcePath} instance that represents a child
     *         of this path.
     */
    IOExitResourcePath newPath(final String child);

    /**
```

```
     * Creates the directory path for the resource denoted by this abstract
     * path, including any necessary but nonexistent parent directories. If the
     * directory path already exists, this method has no effect.
     * <p>
     * If this operation fails, it might have succeeded in creating some of the
     * necessary parent directories.
     *
     * @throws IOException
     *             If the directory path cannot be fully created, when it does
     *             not already exist.
     */
    void makePath() throws IOException;

    /**
     * Obtains the canonical path of the abstract path as a {@link String}.
     * <p>
     * A canonical path is defined as being absolute and unique. For example,
     * the path can be represented as UNIX-style relative path: {@code
     * test/file.txt} but the absolute and unique canonical path representation
     * is: {@code /home/fteuser/test/file.txt}
     *
     * @return The canonical path as a {@link String}.
     * @throws IOException
     *             If the canonical path cannot be determined for any reason.
     */
    String getCanonicalPath() throws IOException;

    /**
     * Tests if this abstract path is an absolute path.
     * <p>
     * For example, a UNIX-style path, {@code /home/fteuser/test} is an absolute
     * path, whereas {@code fteuser/test} is not.
     *
     * @return {@code true} if this abstract path is an absolute path, {@code
     *         false} otherwise.
     */
    boolean isAbsolute();

    /**
     * Tests if the resource denoted by this abstract path exists.
     *
     * @return {@code true} if the resource denoted by this abstract path
     *         exists, {@code false} otherwise.
     * @throws IOException
     *             If the existence of the resource cannot be determined for any
     *             reason.
     */
    boolean exists() throws IOException;

    /**
     * Tests whether the calling application can read the resource denoted by
     * this abstract path.
     *
     * @return {@code true} if the resource for this path exists and can be
     *         read, {@code false} otherwise.
     * @throws IOException
     *             If a problem occurs while attempting to determine if the
     *             resource can be read.
     */
    boolean canRead() throws IOException;

    /**
     * Tests whether the calling application can modify the resource denoted by
     * this abstract path.
     *
     * @return {@code true} if the resource for this path exists and can be
     *         modified, {@code false} otherwise.
     * @throws IOException
     *             If a problem occurs while attempting to determine if the
     *             resource can be modified.
     */
    boolean canWrite() throws IOException;

    /**
     * Tests whether the specified user is permitted to read the resource
     * denoted by this abstract path.
     * <p>
     * When WMQFTE invokes this method, the user identifier is the MQMD user
     * identifier for the requesting transfer.
     *
     * @param userId
     *            User identifier to test for access.
```

```
     * @return {@code true} if the resource for this abstract path exists and is
     *         permitted to be read by the specified user, {@code false}
     *         otherwise.
     * @throws IOException
     *             If a problem occurs while attempting to determine if the user
     *             is permitted to read the resource.
     */
    boolean readPermitted(String userId) throws IOException;

    /**
     * Tests whether the specified user is permitted to modify the resource
     * denoted by this abstract path.
     * <p>
     * When WMQFTE invokes this method, the user identifier is the MQMD user
     * identifier for the requesting transfer.
     *
     * @param userId
     *            User identifier to test for access.
     * @return {@code true} if the resource for this abstract path exists and is
     *         permitted to be modified by the specified user, {@code false}
     *         otherwise.
     * @throws IOException
     *             If a problem occurs while attempting to determine if the user
     *             is permitted to modify the resource.
     */
    boolean writePermitted(String userId) throws IOException;

    /**
     * Tests if the resource denoted by this abstract path is a directory-type
     * resource.
     *
     * @return {@code true} if the resource denoted by this abstract path is a
     *         directory type resource, {@code false} otherwise.
     */
    boolean isDirectory();

    /**
     * Creates the resource denoted by this abstract path, if it does not
     * already exist.
     *
     * @return {@code true} if the resource does not exist and was successfully
     *         created, {@code false} if the resource already existed.
     * @throws RecoverableIOException
     *             If a recoverable problem occurs while attempting to create
     *             the resource. This means that WMQFTE can attempt to recover
     *             the transfer.
     * @throws IOException
     *             If some other I/O problem occurs.
     */
    boolean createNewPath() throws RecoverableIOException, IOException;

    /**
     * Tests if the resource denoted by this abstract path is a file-type
     * resource.
     *
     * @return {@code true} if the resource denoted by this abstract path is a
     *         file type resource, {@code false} otherwise.
     */
    boolean isFile();

    /**
     * Obtains the last modified time for the resource denoted by this abstract
     * path.
     * <p>
     * This time is measured in milliseconds since the epoch (00:00:00 GMT,
     * January 1, 1970).
     *
     * @return The last modified time for the resource denoted by this abstract
     *         path, or a value of 0L if the resource does not exist or a
     *         problem occurs.
     */
    long lastModified();

    /**
     * Deletes the resource denoted by this abstract path.
     * <p>
     * If the resource is a directory, it must be empty for the delete to work.
     *
     * @throws IOException
     *             If the delete of the resource fails for any reason.
     */
    void delete() throws IOException;
```

```
/**
 * Renames the resource denoted by this abstract path to the specified
 * destination abstract path.
 * <p>
 * The rename should still be successful if the resource for the specified
 * destination abstract path already exists and it is possible to replace
 * it.
 *
 * @param destination
 *            The new abstract path for the resource denoted by this
 *            abstract path.
 * @throws IOException
 *             If the rename of the resource fails for any reason.
 */
void renameTo(IOExitResourcePath destination) throws IOException;

/**
 * Creates a new path to use for writing to a temporary resource that did
 * not previously exist.
 * <p>
 * The implementation can choose the abstract path name for the temporary
 * resource. However, for clarity and problem diagnosis, the abstract path
 * name for the temporary resource should be based on this abstract path
 * name with the specified suffix appended and additional characters to make
 * the path unique (for example, sequence numbers), as required.
 * <p>
 * When WMQFTE transfers data to a destination it normally attempts to first
 * write to a temporary resource then on transfer completion renames the
 * temporary resource to the required destination. This method is called by
 * WMQFTE to create a new temporary resource path. The returned path should
 * be new and the resource should not previously exist.
 *
 * @param suffix
 *            Recommended suffix to use for the generated temporary path.
 *
 * @return A new {@link IOExitResourcePath} instance for the temporary
 *         resource path, that did not previously exist.
 * @throws RecoverableIOException
 *             If a recoverable problem occurs whilst attempting to create
 *             the temporary resource. This means that WMQFTE can attempt to
 *             recover the transfer.
 * @throws IOException
 *             If some other I/O problem occurs.
 */
IOExitResourcePath createTempPath(String suffix)
        throws RecoverableIOException, IOException;

/**
 * Opens a {@link IOExitChannel} instance for reading data from the resource
 * denoted by this abstract path. The current data byte position for the
 * resource is expected to be the passed position value, such that when
 * {@link IOExitChannel#read(java.nio.ByteBuffer)} is called, data starting
 * from that position is read.
 *
 * @param position
 *            The required data byte read position.
 * @return A new {@link IOExitChannel} instance allowing data to be read
 *         from the resource denoted by this abstract path.
 * @throws RecoverableIOException
 *             If a recoverable problem occurs while attempting to open the
 *             resource for reading. This means that WMQFTE can attempt to
 *             recover the transfer.
 * @throws IOException
 *             If some other I/O problem occurs.
 */
IOExitChannel openForRead(long position) throws RecoverableIOException,
        IOException;

/**
 * Opens a {@link IOExitChannel} instance for writing data to the resource
 * denoted by this abstract path. Writing of data, using the
 * {@link IOExitChannel#write(java.nio.ByteBuffer)} method, starts at either
 * the beginning of the resource or end of the current data for the
 * resource, depending on the specified append parameter.
 *
 * @param append
 *            When {@code true} indicates that data written to the resource
 *            should be appended to the end of the current data. When
 *            {@code false} indicates that writing of data is to start at
 *            the beginning of the resource; any existing data is lost.
 * @return A new {@link IOExitChannel} instance allowing data to be written
```

```
     *         to the resource denoted by this abstract path.
     * @throws RecoverableIOException
     *             If a recoverable problem occurs whilst attempting to open the
     *             resource for writing. This means that WMQFTE can attempt to
     *             recover the transfer.
     * @throws IOException
     *             If some other I/O problem occurs.
     */
    IOExitChannel openForWrite(boolean append) throws RecoverableIOException,
            IOException;

    /**
     * Tests if the resource denoted by this abstract path is in use by another
     * application. Typically, this is because another application has a lock on
     * the resource either for shared or exclusive access.
     *
     * @return {code true} if resource denoted by this abstract path is in use
     *         by another application, {@code false} otherwise.
     */
    boolean inUse();

    /**
     * Obtains a {@link IOExitProperties} instance for properties associated
     * with the resource denoted by this abstract path.
     * <p>
     * WMQFTE will read these properties to govern how a transfer behaves when
     * interacting with the resource.
     *
     * @return A {@link IOExitProperties} instance for properties associated
     *         with the resource denoted by this abstract path.
     */
    IOExitProperties getProperties();

}
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

Rozhraní IOExitWildcardPath.java

IOExitWildcardPath.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *   Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * Represents a path that denotes a wildcard. This can be used to match multiple
 * resource paths.
 */
public interface IOExitWildcardPath extends IOExitPath {
```

Související úlohy

Použití uživatelských procedur I/O přenosu MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

Rozhraní MonitorExit.java

MonitorExit.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *   Copyright IBM Corp. 2009, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing.  This interface defines a method that will be
 * invoked immediately prior to starting a task as the result of a monitor trigger
 */
public interface MonitorExit {

    /**
     * Invoked immediately prior to starting a task as the result of a monitor
     * trigger.
     *
     * @param environmentMetaData
     *            meta data about the environment in which the implementation
     *            of this method is running.  This information can only be read,
     *            it cannot be updated by the implementation.  The constant
     *            defined in <code>EnvironmentMetaDataConstants</code> class can
     *            be used to access the data held by this map.
     *
     * @param monitorMetaData
     *            meta data to associate with the monitor. The meta data passed
     *            to this method can be altered, and the changes will be
     *            reflected in subsequent exit routine invocations. This map
     *            also contains keys with IBM reserved names.  These entries are
     *            defined in the <code>MonitorMetaDataConstants</code> class and
     *            have special semantics. The the values of the IBM reserved names
     *            cannot be modified by the exit
     *
     * @param taskDetails
     *            An XML String representing the task to be executed as a result of
     *            the monitor triggering. This XML string may be modified by the
     *            exit
     *
     * @return    a monitor exit result object which is used to determine if the
     *            task should proceed, or be cancelled.
     */
    MonitorExitResult onMonitor(Map<String, String> environmentMetaData,
                                Map<String, String> monitorMetaData,
                                Reference<String> taskDetails);
}
```

Související úlohy

Monitorování prostředků MFT

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“SourceTransferStartExitrozhraní .java” na stránce 2156

“SourceTransferEndExit.java” na stránce 2155

“DestinationTransferStartExitrozhraní .java” na stránce 2131

“DestinationTransferEndExit.java” na stránce 2130

“ProtocolBridgeCredentialExit.java” na stránce 2150

ProtocolBridgeCredentialExit.java

ProtocolBridgeCredentialExit.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that are to be invoked as part of
 * user exit routine processing.  This interface defines methods that will
 * be invoked by a protocol bridge agent to map the MQ user ID of the transfer to credentials
 * that are to be used to access the protocol server.
 * There will be one instance of each implementation class per protocol bridge agent. The methods
 * can be called from different threads so the methods must be synchronized.
 */
public interface ProtocolBridgeCredentialExit {

    /**
     * Invoked once when a protocol bridge agent is started. It is intended to initialize
     * any resources that are required by the exit
     *
     * @param bridgeProperties
     *            The values of properties defined for the protocol bridge.
     *            These values can only be read, they cannot be updated by
     *            the implementation.
     *
     *
     * @return    true if the initialization is successful and false if unsuccessful
     *            If false is returned from an exit the protocol bridge agent will not
     *            start
     */

   public boolean initialize(final Map<String>  bridgeProperties);

      /**
      * Invoked once for each transfer to map the MQ user ID in the transfer message to the
      * credentials to be used to access the protocol server
       *
       * @param mqUserId The MQ user ID from which to map to the credentials to be used
      *                 access the protocol server
      * @return         A credential exit result object that contains the result of the map and
      *                 the credentials to use to access the protocol server
     */

   public CredentialExitResult mapMQUserId(final String mqUserId);

   /**
      * Invoked once when a protocol bridge agent is shutdown. It is intended to release
      * any resources that were allocated by the exit
      *
       * @param bridgeProperties
      *            The values of properties defined for the protocol bridge.
      *            These values can only be read, they cannot be updated by
       *            the implementation.
     *
       * @return
   */

   public void shutdown(final Map<String> bridgeProperties);

}
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Mapování pověření pro souborový server pomocí tříd ukončení

Rozhraní ProtocolBridgeCredentialExit2.java

ProtocolBridgeCredentialExit2.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * An interface that is implemented by classes that are invoked as part of user
 * exit routine processing. This interface defines methods that are invoked by a
 * protocol bridge agent to map the MQ user ID of the transfer to credentials
 * used to access a specified protocol bridge server. There will be one instance
 * of each implementation class for each protocol bridge agent. The methods can
 * be called from different threads so the methods must be synchronized.
 */
public interface ProtocolBridgeCredentialExit2 extends
        ProtocolBridgeCredentialExit {

    /**
     * Invoked once for each transfer to map the MQ user ID in the transfer
     * message to the credentials used to access a specified protocol server.
     *
     * @param endPoint
     *            Information that describes the protocol server to be accessed.
     * @param mqUserId
     *            The MQ user ID from which to map the credentials used to
     *            access the protocol server.
     * @return A {@link CredentialExitResult) instance that contains the result
     *         of the map and the credentials to use to access the protocol
     *         server.
     */
    public CredentialExitResult mapMQUserId(
            final ProtocolServerEndPoint endPoint, final String mqUserId);
}
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Mapování pověření pro souborový server pomocí tříd ukončení

Rozhraní ProtocolBridgePropertiesExit2.java

ProtocolBridgePropertiesExit2.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;
import java.util.Properties;
```

```
/**
 * An interface that is implemented by classes that are to be invoked as part of
 * user exit routine processing. This interface defines methods that will be
 * invoked by a protocol bridge agent to look up properties for protocol servers
 * that are referenced in transfers.
 * <p>
 * There will be one instance of each implementation class for each protocol
 * bridge agent. The methods can be called from different threads so the methods
 * must be synchronised.
 */
public interface ProtocolBridgePropertiesExit2 {

    /**
     * Invoked once when a protocol bridge agent is started. It is intended to
     * initialize any resources that are required by the exit.
     *
     * @param bridgeProperties
     *            The values of properties defined for the protocol bridge.
     *            These values can only be read, they cannot be updated by the
     *            implementation.
     * @return {@code true} if the initialization is successful and {@code
     *         false} if unsuccessful. If {@code false} is returned from an exit
     *         the protocol bridge agent will not start.
     */
    public boolean initialize(final Map<String, String> bridgeProperties);

          /**
     * Invoked when the Protocol Bridge needs to access the protocol bridge credentials XML file.
     *
     * @return a {@link String} object giving the location of the ProtocolBridgeCredentials.xml
     */
    public String getCredentialLocation ();

    /**
     * Obtains a set of properties for the specified protocol server name.
     * <p>
     * The returned {@link Properties} must contain entries with key names
     * corresponding to the constants defined in
     * {@link ProtocolServerPropertyConstants} and in particular must include an
     * entry for all appropriate constants described as required.
     *
     * @param protocolServerName
     *            The name of the protocol server whose properties are to be
     *            returned. If a null or a blank value is specified, properties
     *            for the default protocol server are to be returned.
     * @return The {@link Properties} for the specified protocol server, or null
     *         if the server cannot be found.
     */
    public Properties getProtocolServerProperties(
            final String protocolServerName);

    /**
     * Invoked once when a protocol bridge agent is shut down. It is intended to
     * release any resources that were allocated by the exit.
     *
     * @param bridgeProperties
     *            The values of properties defined for the protocol bridge.
     *            These values can only be read, they cannot be updated by the
     *            implementation.
     */
    public void shutdown(final Map<String, String> bridgeProperties);

}
```

Související úlohy

ProtocolBridgePropertiesExit: Hledání vlastností souborového serveru protokolu

Přizpůsobení produktu MFT pomocí uživatelských procedur

Mapování pověření pro souborový server pomocí tříd ukončení

SourceFileExitFileSpecification.java třída

SourceFileExitFileSpecification.java

```
/*
 *   Licensed Materials - Property of IBM
 *
```

```
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *   Copyright IBM Corp. 2012, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * A specification of the file names to use for a file transfer, as evaluated by the
 * agent acting as the source of the transfer.
 */
public final class SourceFileExitFileSpecification {

    private final String sourceFileSpecification;
    private final String destinationFileSpecification;
    private final Map<String, String> sourceFileMetaData;
    private final Map<String, String> destinationFileMetaData;

    /**
     * Constructor.  Creates a source file exit file specification.
     *
     * @param sourceFileSpecification
     *            the source file specification to associate with the source file
     *            exit file specification.
     *
     * @param destinationFileSpecification
     *            the destination file specification to associate with the
     *            source file exit file specification.
     *
     * @param sourceFileMetaData
     *            the source file meta data.
     *
     * @param destinationFileMetaData
     *            the destination file meta data   .
     */
    public SourceFileExitFileSpecification(final String sourceFileSpecification,
                                           final String destinationFileSpecification,
                                           final Map<String, String> sourceFileMetaData,
                                           final Map<String, String> destinationFileMetaData) {
        this.sourceFileSpecification = sourceFileSpecification;
        this.destinationFileSpecification = destinationFileSpecification;
        this.sourceFileMetaData = sourceFileMetaData;
        this.destinationFileMetaData = destinationFileMetaData;
    }

    /**
     * Returns the destination file specification.
     *
     * @return    the destination file specification.  This represents the location,
     *            on the agent acting as the destination for the transfer, where the
     *            file should be written.  Exit routines installed into the agent
     *            acting as the destination for the transfer may override this value.
     */
    public String getDestination() {
        return destinationFileSpecification;
    }

    /**
     * Returns the source file specification.
     *
     * @return    the source file specification.  This represents the location where
     *            the file data will be read from.
     */
    public String getSource() {
        return sourceFileSpecification;
    }

    /**
     * Returns the file meta data that relates to the source file specification.
     *
     * @return the file meta data that relates to the source file specification.
     */
    public Map<String, String> getSourceFileMetaData() {
        return sourceFileMetaData;
    }
```

```
    /**
     * Returns the file meta data that relates to the destination file specification.
     *
     * @return the file meta data that relates to the destination file specification.
     */
    public Map<String, String> getDestinationFileMetaData() {
        return destinationFileMetaData;
    }
}
```

Související pojmy

“Metadata pro uživatelské procedury MFT” na stránce 2117
Existují tři různé typy metadat, které lze dodat do uživatelských procedur pro produkt Managed File Transfer: prostředí, přenos a metadata souboru. Tato metadata jsou prezentována jako mapy dvojic klíč-hodnota Java .

SourceTransferEndExit.java

SourceTransferEndExit.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing.  This interface defines a method that will be
 * invoked immediately after completing a transfer on the agent acting as the
 * source of the transfer.
 */
public interface SourceTransferEndExit {

    /**
     * Invoked immediately after the completion of a transfer on the agent acting as
     * the source of the transfer.
     *
     * @param transferExitResult
     *            a result object reflecting whether or not the transfer completed
     *            successfully.
     *
     * @param sourceAgentName
     *            the name of the agent acting as the source of the transfer.
     *            This is the name of the agent that the implementation of this
     *            method will be invoked from.
     *
     * @param destinationAgentName
     *            the name of the agent acting as the destination of the
     *            transfer.
     *
     * @param environmentMetaData
     *            meta data about the environment in which the implementation
     *            of this method is running.  This information can only be read,
     *            it cannot be updated by the implementation.  The constants
     *            defined in <code>EnvironmentMetaDataConstants</code> class can
     *            be used to access the data held by this map.
     *
     * @param transferMetaData
     *            meta data to associate with the transfer.  The information can
     *            only be read, it cannot be updated by the implementation.  This
     *            map may also contain keys with IBM reserved names.  These
     *            entries are defined in the <code>TransferMetaDataConstants</code>
     *            class and have special semantics.
     *
     * @param fileResults
```

```
     *            a list of file transfer result objects that describe the source
     *            file name, destination file name and result of each file transfer
     *            operation attempted.
     *
     * @return    an optional description to enter into the log message describing
     *            transfer completion.  A value of <code>null</code> can be used
     *            when no description is required.
     */
    String onSourceTransferEnd(TransferExitResult transferExitResult,
                   String sourceAgentName,
                   String destinationAgentName,
                   Map<String, String>environmentMetaData,
                   Map<String, String>transferMetaData,
                   List<FileTransferResult>fileResults);


}
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“SourceTransferStartExitrozhraní .java” na stránce 2156

“DestinationTransferStartExitrozhraní .java” na stránce 2131

“DestinationTransferEndExit.java” na stránce 2130

“Rozhraní MonitorExit.java” na stránce 2149

“ProtocolBridgeCredentialExit.java” na stránce 2150

SourceTransferStartExitrozhraní .java

SourceTransferStartExit.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

import java.util.List;
import java.util.Map;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing.  This interface defines a method that will be
 * invoked immediately prior to starting a transfer on the agent acting as the
 * source of the transfer.
 */
public interface SourceTransferStartExit {

    /**
     * Invoked immediately prior to starting a transfer on the agent acting as
     * the source of the transfer.
     *
     * @param sourceAgentName
     *            the name of the agent acting as the source of the transfer.
     *            This is the name of the agent that the implementation of this
     *            method will be invoked from.
     *
     * @param destinationAgentName
     *            the name of the agent acting as the destination of the
     *            transfer.
     *
     * @param environmentMetaData
     *            meta data about the environment in which the implementation
```

```
     *            of this method is running.  This information can only be read,
     *            it cannot be updated by the implementation.  The constants
     *            defined in <code>EnvironmentMetaDataConstants</code> class can
     *            be used to access the data held by this map.
     *
     * @param transferMetaData
     *            meta data to associate with the transfer.  The meta data passed
     *            to this method can be altered, and the changes to will be
     *            reflected in subsequent exit routine invocations.  This map may
     *            also contain keys with IBM reserved names.  These entries are
     *            defined in the <code>TransferMetaDataConstants</code> class and
     *            have special semantics.
     *
     * @param fileSpecs
     *            a list of file specifications that govern the file data to
     *            transfer.  The implementation of this method can add entries,
     *            remove entries, or modify entries in this list and the changes
     *            will be reflected in the files transferred.
     *
     * @return    a transfer exit result object which is used to determine if the
     *            transfer should proceed, or be cancelled.
     */
    TransferExitResult onSourceTransferStart(String sourceAgentName,
                         String destinationAgentName,
                         Map<String, String> environmentMetaData,
                         Map<String, String>transferMetaData,
                         List<SourceFileExitFileSpecification>fileSpecs);
}
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“SourceFileExitFileSpecification.java třída” na stránce 2153

“SourceTransferEndExit.java” na stránce 2155

“DestinationTransferStartExitrozhraní .java” na stránce 2131

“DestinationTransferEndExit.java” na stránce 2130

“Rozhraní MonitorExit.java” na stránce 2149

“ProtocolBridgeCredentialExit.java” na stránce 2150

Rozhraní TransferExitResult.java

TransferExitResult.java

```
/*
 *   Licensed Materials - Property of IBM
 *
 *   "Restricted Materials of IBM"
 *
 *   5724-H72
 *
 *    Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 *   US Government Users Restricted Rights - Use, duplication or
 *   disclosure restricted by GSA ADP Schedule Contract with
 *   IBM Corp.
 */

package com.ibm.wmqfte.exitroutine.api;

/**
 * The result of invoking a transfer exit routine.  It is composed of a result
 * code, which determines if the transfer should proceed, and an optional explanatory
 * message.  The explanation, if present, is entered into the log message.
 */
public class TransferExitResult {

    private final TransferExitResultCode resultCode;
    private final String explanation;

    /**
     * For convenience, a static "proceed" result with no associated explanation
```

```
     * message.
     */
    public static final TransferExitResult PROCEED_RESULT =
        new TransferExitResult(TransferExitResultCode.PROCEED, null);

    /**
     * Constructor.  Creates a transfer exit result object with a specified result
     * code and explanation.
     *
     * @param resultCode
     *            The result code to associate with the exit result being created.
     *
     * @param explanation
     *            The explanation to associate with the exit result being created.
     *            A value of <code>null</code> can be specified to indicate no
     *            explanation.
     */
    public TransferExitResult(TransferExitResultCode resultCode, String explanation) {
        this.resultCode = resultCode;
        this.explanation = explanation;
    }

    /**
     * Returns the explanation associated with this transfer exit result.
     *
     * @return    the explanation associated with this exit result.
     */
    public String getExplanation() {
        return explanation;
    }

    /**
     * Returns the result code associated with this transfer exit result.
     *
     * @return    the result code associated with this exit result.
     */
    public TransferExitResultCode getResultCode() {
        return resultCode;
    }
}
```

Související úlohy

Přizpůsobení produktu MFT pomocí uživatelských procedur

Související odkazy

“SourceTransferStartExitrozhraní .java” na stránce 2156

“DestinationTransferStartExitrozhraní .java” na stránce 2131

“DestinationTransferEndExit.java” na stránce 2130

“Rozhraní MonitorExit.java” na stránce 2149

“ProtocolBridgeCredentialExit.java” na stránce 2150

Formáty zpráv pro zprávy, které můžete vložit do fronty příkazů agenta MFT

Tato schémata XML definují formáty pro zprávy, které lze vložit do fronty příkazů agenta, aby požadovaly, aby agent provedl akci. Zprávu XML lze umístit do fronty příkazů agenta pomocí příkazů příkazového řádku nebo pomocí aplikace.

- Formát zprávy požadavku na přenos souborů
- Formát zpráv požadavku monitoru MFT
- Formát zprávy požadavku agenta MFT příkazu ping
- Formát zprávy odpovědi agenta MFT

Odkaz na systém zpráv REST API

Referenční informace o messaging REST API.

Další informace o použití konzoly messaging REST APInaleznete v tématu Systém zpráv pomocí konzoly REST API.

REST API - prostředky

Tato kolekce témat poskytuje referenční informace pro každý z prostředků messaging REST API .

Další informace o použití konzoly messaging REST APInaleznete v tématu Systém zpráv pomocí konzoly REST API.

/messaging/qmgr/{qmgrName}/queue/{queueName}/message

Rozhraní REST API systému zpráv umožňuje vložení zpráv do fronty nebo jejich procházení či destruktivní získávání z fronty pomocí prostředku `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` .

POST /messaging/qmgr/{qmgrName}/queue/{queueName}/message

Pomocí metody HTTP POST s prostředkem `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` můžete vkládat zprávy do určené fronty v určeném správci front.

Vloží zprávu IBM MQ obsahující tělo požadavku HTTP do určeného správce front a fronty. Správce front musí být na stejném počítači jako server mqweb. Metoda podporuje pouze textová těla požadavků HTTP . Zprávy jsou odesílány jako zprávy ve formátu `MQSTR` nebo `JMS TextMessage` a jsou vkládány s použitím aktuálního uživatelského kontextu.

V 9.3.0 Rozhraní REST API V3 přidává schopnost určit vlastnosti zprávy definované uživatelem a zahrnout prioritu zprávy. Záhlaví požadavků ibm-mq-md-priority a ibm-mq-usr jsou k dispozici pouze s rozhraním REST API V3. Záhlaví požadavku ibm-mq-md-correlationId má jiný formát v rozhraní REST API V3. Záhlaví může být ID specifické pro aplikaci, nebo v případě kódovaného řetězce vyžaduje předponu `ID:` . Pokud váš požadavek POST obsahuje zprávy definované uživatelem nebo ID korelace specifické pro aplikaci, bude zpráva formátována jako `JMS TextMessage`.

URL prostředku

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

V 9.3.0 `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

qmgrName
Určuje název správce front, ke kterému se má připojit pro systém zpráv. Správce front musí být na stejném počítači jako server mqweb.

V 9.3.3 V produktu IBM MQ 9.3.3se můžete připojit buď k lokálnímu správci front, nebo ke vzdálenému správci front. Název, který zadáte pro parametr **qmgrName** , závisí na konfiguraci serveru mqweb:

- Je-li váš server mqweb konfigurován tak, aby se připojoval pouze k lokálním správcům front, použijte název správce front.
- Je-li váš server mqweb konfigurován pro připojení vzdálených správců front, použijte jedinečný název správce front určený v informacích o připojení vzdáleného správce front.

Pomocí příkazu **dspmqweb properties** a zobrazením vlastnosti **mqRestMessagingConnectionMode** můžete určit, zda je váš server mqweb konfigurován pro připojení k lokálním správcům front nebo vzdáleným správcům front.

V názvu správce front se rozlišují velká a malá písmena.

Pokud název správce front obsahuje dopředné lomítko, tečku nebo znak procent, musí být tyto znaky zakódovány pomocí adresy URL:

- Dopředné lomítko musí být zakódováno jako %2F.
- Tečka musí být zakódována jako %2E.
- Znak procenta musí být zakódován jako %25.

queueName
Určuje název fronty, do které má být zpráva vložena.

Fronta musí být definována jako lokální, vzdálená nebo alias pro určeného správce front-může také odkazovat na klastrovanou frontu.

V názvu fronty se rozlišují malá a velká písmena.

Pokud název fronty obsahuje dopředné lomítko nebo znak procenta, musí být tyto znaky URL zakódovány:

- Dopředné lomítko,/, musí být zakódováno jako %2F.
- Znak procenta% musí být zakódován jako %25.

Můžete použít HTTP místo HTTPS , pokud povolíte připojení HTTP . Další informace o povolení HTTPnaleznete v tématu Konfigurace portů HTTP a HTTPS.

Záhlaví požadavku

S požadavkem musí být odeslána následující záhlaví:

Autorizace
Toto záhlaví musí být odesláno, pokud používáte základní ověření. Další informace viz Použití základního ověření HTTP pomocí rozhraní REST API.

Content-Type
Toto záhlaví musí být odesláno s jednou z následujících hodnot:

- `text/plain;charset=utf-8`
- `text/html;charset=utf-8`
- `text/xml;charset=utf-8`
- `application/json;charset=utf-8`
- `application/xml;charset=utf-8`

ibm-mq-rest-csrf-token
Toto záhlaví musí být nastaveno, ale hodnota může být libovolná, včetně prázdné hodnoty.

Následující záhlaví lze volitelně odeslat s požadavkem:

Accept-Language
Toto záhlaví určuje požadovaný jazyk pro všechny výjimky nebo chybové zprávy vrácené v těle zprávy odpovědi.

REST API V1 REST API V2 **ibm-mq-md-correlationId**

Toto záhlaví nastavuje ID korelace vytvořené zprávy. Záhlaví je určeno jako hexadecimální řetězec o délce 48 znaků, který představuje 24 bajtů. Nepřipojujte k hodnotě předponu `"ID:"`, rozhraní REST API přidá tento řetězec automaticky.

Například: `ibm-mq-md-correlationId: 414d5120514d4144455620202020202067d8bf5923582e02`

REST API V3 V 9.3.0 **ibm-mq-md-correlationId**

Toto záhlaví nastavuje ID korelace vytvořené zprávy. ID korelace může mít jednu z následujících forem:

- 48znakový hexadecimální kódovaný řetězec, představující 24 bajtů, s předponou s řetězcem `"ID:"`. Příklad: `ibm-mq-md-correlationId: ID:414d5120514d4144455620202020202067d8bf5923582e02`
- Hodnota specifická pro aplikaci. Hodnota je řetězec specifický pro aplikaci: `ibm-mq-md-correlationId: My-Custom-CorrelId`

  Zadáte-li tento formát ID korelace, cíl zprávy bude cílen jako WMQ_CLIENT_JMS_COMPLIANT, a bude tedy obsahovat záhlaví MQRFH2 .

ibm-mq-md-vypršení platnosti

Toto záhlaví nastavuje dobu trvání vypršení platnosti pro vytvořenou zprávu. Vypršení platnosti zprávy začíná od okamžiku, kdy zpráva dorazí do fronty. V důsledku toho je latence sítě ignorována. Záhlaví musí být uvedeno jako jedna z následujících hodnot:

bez omezení
Platnost zprávy nevyprší.

Tato hodnota je výchozí hodnota.

Celočíselná hodnota
Milisekundy před vypršením platnosti zprávy.

Omezeno na rozsah 0-99999999900.

ibm-mq-md-persistence-trvalost

Toto záhlaví nastavuje perzistenci pro vytvořenou zprávu. Záhlaví je určeno jako jedna z následujících hodnot:

nonPersistent
Zpráva nepřežije selhání systému nebo restartování správce front.

Tato hodnota je výchozí hodnota.

Trvalý
Zpráva přežije selhání systému nebo restartuje správce front.

REST API V3 V 9.3.0 **ibm-mq-md-priorita**

Pro rozhraní REST API V3toto záhlaví nastavuje prioritu vytvořené zprávy. Záhlaví je určeno jako jedna z následujících hodnot:

asDestination
Zpráva používá prioritu uvedenou v atributu DEFPRTY základního objektu fronty IBM MQ .

Tato hodnota je výchozí hodnota.

Celočíselná hodnota
Určete skutečnou prioritu jako celé číslo v rozsahu 0-9.

Například: `ibm-mq-md-priority: asDestination`

Pro rozhraní REST API V1 a rozhraní REST API V2je priorita zprávy pro POST vždy 4.

ibm-mq-md-replyTo

Toto záhlaví nastaví cíl odpovědi pro vytvořenou zprávu. Formát záhlaví používá standardní notaci zadání fronty pro odpověď a volitelného správce front: `replyQueue[@replyQmgr]`

Například: `ibm-mq-md-replyTo: myReplyQueue@myReplyQMgr`

REST API V3 V 9.3.0 **ibm-mq-usr**

Toto záhlaví nastavuje vlastnosti definované uživatelem pro zprávu požadavku.

Vlastnosti mají následující syntaxi:

```
ibm-mq-usr: property_name; user_value; user_type
```

property_name
Název zadané uživatelské vlastnosti. Musí se jednat o platný název vlastnosti JMS.

hodnota uživatele
Hodnota vlastnosti.

typ_uživatele
Typ vlastnosti:

- `boolean` (true/false, MQBOOL)
- `byte` (8bitové celé číslo, MQINT8)
- `short` (16bitové celé číslo, MQINT16)
- `integer` (32bitové celé číslo, MQINT32)
- `long` (64bitové celé číslo, MQINT64)
- `float` (32bitová realita, MQFLOAT32)
- `double` (64bitová realita, MQFLOAT64)
- `string` (řetězec v uvozovkách)

Pro zprávu můžete nastavit více vlastností. Můžete buď zadat více vlastností oddělených čárkami v jednom záhlaví požadavku ibm-mq-usr, nebo můžete použít dvě nebo více samostatných instancí záhlaví požadavku ibm-mq-usr.

Můžete například nastavit více vlastností definovaných uživatelem v jednom záhlaví: `ibm-mq-usr: userPropertyA;5;byte,userPropertyB;-10;integer`

Nebo můžete nastavit více vlastností definovaných uživatelem v oddělených instancích záhlaví:`ibm-mq-usr: userPropertyA;5;byte ibm-mq-usr: userPropertyB;-10;integer`

Formát těla požadavku

Tělo požadavku musí být textové a musí používat kódování UTF-8 . Není vyžadována žádná specifická struktura textu. Vytvoří se zpráva ve formátu `MQSTR` obsahující text těla požadavku a vloží se do uvedené fronty.

REST API V3 V 9.3.0 Pokud jsou použity uživatelem definované vlastnosti rozhraní REST API V3 nebo funkce ID korelace specifické pro aplikaci, pak se vytvoří zpráva ve formátu `JMS TextMessage` obsahující text těla požadavku a vloží se do uvedené fronty.

Další informace viz příklady.

Požadavky na zabezpečení

Volající musí být ověřen na serveru mqweb. Role `MQWebAdmin` a `MQWebAdminRO` nelze použít pro messaging REST API. Další informace o zabezpečení REST APInaleznete v tématu Zabezpečení produktů IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak administrative REST API.

Činiteli zabezpečení volajícího musí být udělena schopnost vkládat zprávy do určené fronty:

- Fronta, která je zadána v části *{queueName}* adresy URL prostředku, musí být povolena v produktu PUT.
- MQ Appliance ALW Pro frontu určenou částí *{queueName}* prostředku URLmusí být činiteli zabezpečení volajícího uděleno oprávnění +PUT .

- z/OS Pro frontu, která je určena částí *{queueName}* prostředku URL, musí být přístup UPDATE udělen činiteli zabezpečení volajícího.

Tímto činitelem zabezpečení může být uživatel, který spustil server mqweb, nebo uživatel, který je přihlášen k serveru mqweb. Pokud je správcem front, ke kterému se připojujete, vzdálený správce front, může být namísto toho činitelem zabezpečení uživatel zadaný pomocí pověření v informacích o připojení vzdáleného správce front nebo jiný uživatel určený pomocí pravidel zabezpečení kanálu. Další informace naleznete v tématu Určení činitele zabezpečení používaného produktem messaging REST API.

ALW Na systému AIX, Linux, and Windows můžete udělit oprávnění k činitelům zabezpečení, aby mohli používat prostředky IBM MQ, pomocí příkazu **setmqaut**. Další informace viz téma **setmqaut** (udělit nebo odvolat oprávnění).

z/OS V systému z/OS viz Nastavení zabezpečení v systému z/OS.

Používáte-li rozšířené zabezpečení zpráv (AMS) s produktem messaging REST API, mějte na paměti, že všechny zprávy jsou šifrovány pomocí kontextu serveru mqweb, nikoli kontextu uživatele, který zprávu zveřejňuje.

Stavové kódy odezvy

201
Zpráva byla úspěšně vytvořena a odeslána.

400
Byla zadána neplatná data.

Byla například zadána neplatná hodnota záhlaví požadavku.

401
Neověřeno.

Volající musí být ověřen na serveru mqweb a musí být členem nejméně jedné z rolí `MQWebAdmin`, `MQWebAdminRO` nebo `MQWebUser`. Musí být také uvedeno záhlaví `ibm-mq-rest-csrf-token` . Další informace viz “Požadavky na zabezpečení” na stránce 2162.

403
Neautorizováno.

Volající je ověřen na serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem nebo k podmnožině požadovaných prostředků IBM MQ nebo není v roli `MQWebUser` . Další informace o požadovaném přístupu viz “Požadavky na zabezpečení” na stránce 2162.

404
Fronta neexistuje.

405
Fronta je blokována PUT.

415
Záhlaví nebo tělo zprávy je nepodporovaný typ média.

Například záhlaví `Content-Type` je nastaveno na nepodporovaný typ média.

500
Problém se serverem nebo kód chyby z IBM MQ.

502
Aktuální činitel zabezpečení nemůže odeslat zprávu, protože poskytovatel systému zpráv nepodporuje požadovanou funkci. Například, pokud je cesta ke třídě serveru mqweb neplatná.

503
Správce front není spuštěn.

Záhlaví odezvy

S odezvou jsou vrácena následující záhlaví:

Obsah-Jazyk
Určuje identifikátor jazyka zprávy odpovědi v případě chyb nebo výjimek. Používá se ve spojení se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakékoli chybové stavy nebo podmínky výjimky. Výchozí hodnota serveru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP , a to i v případě, že není k dispozici žádný obsah. Při úspěchu je hodnota nula.

Content-Type
Určuje typ těla odezvy. Při úspěchu je hodnota `text/plain;charset=utf-8`. V případě chyb nebo výjimek je hodnota `application/json;charset=utf-8`.

REST API V1 REST API V2 **ibm-mq-md-messageId**
Uvádí ID zprávy, které je přiděleno IBM MQ této zprávě. Stejně jako záhlaví požadavku `ibm-mq-md-correlationId` je reprezentováno 48znakovým hexadecimálním kódovaným řetězcem, který představuje 24 bajtů.

Příklad:

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

REST API V3 V 9.3.0 **ibm-mq-md-messageId**
Uvádí ID zprávy, které je přiděleno IBM MQ této zprávě. Stejně jako záhlaví požadavku `ibm-mq-md-correlationId` je reprezentováno jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů, s předponou řetězce `ID:`.

Příklad:

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

V 9.3.3 **ibm-mq-vyřešeno-qmgr**
Určuje název správce front, který byl použit pro požadavek. Pokud byl v prostředku URLpoužit jedinečný název, toto záhlaví identifikuje název správce front, který je k tomuto jedinečnému názvu přidružen. Pokud jedinečný název použitý v prostředku URL odkazuje na skupinu správců front, toto záhlaví identifikuje, který správce front v rámci skupiny byl použit.

Formát těla odezvy

Tělo odezvy je prázdné, pokud je zpráva úspěšně odeslána. Pokud dojde k chybě, tělo odezvy obsahuje chybovou zprávu. Další informace viz REST API ošetření chyb.

Příklady

Následující příklady používají adresu URL prostředku v2. Používáte-li starší verzi produktu IBM MQ než IBM MQ 9.1.5, musíte místo toho použít adresu URL prostředku v1. To znamená, že v adrese URL prostředku nahradíte `v1` tam, kde příklad adresy URL používá `v2`.

Následující příklad protokoluje uživatele s názvem `mquser` s heslem `mquser`. V cURLmůže požadavek na přihlášení vypadat jako v následujícím příkladu Windows . Token LTPA je uložen v souboru `cookiejar.txt` pomocí příznaku `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v2/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"mquser\",\"password\":\"mquser\"}"
-c c:\cookiejar.txt
```

Po přihlášení uživatele se token LTPA a záhlaví `ibm-mq-rest-csrf-token` HTTP použijí k ověření dalších požadavků. `ibm-mq-rest-csrf-token` `token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- Následující příklad Windows cURL odešle zprávu do fronty Q1 ve správci front QM1s použitím výchozích voleb. Zpráva obsahuje text *"Ahoj světe!"*:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain;charset=utf-8" --data "Hello World!"
```

- Následující příklad Windows cURL odešle trvalou zprávu do fronty Q1 ve správci front QM1s vypršením 2 minut. Zpráva obsahuje text *"Ahoj světe!"*:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain;charset=utf-8" -H "ibm-mq-md-persistence: persistent"
-H "ibm-mq-md-expiry: 120000" --data "Hello World!"
```

- Následující příklad Windows cURL odešle dočasnou zprávu do fronty Q1 ve správci front QM1bez vypršení platnosti a definovaného ID korelace. Zpráva obsahuje text *"Ahoj světe!"*:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Content-Type: text/plain;charset=utf-8" -H "ibm-mq-md-persistence: nonPersistent"
-H "ibm-mq-md-expiry: unlimited" -H "ibm-mq-md-correlationId:
414d5120514d4144455620202020202067d8b
f5923582e02" --data "Hello World!"
```

GET /messaging/qmgr/{qmgrName}/queue/{queueName}/message

Pomocí metody HTTP GET s prostředkem `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` můžete procházet zprávy z přidruženého správce front a fronty.

Prochází první dostupnou zprávu z určeného správce front a fronty. Správce front musí být na stejném počítači jako server mqweb. Tělo zprávy je vráceno v těle odezvy HTTP . Zpráva musí mít formát `MQSTR` nebo `JMS TextMessage` a je přijata s použitím aktuálního uživatelského kontextu.

Všechny zprávy jsou ponechány ve frontě a volajícímu je vrácen odpovídající stavový kód pro případné nevhodné zprávy. Například zpráva, která nemá formát `MQSTR` nebo `JMS TextMessage` .

V 9.3.0 Rozhraní REST API V3 přidává schopnost určit vlastnosti zpráv definované uživatelem a zahrnout prioritu zpráv se zprávami. Záhlaví odezvy ibm-mq-md-priority a ibm-mq-usr jsou k dispozici pouze s rozhraním REST API V3. Záhlaví požadavku ibm-mq-md-correlationId má jiný formát v rozhraní REST API V3. Záhlaví může být ID specifické pro aplikaci, nebo v případě kódovaného řetězce si zachová předponu `ID:` . Záhlaví odezvy ibm-mq-md-messageId a parametr dotazu mají jiný formát v rozhraní REST API V3, uchová předponu `ID:` .

- "URL prostředku" na stránce 2165
- "Volitelné parametry dotazu" na stránce 2166
- "Záhlaví požadavku" na stránce 2167
- "Formát těla požadavku" na stránce 2167
- "Požadavky na zabezpečení" na stránce 2167
- "Stavové kódy odezvy" na stránce 2168
- "Záhlaví odezvy" na stránce 2169
- "Formát těla odezvy" na stránce 2171
- "Příklady" na stránce 2171

URL prostředku

`https://`*`host`*`:`*`port`*`/ibmmq/rest/v1/messaging/qmgr/`*`{qmgrName}`*`/queue/`*`{queueName}`*`/message`

`https://`*`host`*`:`*`port`*`/ibmmq/rest/v2/messaging/qmgr/`*`{qmgrName}`*`/queue/`*`{queueName}`*`/message`

V 9.3.0 `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

qmgrName
Určuje název správce front, ke kterému se má připojit pro systém zpráv. Správce front musí být na stejném počítači jako server mqweb.

V 9.3.3 V produktu IBM MQ 9.3.3se můžete připojit buď k lokálnímu správci front, nebo ke vzdálenému správci front. Název, který zadáte pro parametr **qmgrName** , závisí na konfiguraci serveru mqweb:

- Je-li váš server mqweb konfigurován tak, aby se připojoval pouze k lokálním správcům front, použijte název správce front.
- Je-li váš server mqweb konfigurován pro připojení vzdálených správců front, použijte jedinečný název správce front určený v informacích o připojení vzdáleného správce front.

Pomocí příkazu **dspmqweb properties** a zobrazením vlastnosti **mqRestMessagingConnectionMode** můžete určit, zda je váš server mqweb konfigurován pro připojení k lokálním správcům front nebo vzdáleným správcům front.

V názvu správce front se rozlišují velká a malá písmena.

Pokud název správce front obsahuje dopředné lomítko, tečku nebo znak procent, musí být tyto znaky zakódovány pomocí adresy URL:

- Dopředné lomítko (/) musí být zakódováno jako %2F.
- Znaménko procent (%) musí být zkódováno jako %25.

queueName
Určuje název fronty, ze které má být zpráva procházena.

Fronta musí být definována jako lokální nebo alias, který odkazuje na lokální frontu.

V názvu fronty se rozlišují malá a velká písmena.

Pokud název fronty obsahuje dopředné lomítko nebo znak procenta, musí být tyto znaky URL zakódovány:

- Dopředné lomítko,/, musí být zakódováno jako %2F.
- Znak procenta% musí být zakódován jako %25.

Můžete použít HTTP místo HTTPS , pokud povolíte připojení HTTP . Další informace o povolení HTTPnaleznete v tématu Konfigurace portů HTTP a HTTPS.

Volitelné parametry dotazu

REST API V1 REST API V2 **correlationId=*hexValue***
Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID korelace.

hexValue
Parametr dotazu musí být uveden jako hexadecimální řetězec o délce 48 znaků, který představuje 24 bajtů.

Příklad:

```
../message?correlationId=414d5120514d414445562020202020202067d8bf5923582e02
```

REST API V3 V 9.3.0 **correlationId= ID:*hexValue* nebo correlationId=*application_specific_value***
Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID korelace.

hexValue
Parametr dotazu musí být uveden jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů a předchází mu řetězec `"ID:"`.

Příklad:

```
../message?correlationId=ID:414d5120514d4144455620202020202067d8bf5923582e02
```

specifická_hodnota aplikace
Parametr dotazu lze zadat jako řetězec specifický pro aplikaci.

Příklad:

```
../message?correlationId=My-Custom-CorrelId
```

REST API V1 REST API V2 **messageId=*hexValue***
Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID zprávy.

hexValue
Parametr dotazu musí být uveden jako hexadecimální řetězec o délce 48 znaků, který představuje 24 bajtů.

Příklad:

```
../message?messageId=414d5120514d4144455620202020202067d8ce5923582f07
```

REST API V3 V 9.3.0 **messageId= ID:*hexValue***
Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID zprávy.

hexValue
Parametr dotazu musí být uveden jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů a předchází mu řetězec `"ID:"`.

Příklad:

```
../message?messageId=ID:414d5120514d4144455620202020202067d8ce5923582f07
```

Záhlaví požadavku

S požadavkem musí být odeslána následující záhlaví:

Autorizace
Toto záhlaví musí být odesláno, pokud používáte základní ověření. Další informace viz Použití základního ověření HTTP pomocí rozhraní REST API.

ibm-mq-rest-csrf-token
Toto záhlaví musí být nastaveno, ale hodnota může být libovolná, včetně prázdné hodnoty.

Následující záhlaví lze volitelně odeslat s požadavkem:

Přijmout-znaková sada
Toto záhlaví lze použít k označení, která znaková sada je pro odezvu přijatelná. Je-li uvedeno, toto záhlaví musí být nastaveno jako UTF-8.

Accept-Language
Toto záhlaví určuje požadovaný jazyk pro všechny výjimky nebo chybové zprávy vrácené v těle zprávy odpovědi.

Formát těla požadavku

Není.

Požadavky na zabezpečení

Volající musí být ověřen na serveru mqweb. Role `MQWebAdmin` a `MQWebAdminRO` nelze použít pro messaging REST API. Další informace o zabezpečení REST APInaleznete v tématu Zabezpečení produktů IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak administrative REST API.

Činitel zabezpečení volajícího musí mít udělenu schopnost procházet zprávy z určené fronty:

- Fronta, která je zadána v části *{queueName}* adresy URL prostředku, musí být povolena v produktu BROWSE.
- ALW MQ Appliance Pro frontu určenou v části *{queueName}* adresy URL prostředku musí být činiteli zabezpečení volajícího uděleno oprávnění +GET, +INQ a +BROWSE.
- z/OS Pro frontu, která je zadána částí *{queueName}* adresy URL prostředku, UPDATE, musí být udělen přístup k činiteli zabezpečení volajícího.

Tímto činitelem zabezpečení může být uživatel, který spustil server mqweb, nebo uživatel, který je přihlášen k serveru mqweb. Pokud je správcem front, ke kterému se připojujete, vzdálený správce front, může být namísto toho činitelem zabezpečení uživatel zadaný pomocí pověření v informacích o připojení vzdáleného správce front nebo jiný uživatel určený pomocí pravidel zabezpečení kanálu. Další informace naleznete v tématu Určení činitele zabezpečení používaného produktem messaging REST API.

ALW Na systému AIX, Linux, and Windows můžete udělit oprávnění k činitelům zabezpečení, aby mohli používat prostředky IBM MQ, pomocí příkazu **setmqaut**. Další informace viz téma **setmqaut** (udělit nebo odvolat oprávnění).

z/OS V systému z/OS viz Nastavení zabezpečení v systému z/OS.

Stavové kódy odezvy

200
Zpráva byla úspěšně přijata.

204
Nejsou k dispozici žádné zprávy.

400
Byla zadána neplatná data.

Byla například zadána neplatná hodnota parametru dotazu.

401
Neověřeno.

Volající musí být ověřen na serveru mqweb a musí být členem nejméně jedné z rolí MQWebAdmin, MQWebAdminRO nebo MQWebUser. Musí být také uvedeno záhlaví ibm-mq-rest-csrf-token . Další informace viz “Požadavky na zabezpečení” na stránce 2167.

403
Neautorizováno.

Volající je ověřen na serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem nebo k podmnožině požadovaných prostředků IBM MQ nebo není v roli MQWebUser . Další informace o požadovaném přístupu viz “Požadavky na zabezpečení” na stránce 2167.

404
Fronta neexistuje.

500
Problém se serverem nebo kód chyby z IBM MQ.

501
Odpověď HTTP nelze sestavit.

Například přijatá zpráva má nesprávný typ nebo má správný typ, ale tělo nelze zpracovat.

502
Aktuální činitel zabezpečení nemůže přijmout zprávu, protože poskytovatel systému zpráv nepodporuje požadovanou funkci. Například, pokud je cesta ke třídě serveru mqweb neplatná.

503
Správce front není spuštěn.

Záhlaví odezvy

S odezvou jsou vrácena následující záhlaví:

Obsah-Jazyk
Určuje identifikátor jazyka zprávy odpovědi v případě chyb nebo výjimek. Používá se ve spojení se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakékoli chybové stavy nebo podmínky výjimky. Výchozí hodnota serveru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP , a to i v případě, že není k dispozici žádný obsah. Hodnota obsahuje délku (bajty) dat zprávy.

Content-Type
Určuje typ obsahu vráceného v těle odezvy přijaté zprávy. Při úspěchu je hodnota `text/plain;charset=utf-8`. V případě chyb nebo výjimek je hodnota `application/json;charset=utf-8`.

REST API V1 REST API V2 **ibm-mq-md-correlationId**
Určuje ID korelace přijaté zprávy. Záhlaví je vráceno, pokud přijatá zpráva obsahuje platné ID korelace. Je reprezentován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů.

Příklad:

```
ibm-mq-md-correlationId: 414d5120514d4144455620202020202067d8bf5923582e02
```

REST API V3 V 9.3.0 **ibm-mq-md-correlationId**
Určuje ID korelace přijaté zprávy. Záhlaví je vráceno, pokud přijatá zpráva obsahuje platné ID korelace. ID korelace může mít jednu z následujících forem:

- 48znakový hexadecimální kódovaný řetězec, představující 24 bajtů, s předponou s řetězcem `"ID:"`. Příklad:

  ```
  ibm-mq-md-correlationId: ID:414d5120514d4144455620202020202067d8bf5923582e02
  ```

- Hodnota specifická pro aplikaci. Hodnota je řetězec specifický pro aplikaci:

  ```
  ibm-mq-md-correlationId: My-Custom-CorrelId
  ```

ibm-mq-md-vypršení platnosti
Určuje zbývající dobu trvání vypršení platnosti přijaté zprávy. Záhlaví může mít jednu z následujících hodnot:

bez omezení
Platnost zprávy nevyprší.

Celočíselná hodnota
Zbývající milisekundy před vypršením platnosti zprávy.

REST API V1 REST API V2 **ibm-mq-md-messageId**
Uvádí ID zprávy, které je přiděleno IBM MQ této zprávě. Stejně jako záhlaví `ibm-mq-md-correlationId` je reprezentováno jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů.

Příklad:

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

REST API V3 V 9.3.0 **ibm-mq-md-messageId**

Uvádí ID zprávy, které je přiděleno IBM MQ této zprávě. Stejně jako záhlaví `ibm-mq-md-correlationId` je reprezentován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů, s předponou s řetězcem `"ID:"`

Příklad:

```
ibm-mq-md-messageId: ID:414d5120514d414445562020202020202067d8ce5923582f07
```

ibm-mq-md-persistence-trvalost

Určuje trvání přijaté zprávy. Záhlaví může mít jednu z následujících hodnot:

nonPersistent
Zpráva nepřežije selhání systému nebo restartování správce front.

Trvalý
Zpráva přežije selhání systému nebo restartuje správce front.

REST API V3 V 9.3.0 **ibm-mq-md-priorita**

Vrátí nastavení priority zprávy. Příklad:

```
ibm-mq-md-priority: 3
```

ibm-mq-md-replyTo

Určuje cíl odpovědi pro přijatou zprávu. Formát záhlaví používá standardní notaci fronty pro odpověď a správce front `replyQueue@replyQmgr`.

Příklad:

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMgr
```

V 9.3.3 **ibm-mq-vyřešeno-qmgr**

Určuje název správce front, který byl použit pro požadavek. Pokud byl v prostředku URLpoužit jedinečný název, toto záhlaví identifikuje název správce front, který je k tomuto jedinečnému názvu přidružen. Pokud jedinečný název použitý v prostředku URL odkazuje na skupinu správců front, toto záhlaví identifikuje, který správce front v rámci skupiny byl použit.

REST API V3 V 9.3.0 **ibm-mq-usr**

Vrací uživatelem definované vlastnosti zprávy. Pro zprávu lze nastavit více vlastností. V takovém případě budou existovat dvě nebo více samostatných instancí záhlaví odezvy ibm-mq-usr.

Příklad:

```
ibm-mq-usr: myIProp;5;short
ibm-mq-usr: mySProp;"hi";string
ibm-mq-usr: myBProp;true;boolean
```

Vlastnosti mají následující syntaxi:

```
ibm-mq-usr: property_name; user_value; user_type
```

property_name
Název zadané uživatelské vlastnosti. Musí se jednat o platný název vlastnosti JMS.

hodnota uživatele
Hodnota vlastnosti.

typ_uživatele
Typ vlastnosti:

- `boolean` (true/false, MQBOOL)
- `byte` (8bitové celé číslo, MQINT8)

- `short` (16bitové celé číslo, MQINT16)
- `integer` (32bitové celé číslo, MQINT32)
- `long` (64bitové celé číslo, MQINT64)
- `float` (32bitová realita, MQFLOAT32)
- `double` (64bitová realita, MQFLOAT64)
- `string` (řetězec v uvozovkách)

Formát těla odezvy

Při úspěchu tělo odezvy obsahuje tělo zprávy z přijaté zprávy. Pokud dojde k chybě, tělo odezvy obsahuje chybovou zprávu ve formátu JSON. Obě odpovědi jsou kódovány UTF-8 . Další informace viz REST API ošetření chyb.

Mějte na paměti, že při příjmu zprávy jsou podporovány pouze zprávy ve formátu IBM MQ `MQSTR` nebo `JMS TextMessage` .

Procházení fronty, která byla označena jako GET, nevrací žádný obsah.

Pokud procházená fronta obsahuje zprávy s duplicitními identifikátory zpráv, vrátí se při filtrování podle identifikátoru zprávy první zpráva.

Příklady

Následující příklady používají adresu URL prostředku v2. Používáte-li starší verzi produktu IBM MQ než IBM MQ 9.1.5, musíte místo toho použít adresu URL prostředku v1. To znamená, že v adrese URL prostředku nahradíte `v1` tam, kde příklad adresy URL používá `v2`.

Následující příklad protokoluje uživatele s názvem `mquser` s heslem `mquser`. V cURLmůže požadavek na přihlášení vypadat jako v následujícím příkladu Windows . Token LTPA je uložen v souboru `cookiejar.txt` pomocí příznaku `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v2/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"mquser\",\"password\":\"mquser\"}"
-c c:\cookiejar.txt
```

Po přihlášení uživatele se token LTPA a záhlaví `ibm-mq-rest-csrf-token` HTTP použijí k ověření dalších požadavků. `ibm-mq-rest-csrf-token` `token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- Následující příklad Windows cURL prochází další dostupnou zprávu z fronty `Q1` ve správci front `QM1`s použitím výchozích voleb:

  ```
  curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
  -X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
  -H "Accept: text/plain"
  ```

- REST API V1 REST API V2 Následující příklad Windows cURL prochází zprávu se specifickým ID korelace `000000000000000000000000000000000000000000000abcdabcdz` fronty `Q1` ve správci front `QM1`:

  ```
  curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message?
  correlationId=000000000000000000000000000000000000000000000abcdabcd"
  -X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
  -H "Accept: text/plain"
  ```

- REST API V3 V 9.3.0

Následující příklad Windows cURL je stejný jako předchozí příklad, ale používá rozhraní REST API V3. V příkladu jsou uvedeny pouze zprávy s odpovídajícím ID korelace 000000000000000000000000000000000000000000000000abcdabcdz fronty Q1 ve správci front QM1:

```
curl -k "https://localhost:9443/ibmmq/rest/v3/messaging/qmgr/QM1/queue/Q1/message?
correlationId=ID:000000000000000000000000000000000000000000000000abcdabcd"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: text/plain"
```

ODSTRANIT /messaging/qmgr/{qmgrName}/queue/{queueName}/message

Pomocí metody HTTP DELETE s prostředkem /messaging/qmgr/{qmgrName}/queue/{queueName}/message můžete destruktivně získat zprávy z přidruženého správce front a fronty.

Destruktivně získá další dostupnou zprávu z určeného správce front a fronty a vrátí tělo zprávy v těle odezvy HTTP . Správce front musí být na stejném počítači jako server mqweb. Zpráva musí mít formát MQSTR nebo JMS TextMessagea je přijata s použitím aktuálního uživatelského kontextu.

Nekompatibilní zprávy jsou ponechány ve frontě a volajícímu je vrácen odpovídající stavový kód. Například zpráva, která nemá formát MQSTR nebo JMS TextMessage .

V 9.3.0 Rozhraní REST API V3 přidává schopnost určit vlastnosti zpráv definované uživatelem a zahrnout prioritu zpráv se zprávami. Záhlaví odezvy ibm-mq-md-priority a ibm-mq-usr jsou k dispozici pouze s rozhraním REST API V3. Záhlaví požadavku ibm-mq-md-correlationId má jiný formát v rozhraní REST API V3. Záhlaví může být ID specifické pro aplikaci, nebo v případě kódovaného řetězce si zachová předponu ID: . Záhlaví odezvy ibm-mq-md-messageId a parametr dotazu mají jiný formát v rozhraní REST API V3, uchová předponu ID: .

- "URL prostředku" na stránce 2172
- "Volitelné parametry dotazu" na stránce 2173
- "Záhlaví požadavku" na stránce 2174
- "Formát těla požadavku" na stránce 2174
- "Požadavky na zabezpečení" na stránce 2174
- "Stavové kódy odezvy" na stránce 2175
- "Záhlaví odezvy" na stránce 2176
- "Formát těla odezvy" na stránce 2178
- "Příklady" na stránce 2178

URL prostředku

https://*host*:*port*/ibmmq/rest/v1/messaging/qmgr/*{qmgrName}*/queue/*{queueName}*/message

https://*host*:*port*/ibmmq/rest/v2/messaging/qmgr/*{qmgrName}*/queue/*{queueName}*/message

V 9.3.0 https://*host*:*port*/ibmmq/rest/v3/messaging/qmgr/*{qmgrName}*/queue/*{queueName}*/message

qmgrName

Určuje název správce front, ke kterému se má připojit pro systém zpráv. Správce front musí být na stejném počítači jako server mqweb.

V 9.3.3 V produktu IBM MQ 9.3.3se můžete připojit buď k lokálnímu správci front, nebo ke vzdálenému správci front. Název, který zadáte pro parametr **qmgrName** , závisí na konfiguraci serveru mqweb:

- Je-li váš server mqweb konfigurován tak, aby se připojoval pouze k lokálním správcům front, použijte název správce front.

- Je-li váš server mqweb konfigurován pro připojení vzdálených správců front, použijte jedinečný název správce front určený v informacích o připojení vzdáleného správce front.

Pomocí příkazu **dspmqweb properties** a zobrazením vlastnosti **mqRestMessagingConnectionMode** můžete určit, zda je váš server mqweb konfigurován pro připojení k lokálním správcům front nebo vzdáleným správcům front.

V názvu správce front se rozlišují velká a malá písmena.

Pokud název správce front obsahuje dopředné lomítko, tečku nebo znak procent, musí být tyto znaky zakódovány pomocí adresy URL:

- Dopředné lomítko (/) musí být zakódováno jako %2F.
- Znaménko procent (%) musí být zkódováno jako %25.

queueName

Uvádí název fronty, ze které se má získat další zpráva.

Fronta musí být definována jako lokální nebo alias odkazující na lokální frontu.

V názvu fronty se rozlišují malá a velká písmena.

Pokud název fronty obsahuje dopředné lomítko nebo znak procenta, musí být tyto znaky URL zakódovány:

- Dopředné lomítko,/, musí být zakódováno jako %2F.
- Znak procenta% musí být zakódován jako %25.

Můžete použít HTTP místo HTTPS , pokud povolíte připojení HTTP . Další informace o povolení HTTPnaleznete v tématu Konfigurace portů HTTP a HTTPS.

Volitelné parametry dotazu

REST API V1 REST API V2 **correlationId=*hexValue***

Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID korelace.

hexValue

Parametr dotazu musí být uveden jako hexadecimální řetězec o délce 48 znaků, který představuje 24 bajtů.

Příklad:

```
../message?correlationId=414d5120514d414445562020202020202067d8bf5923582e02
```

REST API V3 V 9.3.0 **correlationId= ID:*hexValue* nebo correlationId=*application_specific_value***

Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID korelace.

hexValue

Parametr dotazu musí být uveden jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů a předchází mu řetězec `"ID:"`.

Příklad:

```
../message?correlationId=ID:414d5120514d414445562020202020202067d8bf5923582e02
```

specifická_hodnota aplikace

Parametr dotazu lze zadat jako řetězec specifický pro aplikaci.

Příklad:

```
../message?correlationId=My-Custom-CorrelId
```

REST API V1 REST API V2 **messageId=*hexValue***

Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID zprávy.

hexValue
Parametr dotazu musí být uveden jako hexadecimální řetězec o délce 48 znaků, který představuje 24 bajtů.

Příklad:

```
../message?messageId=414d5120514d414445562020202020067d8ce5923582f07
```

REST API V3 V 9.3.0 **messageId= ID:*hexValue***
Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID zprávy.

hexValue
Parametr dotazu musí být uveden jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů a předchází mu řetězec `"ID:"`.

Příklad:

```
../message?messageId=ID:414d5120514d414445562020202020067d8ce5923582f07
```

wait=*integerValue*
Určuje, že metoda HTTP bude čekat *integerValue* milisekund na zpřístupnění další zprávy.

integerValue
Parametr dotazu musí být zadán jako celé číslo představující dobu trvání v milisekundách. Maximální hodnota je 2147483647.

Příklad:

```
../message?wait=120000
```

Záhlaví požadavku

S požadavkem musí být odeslána následující záhlaví:

Autorizace
Toto záhlaví musí být odesláno, pokud používáte základní ověření. Další informace viz Použití základního ověření HTTP pomocí rozhraní REST API.

ibm-mq-rest-csrf-token
Toto záhlaví musí být nastaveno, ale hodnota může být libovolná, včetně prázdné hodnoty.

Následující záhlaví lze volitelně odeslat s požadavkem:

Přijmout-znaková sada
Toto záhlaví lze použít k označení, která znaková sada je pro odezvu přijatelná. Je-li uvedeno, toto záhlaví musí být nastaveno jako UTF-8.

Accept-Language
Toto záhlaví určuje požadovaný jazyk pro všechny výjimky nebo chybové zprávy vrácené v těle zprávy odpovědi.

Formát těla požadavku

Není.

Požadavky na zabezpečení

Volající musí být ověřen na serveru mqweb. Role `MQWebAdmin` a `MQWebAdminRO` nelze použít pro messaging REST API. Další informace o zabezpečení REST APInaleznete v tématu Zabezpečení produktů IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak administrative REST API.

Činiteli zabezpečení volajícího musí být udělena schopnost získat zprávy z určené fronty:

- Fronta, která je zadána v části *{queueName}* adresy URL prostředku, musí být povolena v produktu GET.
- MQ Appliance ALW Pro frontu určenou v části *{queueName}* adresy URL prostředku musí být činiteli zabezpečení volajícího uděleno oprávnění +GET, +INQ a +BROWSE.
- z/OS Pro frontu, která je zadána částí *{queueName}* adresy URL prostředku, UPDATE, musí být udělen přístup k činiteli zabezpečení volajícího.

Tímto činitelem zabezpečení může být uživatel, který spustil server mqweb, nebo uživatel, který je přihlášen k serveru mqweb. Pokud je správcem front, ke kterému se připojujete, vzdálený správce front, může být namísto toho činitelem zabezpečení uživatel zadaný pomocí pověření v informacích o připojení vzdáleného správce front nebo jiný uživatel určený pomocí pravidel zabezpečení kanálu. Další informace naleznete v tématu Určení činitele zabezpečení používaného produktem messaging REST API.

ALW Na systému AIX, Linux, and Windows můžete udělit oprávnění k činitelům zabezpečení, aby mohli používat prostředky IBM MQ, pomocí příkazu **setmqaut**. Další informace viz téma **setmqaut** (udělit nebo odvolat oprávnění).

z/OS V systému z/OS viz Nastavení zabezpečení v systému z/OS.

Stavové kódy odezvy

200
Zpráva byla úspěšně přijata.

204
Nejsou k dispozici žádné zprávy.

400
Byla zadána neplatná data.

Byla například zadána neplatná hodnota parametru dotazu.

401
Neověřeno.

Volající musí být ověřen na serveru mqweb a musí být členem nejméně jedné z rolí `MQWebAdmin`, `MQWebAdminRO` nebo `MQWebUser`. Musí být také uvedeno záhlaví `ibm-mq-rest-csrf-token` . Další informace viz “Požadavky na zabezpečení” na stránce 2174.

403
Neautorizováno.

Volající je ověřen na serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem nebo k podmnožině požadovaných prostředků IBM MQ nebo není v roli `MQWebUser` . Další informace o požadovaném přístupu viz “Požadavky na zabezpečení” na stránce 2174.

404
Fronta neexistuje.

405
Fronta je zablokována GET.

500
Problém se serverem nebo kód chyby z IBM MQ.

501
Odpověď HTTP nelze sestavit.

Například přijatá zpráva má nesprávný typ nebo má správný typ, ale tělo nelze zpracovat.

502
Aktuální činitel zabezpečení nemůže přijmout zprávu, protože poskytovatel systému zpráv nepodporuje požadovanou funkci. Například, pokud je cesta ke třídě serveru mqweb neplatná.

503
Správce front není spuštěn.

Záhlaví odezvy

S odezvou jsou vrácena následující záhlaví:

Obsah-Jazyk
Určuje identifikátor jazyka zprávy odpovědi v případě chyb nebo výjimek. Používá se ve spojení se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakékoli chybové stavy nebo podmínky výjimky. Výchozí hodnota serveru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP , a to i v případě, že není k dispozici žádný obsah. Hodnota obsahuje délku (bajty) dat zprávy.

Content-Type
Určuje typ obsahu vráceného v těle odezvy přijaté zprávy. Při úspěchu je hodnota `text/plain;charset=utf-8`. V případě chyb nebo výjimek je hodnota `application/json;charset=utf-8`.

REST API V1 REST API V2 **ibm-mq-md-correlationId**
Určuje ID korelace přijaté zprávy. Záhlaví je vráceno, pokud přijatá zpráva obsahuje platné ID korelace. Je reprezentován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů.

Příklad:

```
ibm-mq-md-correlationId: 414d5120514d4144455620202020202067d8bf5923582e02
```

REST API V3 V 9.3.0 **ibm-mq-md-correlationId**
Určuje ID korelace přijaté zprávy. Záhlaví je vráceno, pokud přijatá zpráva obsahuje platné ID korelace. ID korelace může mít jednu z následujících forem:

- 48znakový hexadecimální kódovaný řetězec, představující 24 bajtů, s předponou s řetězcem `"ID:"`. Příklad:

  ```
  ibm-mq-md-correlationId: ID:414d5120514d4144455620202020202067d8bf5923582e02
  ```

- Hodnota specifická pro aplikaci. Hodnota je řetězec specifický pro aplikaci:

  ```
  ibm-mq-md-correlationId: My-Custom-CorrelId
  ```

ibm-mq-md-vypršení platnosti
Určuje zbývající dobu trvání vypršení platnosti přijaté zprávy. Záhlaví může mít jednu z následujících hodnot:

bez omezení
Platnost zprávy nevyprší.

Celočíselná hodnota
Zbývající milisekundy před vypršením platnosti zprávy.

REST API V1 REST API V2 **ibm-mq-md-messageId**
Uvádí ID zprávy, které je přiděleno IBM MQ této zprávě. Stejně jako záhlaví `ibm-mq-md-correlationId` je reprezentováno jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů.

Příklad:

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

REST API V3 V 9.3.0 **ibm-mq-md-messageId**

Uvádí ID zprávy, které je přiděleno IBM MQ této zprávě. Stejně jako záhlaví `ibm-mq-md-correlationId` je reprezentován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů, s předponou s řetězcem `"ID:"`

Příklad:

```
ibm-mq-md-messageId: ID:414d5120514d414445562020202020202067d8ce5923582f07
```

ibm-mq-md-persistence-trvalost

Určuje trvání přijaté zprávy. Záhlaví může mít jednu z následujících hodnot:

nonPersistent
Zpráva nepřežije selhání systému nebo restartování správce front.

Trvalý
Zpráva přežije selhání systému nebo restartuje správce front.

REST API V3 V 9.3.0 **ibm-mq-md-priorita**

Vrátí nastavení priority zprávy. Příklad:

```
ibm-mq-md-priority: 3
```

ibm-mq-md-replyTo

Určuje cíl odpovědi pro přijatou zprávu. Formát záhlaví používá standardní notaci fronty pro odpověď a správce front `replyQueue@replyQmgr`.

Příklad:

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMgr
```

V 9.3.3 **ibm-mq-vyřešeno-qmgr**

Určuje název správce front, který byl použit pro požadavek. Pokud byl v prostředku URLpoužit jedinečný název, toto záhlaví identifikuje název správce front, který je k tomuto jedinečnému názvu přidružen. Pokud jedinečný název použitý v prostředku URL odkazuje na skupinu správců front, toto záhlaví identifikuje, který správce front v rámci skupiny byl použit.

REST API V3 V 9.3.0 **ibm-mq-usr**

Vrací uživatelem definované vlastnosti zprávy. Pro zprávu lze nastavit více vlastností. V takovém případě budou existovat dvě nebo více samostatných instancí záhlaví odezvy ibm-mq-usr.

Příklad:

```
ibm-mq-usr: myIProp;5;short
ibm-mq-usr: mySProp;"hi";string
ibm-mq-usr: myBProp;true;boolean
```

Vlastnosti mají následující syntaxi:

```
ibm-mq-usr: property_name; user_value; user_type
```

property_name
Název zadané uživatelské vlastnosti. Musí se jednat o platný název vlastnosti JMS.

hodnota uživatele
Hodnota vlastnosti.

typ_uživatele
Typ vlastnosti:

- `boolean` (true/false, MQBOOL)
- `byte` (8bitové celé číslo, MQINT8)

- `short` (16bitové celé číslo, MQINT16)
- `integer` (32bitové celé číslo, MQINT32)
- `long` (64bitové celé číslo, MQINT64)
- `float` (32bitová realita, MQFLOAT32)
- `double` (64bitová realita, MQFLOAT64)
- `string` (řetězec v uvozovkách)

Formát těla odezvy

Při úspěchu tělo odezvy obsahuje tělo zprávy z přijaté zprávy. Pokud dojde k chybě, tělo odezvy obsahuje chybovou zprávu ve formátu JSON. Obě odpovědi jsou kódovány UTF-8 . Další informace viz REST API ošetření chyb.

Mějte na paměti, že při příjmu zprávy jsou podporovány pouze zprávy ve formátu IBM MQ `MQSTR` a JMS `TextMessage` . Následně jsou všechny zprávy přijímány pod synchronizačním bodem a všechny neošetřené zprávy jsou ponechány ve frontě. Frontu IBM MQ lze konfigurovat tak, aby se tyto nezpracovatelné zprávy přesunovaly do alternativního místa určení. Další informace naleznete v tématu Zpracování nezpracovatelných zpráv ve třídách IBM MQ pro JMS.

Příklady

Následující příklady používají adresu URL prostředku v2. Používáte-li starší verzi produktu IBM MQ než IBM MQ 9.1.5, musíte místo toho použít adresu URL prostředku v1. To znamená, že v adrese URL prostředku nahradíte `v1` tam, kde příklad adresy URL používá `v2`.

Následující příklad protokoluje uživatele s názvem `mquser` s heslem `mquser`. V cURLmůže požadavek na přihlášení vypadat jako v následujícím příkladu Windows . Token LTPA je uložen v souboru `cookiejar.txt` pomocí příznaku `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v2/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"mquser\",\"password\":\"mquser\"}"
-c c:\cookiejar.txt
```

Po přihlášení uživatele se token LTPA a záhlaví `ibm-mq-rest-csrf-token` HTTP použijí k ověření dalších požadavků. `ibm-mq-rest-csrf-token` `token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- Následující příklad Windows cURL odebere další dostupnou zprávu z fronty `Q1` ve správci front `QM1`s použitím výchozích voleb:

  ```
  curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
  -X DELETE -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
  -H "Accept: text/plain"
  ```

- Následující příklad Windows cURL odebere zprávu se specifickým ID korelace `00000000000000000000000000000000000000000000abcdabcd`z fronty `Q1` ve správci front `QM1`:

  ```
  curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message?
  correlationId=00000000000000000000000000000000000000000000abcdabcd"
  -X DELETE -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
  -H "Accept: text/plain"
  ```

- Následující příklad Windows cURL odebere zprávu se specifickým ID korelace `00000000000000000000000000000000000000000000abcdabcd`z fronty `Q1` ve správci front `QM1`, která bude čekat až 30 sekund, než bude zpráva k dispozici. Pokud uplyne 30 sekund bez vložení uvedené zprávy do fronty, volání DELETE se vrátí bez zprávy:

  ```
  curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message?
  correlationId=00000000000000000000000000000000000000000000abcdabcd&wait=30000"
  -X DELETE -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
  -H "Accept: text/plain"
  ```

/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist

Pomocí metody HTTP GET s prostředkem `/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist` můžete získat seznam dostupných zpráv z určené fronty v zadaném správci front.

GET /messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist

Pomocí metody HTTP GET s prostředkem `/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist` můžete získat seznam dostupných zpráv z určené fronty v zadaném správci front.

Prochází souhrnný seznam zpráv z určeného správce front a fronty. Správce front musí být na stejném počítači jako server mqweb. Souhrnná data jsou vrácena v těle odezvy HTTP jako pole ve formátu JSON. Data neobsahují informační obsah zpráv a jsou přijímána pomocí aktuálního uživatelského kontextu. Z přidružené fronty nejsou odebrány žádné zprávy.

Je-li učiněn požadavek na získání seznamu dostupných zpráv z fronty, která je GET zablokována, vrátí se prázdné pole JSON.

- "URL prostředku" na stránce 2179
- "Volitelné parametry dotazu" na stránce 2180
- "Záhlaví požadavku" na stránce 2181
- "Formát těla požadavku" na stránce 2181
- "Požadavky na zabezpečení" na stránce 2181
- "Stavové kódy odezvy" na stránce 2182
- "Záhlaví odezvy" na stránce 2182
- "Formát těla odezvy" na stránce 2183
- "Příklady" na stránce 2183

URL prostředku

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

V 9.3.0 `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

qmgrName

Určuje název správce front, ke kterému se má připojit pro systém zpráv. Správce front musí být na stejném počítači jako server mqweb.

V 9.3.3 V produktu IBM MQ 9.3.3se můžete připojit buď k lokálnímu správci front, nebo ke vzdálenému správci front. Název, který zadáte pro parametr **qmgrName** , závisí na konfiguraci serveru mqweb:

- Je-li váš server mqweb konfigurován tak, aby se připojoval pouze k lokálním správcům front, použijte název správce front.
- Je-li váš server mqweb konfigurován pro připojení vzdálených správců front, použijte jedinečný název správce front určený v informacích o připojení vzdáleného správce front.

Pomocí příkazu **dspmqweb properties** a zobrazením vlastnosti **mqRestMessagingConnectionMode** můžete určit, zda je váš server mqweb konfigurován pro připojení k lokálním správcům front nebo vzdáleným správcům front.

V názvu správce front se rozlišují velká a malá písmena.

Pokud název správce front obsahuje dopředné lomítko, tečku nebo znak procent, musí být tyto znaky zakódovány pomocí adresy URL:

- Dopředné lomítko (/) musí být zakódováno jako %2F.
- Znaménko procent (%) musí být zkódováno jako %25.

queueName
Určuje název fronty, ze které se mají procházet zprávy.

Fronta musí být definována jako lokální nebo alias, který odkazuje na lokální frontu.

V názvu fronty se rozlišují malá a velká písmena.

Pokud název fronty obsahuje dopředné lomítko nebo znak procenta, musí být tyto znaky URL zakódovány:

- Dopředné lomítko,/, musí být zakódováno jako %2F.
- Znak procenta% musí být zakódován jako %25.

Můžete použít HTTP místo HTTPS , pokud povolíte připojení HTTP . Další informace o povolení HTTPnaleznete v tématu Konfigurace portů HTTP a HTTPS.

Volitelné parametry dotazu

REST API V2 **correlationId=*hexValue***
Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID korelace.

hexValue
Parametr dotazu musí být uveden jako hexadecimální řetězec o délce 48 znaků, který představuje 24 bajtů.

Příklad:

```
../messagelist?correlationId=414d5120514d4144455620202020202067d8bf5923582e02
```

REST API V3 V 9.3.0 **correlationId= ID:*hexValue* nebo correlationId=*application_specific_value***
Určuje, že metoda HTTP vrací seznam zpráv s odpovídajícím ID korelace.

hexValue
Parametr dotazu musí být uveden jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů a předchází mu řetězec `"ID:"`.

Příklad:

```
../message?correlationId=ID:414d5120514d4144455620202020202067d8bf5923582e02
```

specifická_hodnota aplikace
Parametr dotazu lze zadat jako řetězec specifický pro aplikaci.

Příklad:

```
../message?correlationId=My-Custom-CorrelId
```

REST API V1 REST API V2 **messageId=*hexValue***
Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID zprávy.

hexValue
Parametr dotazu musí být uveden jako hexadecimální řetězec o délce 48 znaků, který představuje 24 bajtů.

Příklad:

```
../message?messageId=414d5120514d4144455620202020202067d8ce5923582f07
```

REST API V3 V 9.3.0 **messageId= ID:*hexValue***

Určuje, že metoda HTTP vrátí další zprávu s odpovídajícím ID zprávy.

hexValue

Parametr dotazu musí být uveden jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů a předchází mu řetězec `"ID:"`.

Příklad:

```
../message?messageId=ID:414d5120514d414445562020202020202067d8ce5923582f07
```

limit=*integerValue*

Určuje, že tělo odezvy metody HTTP je omezeno na prvky JSON *integerValue* .

integerValue

Parametr dotazu musí být uveden jako celočíselná hodnota, která představuje maximální počet prvků obsažených v těle odezvy JSON.

Výchozí hodnota je 10 a maximální hodnota je 2147483647.

Příklad:

```
../messagelist?limit=250
```

Záhlaví požadavku

S požadavkem musí být odeslána následující záhlaví:

Autorizace

Toto záhlaví musí být odesláno, pokud používáte základní ověření. Další informace viz Použití základního ověření HTTP pomocí rozhraní REST API.

ibm-mq-rest-csrf-token

Toto záhlaví musí být nastaveno, ale hodnota může být libovolná, včetně prázdné hodnoty.

Následující záhlaví lze volitelně odeslat s požadavkem:

Přijmout-znaková sada

Toto záhlaví lze použít k označení, která znaková sada je pro odezvu přijatelná. Je-li uvedeno, toto záhlaví musí být nastaveno jako `UTF-8`.

Accept-Language

Toto záhlaví určuje požadovaný jazyk pro všechny výjimky nebo chybové zprávy vrácené v těle zprávy odpovědi.

Formát těla požadavku

Není.

Požadavky na zabezpečení

Volající musí být ověřen na serveru mqweb. Role `MQWebAdmin` a `MQWebAdminRO` nelze použít pro messaging REST API. Další informace o zabezpečení REST APInaleznete v tématu Zabezpečení produktů IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak administrative REST API.

Činitel zabezpečení volajícího musí mít udělenu schopnost procházet zprávy z určené fronty:

- Fronta, která je zadána v části *{queueName}* adresy URL prostředku, musí být povolena v produktu `BROWSE`.

- ALW MQ Appliance Pro frontu určenou v části *{queueName}* adresy URL prostředku musí být činiteli zabezpečení volajícího uděleno oprávnění +GET, +INQ a +BROWSE.
- z/OS Pro frontu, která je zadána částí *{queueName}* adresy URL prostředku, UPDATE, musí být udělen přístup k činiteli zabezpečení volajícího.

Tímto činitelem zabezpečení může být uživatel, který spustil server mqweb, nebo uživatel, který je přihlášen k serveru mqweb. Pokud je správcem front, ke kterému se připojujete, vzdálený správce front, může být namísto toho činitelem zabezpečení uživatel zadaný pomocí pověření v informacích o připojení vzdáleného správce front nebo jiný uživatel určený pomocí pravidel zabezpečení kanálu. Další informace naleznete v tématu Určení činitele zabezpečení používaného produktem messaging REST API.

ALW Na systému AIX, Linux, and Windows můžete udělit oprávnění k činitelům zabezpečení, aby mohli používat prostředky IBM MQ, pomocí příkazu **setmqaut**. Další informace viz téma **setmqaut** (udělit nebo odvolat oprávnění).

z/OS V systému z/OS viz Nastavení zabezpečení v systému z/OS.

Stavové kódy odezvy

200
Seznam zpráv byl úspěšně přijat.

400
Byla zadána neplatná data.

Byla například zadána neplatná hodnota parametru dotazu.

401
Neověřeno.

Volající musí být ověřen na serveru mqweb a musí být členem nejméně jedné z rolí `MQWebAdmin`, `MQWebAdminRO` nebo `MQWebUser`. Musí být také uvedeno záhlaví `ibm-mq-rest-csrf-token` . Další informace viz “Požadavky na zabezpečení” na stránce 2181.

403
Neautorizováno.

Volající je ověřen na serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem nebo k podmnožině požadovaných prostředků IBM MQ nebo není v roli `MQWebUser` . Další informace o požadovaném přístupu viz “Požadavky na zabezpečení” na stránce 2181.

404
Fronta neexistuje.

500
Problém se serverem nebo kód chyby z IBM MQ.

501
Odpověď HTTP nelze sestavit.

Například přijatá zpráva má nesprávný typ nebo má správný typ, ale tělo nelze zpracovat.

502
Aktuální činitel zabezpečení nemůže přijmout zprávu, protože poskytovatel systému zpráv nepodporuje požadovanou funkci. Například, pokud je cesta ke třídě serveru mqweb neplatná.

503
Správce front není spuštěn.

Záhlaví odezvy

Obsah-Jazyk
Určuje identifikátor jazyka zprávy odpovědi v případě chyb nebo výjimek. Používá se ve spojení se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakékoli chybové stavy

nebo podmínky výjimky. Výchozí hodnota serveru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP , a to i v případě, že není k dispozici žádný obsah. Hodnota obsahuje délku dat zprávy v bajtech.

Content-Type
Určuje typ těla odezvy. Hodnota je `application/json;charset=utf-8`.

V 9.3.3 **ibm-mq-vyřešeno-qmgr**
Určuje název správce front, který byl použit pro požadavek. Pokud byl v prostředku URLpoužit jedinečný název, toto záhlaví identifikuje název správce front, který je k tomuto jedinečnému názvu přidružen. Pokud jedinečný název použitý v prostředku URL odkazuje na skupinu správců front, toto záhlaví identifikuje, který správce front v rámci skupiny byl použit.

Formát těla odezvy

Při úspěchu je tělo odezvy kódovanou odezvou UTF-8 . Odezva obsahuje vnější objekt JSON, který obsahuje jediné pole JSON s názvem `messages`. Každý prvek v poli je objekt JSON, který obsahuje informace o zprávě ve frontě. Každý prvek obsahuje následující atributy:

REST API V1 REST API V2 **correlationId**
Určuje ID korelace zprávy. Hodnota je vrácena v případě, že zpráva obsahuje platné ID korelace.

REST API V3 V 9.3.0 **correlationId**
Určuje ID korelace zprávy. Hodnota je vrácena v případě, že zpráva obsahuje platné ID korelace. ID korelace má předponu `"ID:"` nebo může mít hodnotu specifickou pro aplikaci.

REST API V1 REST API V2 **messageId**
Určuje ID zprávy přidělené produktem IBM MQ této zprávě. Je reprezentován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů.

REST API V3 V 9.3.0 **messageId**
Určuje ID zprávy přidělené produktem IBM MQ této zprávě. Je reprezentován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů. Před ID zprávy je uveden řetězec `"ID:"`.

formát
Určuje pole formátu MQMD. Za normálních okolností budou textové zprávy obsahovat hodnotu IBM MQ `MQSTR` .

Je-li učiněn požadavek na získání seznamu zpráv ve frontě, která má zablokován příkaz GET, vrátí se prázdné pole JSON.

Pokud dojde k chybě, tělo odezvy obsahuje chybovou zprávu ve formátu JSON. Další informace viz REST API ošetření chyb.

Příklady

Následující příklady používají adresu URL prostředku v2. Používáte-li starší verzi produktu IBM MQ než IBM MQ 9.1.5, musíte místo toho použít adresu URL prostředku v1. To znamená, že v adrese URL prostředku nahradíte v1 tam, kde příklad adresy URL používá v2.

Následující příklad protokoluje uživatele s názvem `mquser` s heslem `mquser`. V cURLmůže požadavek na přihlášení vypadat jako v následujícím příkladu Windows . Token LTPA je uložen v souboru `cookiejar.txt` pomocí příznaku `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v2/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"mquser\",\"password\":\"mquser\"}"
-c c:\cookiejar.txt
```

Po přihlášení uživatele se token LTPA a záhlaví `ibm-mq-rest-csrf-token` HTTP použijí k ověření dalších požadavků. `ibm-mq-rest-csrf-token token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- V následujícím příkladu Windows cURL je uvedeno dalších deset dostupných zpráv z fronty `Q1` ve správci front `QM1`s použitím výchozích voleb:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/messagelist"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: application/json"
```

- V následujícím příkladu Windows cURL jsou uvedeny další dvě stě dostupných zpráv z fronty `Q1` ve správci front `QM1`s použitím výchozích voleb:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/messagelist?
limit=200"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: application/json"
```

- REST API V1 REST API V2 V následujícím příkladu Windows cURL jsou uvedeny pouze zprávy s odpovídajícím ID korelace `000000000000000000000000000000000000000000abcdabcdz` fronty `Q1` ve správci front `QM1`:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/messagelist?
correlationId=000000000000000000000000000000000000000000abcdabcd"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: application/json"
```

- REST API V3 V 9.3.0

  Následující příklad Windows cURL je stejný jako předchozí příklad, ale používá rozhraní REST API V3. V příkladu jsou uvedeny pouze zprávy s odpovídajícím ID korelace `000000000000000000000000000000000000000000abcdabcdz` fronty `Q1` ve správci front `QM1`:

```
curl -k "https://localhost:9443/ibmmq/rest/v3/messaging/qmgr/QM1/queue/Q1/messagelist?
correlationId=ID:000000000000000000000000000000000000000000abcdabcd"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: application/json"
```

/messaging/qmgr/{qmgrName}/topic/{topicString}/message

Pomocí metody HTTP POST s prostředkem `/messaging/qmgr/{qmgrName}/topic/{topicString}/message` můžete publikovat zprávy do určeného tématu v určeném správci front.

POST /messaging/qmgr/{qmgrName}/topic/{topicString}/message

Pomocí metody HTTP POST s prostředkem `/messaging/qmgr/{qmgrName}/topic/{topicString}/message` můžete publikovat zprávy do určeného tématu v určeném správci front.

Publikuje textovou zprávu v těle požadavku HTTP do určeného správce front a tématu. Správce front musí být na stejném počítači jako server mqweb a podporovány jsou pouze textové zprávy. Zprávy jsou publikovány jako zprávy ve formátu MQSTR nebo `JMS TextMessage` s použitím aktuálního kontextu uživatele a mají výchozí prioritu zpráv 4 pro rozhraní API REST V2a `asDestination` pro rozhraní API REST V3.

V 9.3.0 Rozhraní REST API V3 přidává schopnost určit vlastnosti zprávy definované uživatelem a zahrnout prioritu zprávy. Záhlaví požadavků ibm-mq-md-priority a ibm-mq-usr jsou k dispozici pouze s rozhraním REST API V3. Záhlaví požadavku ibm-mq-md-correlationId má jiný formát v rozhraní REST API V3. Záhlaví může být ID specifické pro aplikaci, nebo v případě kódovaného řetězce vyžaduje předponu `ID:` . Pokud váš požadavek POST obsahuje zprávy definované uživatelem nebo ID korelace specifické pro aplikaci, bude zpráva formátována jako `JMS TextMessage`.

- "URL prostředku" na stránce 2185
- "Záhlaví požadavku" na stránce 2186

URL prostředku

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/topic/{topicString}/message`

V 9.3.0 `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/topic/{topicString}/message`

qmgrName

Určuje název správce front, ke kterému se má připojit pro systém zpráv.

V 9.3.3 V produktu IBM MQ 9.3.3se můžete připojit buď k lokálnímu správci front, nebo ke vzdálenému správci front. Název, který zadáte pro parametr **qmgrName** , závisí na konfiguraci serveru mqweb:

- Je-li váš server mqweb konfigurován tak, aby se připojoval pouze k lokálním správcům front, použijte název správce front.
- Je-li váš server mqweb konfigurován pro připojení vzdálených správců front, použijte jedinečný název správce front určený v informacích o připojení vzdáleného správce front.

Pomocí příkazu **dspmqweb properties** a zobrazením vlastnosti **mqRestMessagingConnectionMode** můžete určit, zda je váš server mqweb konfigurován pro připojení k lokálním správcům front nebo vzdáleným správcům front.

V názvu se rozlišují velká a malá písmena.

Pokud název obsahuje dopředné lomítko, tečku nebo znak procenta, musí být tyto znaky URL zakódovány:

- Dopředné lomítko musí být zakódováno jako %2F.
- Tečka musí být zakódována jako %2E.
- Znak procenta musí být zakódován jako %25.

topicString

Určuje řetězec tématu, v němž má být zpráva publikována.

V řetězci tématu se rozlišují malá a velká písmena. Řetězec tématu může obsahovat více úrovní témat oddělených oddělovačem dopředného lomítka.

Pokud řetězec tématu obsahuje znak procenta, tečku nebo otazník, musí být tyto znaky URL zakódovány:

- Znak procenta musí být zakódován jako %25.
- Tečka musí být zakódována jako %2E.
- Otazník musí být zakódován jako %3F.

Pokud řetězec tématu začíná nebo končí dopředným lomítkem, musí být zakódován znakem %2F.

Chcete-li například publikovat do řetězce tématu, postupujte takto:

- `sport/football` ve správci front `MY.QMGR`použijte následující URL:

```
https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/MY%2EQMGR/topic/sport/football/message
```

- `/sport/football` ve správci front `MY.QMGR`použijte následující URL:

```
https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/MY%2EQMGR/topic/%2Fsport/football/
message
```

Můžete použít HTTP místo HTTPS , pokud povolíte připojení HTTP . Další informace o povolení HTTPnaleznete v tématu Konfigurace portů HTTP a HTTPS.

Záhlaví požadavku

S požadavkem musí být odeslána následující záhlaví:

Autorizace
Toto záhlaví musí být odesláno, pokud používáte základní ověření. Další informace viz Použití základního ověření HTTP pomocí rozhraní REST API.

Content-Type
Toto záhlaví musí být odesláno s jednou z následujících hodnot:

- `text/plain;charset=utf-8`
- `text/html;charset=utf-8`
- `text/xml;charset=utf-8`
- `application/json;charset=utf-8`
- `application/xml;charset=utf-8`

ibm-mq-rest-csrf-token
Toto záhlaví musí být nastaveno, ale hodnota může být libovolná, včetně prázdné hodnoty.

Následující záhlaví lze volitelně odeslat s požadavkem:

Accept-Language
Toto záhlaví určuje požadovaný jazyk pro všechny výjimky nebo chybové zprávy vrácené v těle zprávy odpovědi.

ibm-mq-md-vypršení platnosti
Toto záhlaví nastavuje dobu trvání vypršení platnosti pro vytvořenou zprávu. Vypršení platnosti zprávy začíná od okamžiku, kdy zpráva dorazí do správce front. V důsledku toho je latence sítě ignorována. Záhlaví musí být uvedeno jako jedna z následujících hodnot:

bez omezení
Platnost zprávy nevyprší.

Tato hodnota je výchozí hodnota.

Celočíselná hodnota
Milisekundy před vypršením platnosti zprávy.

Omezeno na rozsah 0-99999999900.

ibm-mq-md-persistence-trvalost
Toto záhlaví nastavuje perzistenci pro vytvořenou zprávu. Záhlaví musí být uvedeno jako jedna z následujících hodnot:

nonPersistent
Zpráva nepřežije selhání systému nebo restartování správce front.

Tato hodnota je výchozí hodnota.

Trvalý
Zpráva přežije selhání systému nebo restartuje správce front.

REST API | V3 | V 9.3.0 **ibm-mq-md-priorita**
Toto záhlaví nastavuje prioritu vytvořené zprávy. Záhlaví musí být uvedeno jako jedna z následujících hodnot:

asDestination
Zpráva používá prioritu uvedenou v atributu DEFPRTY základního objektu fronty IBM MQ .

Celočíselná hodnota
Určete skutečnou prioritu jako celé číslo v rozsahu 0-9.

Příklad:

```
ibm-mq-md-priority: asDestination
```

ibm-mq-md-replyTo
Toto záhlaví nastaví cíl odpovědi pro vytvořenou zprávu. Formát záhlaví používá standardní notaci zadání fronty pro odpověď a volitelného správce front: `replyQueue[@replyQmgr]`

Příklad:

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMgr
```

REST API V3 V 9.3.0 **ibm-mq-usr**
Nastavte uživatelem definované vlastnosti zprávy požadavku. Pro zprávu lze nastavit více vlastností. Můžete zadat více vlastností oddělených čárkami v jednom záhlaví požadavku ibm-mq-usr, nebo můžete použít dvě nebo více samostatných instancí záhlaví požadavku ibm-mq-usr.

Příklad:

```
ibm-mq-usr: myIProp;5;short
ibm-mq-usr: mySProp;"hi";string
ibm-mq-usr: myBProp;true;boolean
ibm-mq-usr: myA;5;byte,myB;-10;integer
```

Vlastnosti mají následující syntaxi:

```
ibm-mq-usr: property_name; user_value; user_type
```

property_name
Název zadané uživatelské vlastnosti. Musí se jednat o platný název vlastnosti JMS.

hodnota uživatele
Hodnota vlastnosti.

typ_uživatele
Typ vlastnosti:

- `boolean` (true/false, MQBOOL)
- `byte` (8bitové celé číslo, MQINT8)
- `short` (16bitové celé číslo, MQINT16)
- `integer` (32bitové celé číslo, MQINT32)
- `long` (64bitové celé číslo, MQINT64)
- `float` (32bitová realita, MQFLOAT32)
- `double` (64bitová realita, MQFLOAT64)
- `string` (řetězec v uvozovkách)

Formát těla požadavku

Tělo požadavku musí být textové a musí používat kódování UTF-8 . Není vyžadována žádná specifická struktura textu. Vytvoří se zpráva ve formátu `MQSTR` obsahující text těla požadavku a publikuje se do uvedeného tématu.

REST API V3 V 9.3.0 Pokud jsou použity uživatelem definované vlastnosti rozhraní REST API V3 nebo funkce ID korelace specifické pro aplikaci, pak se vytvoří zpráva ve formátu `JMS TextMessage` obsahující text těla požadavku a vloží se do uvedené fronty.

Další informace viz příklady.

Požadavky na zabezpečení

Volající musí být ověřen na serveru mqweb. Role `MQWebAdmin` a `MQWebAdminRO` nelze použít pro messaging REST API. Další informace o zabezpečení REST APInaleznete v tématu Zabezpečení produktů IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak administrative REST API.

Činiteli zabezpečení volajícího musí být udělena schopnost publikovat zprávy do uvedeného tématu:

- Téma určené částí *{topicString}* prostředku URL musí být `PUBLISH` povoleno.
- MQ Appliance ALW Pro téma, které je určeno částí *{topicString}* prostředku URL, musí být činiteli zabezpečení volajícího uděleno oprávnění +PUB .
- z/OS Pro téma, které je určeno částí *{topicString}* prostředku URL, musí být přístup `UPDATE` udělen činiteli zabezpečení volajícího.

Tímto činitelem zabezpečení může být uživatel, který spustil server mqweb, nebo uživatel, který je přihlášen k serveru mqweb. Pokud je správcem front, ke kterému se připojujete, vzdálený správce front, může být namísto toho činitelem zabezpečení uživatel zadaný pomocí pověření v informacích o připojení vzdáleného správce front nebo jiný uživatel určený pomocí pravidel zabezpečení kanálu. Další informace naleznete v tématu Určení činitele zabezpečení používaného produktem messaging REST API.

ALW Na systému AIX, Linux, and Windows můžete udělit oprávnění k činitelům zabezpečení, aby mohli používat prostředky IBM MQ, pomocí příkazu **setmqaut**. Další informace viz téma **setmqaut** (udělit nebo odvolat oprávnění).

z/OS V systému z/OS viz Nastavení zabezpečení v systému z/OS.

Používáte-li rozšířené zabezpečení zpráv (AMS) s produktem messaging REST API, mějte na paměti, že všechny zprávy jsou šifrovány pomocí kontextu serveru mqweb, nikoli kontextu uživatele, který zprávu zveřejňuje.

Stavové kódy odezvy

201
Zpráva byla úspěšně vytvořena a publikována.

400
Byla zadána neplatná data.

Byla například zadána neplatná hodnota záhlaví požadavku.

401
Neověřeno.

Volající musí být ověřen na serveru mqweb a musí být členem nejméně jedné z rolí `MQWebAdmin`, `MQWebAdminRO` nebo `MQWebUser`. Musí být také uvedeno záhlaví `ibm-mq-rest-csrf-token` . Další informace viz “Požadavky na zabezpečení” na stránce 2188.

403
Neautorizováno.

Volající je ověřen na serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem nebo k podmnožině požadovaných prostředků IBM MQ nebo není v roli `MQWebUser` . Další informace o požadovaném přístupu viz “Požadavky na zabezpečení” na stránce 2188.

404
Správce front neexistuje.

405
Téma je blokováno PUBLISH.

415
Záhlaví nebo tělo zprávy je nepodporovaný typ média.

Například záhlaví `Content-Type` je nastaveno na nepodporovaný typ média.

500
Problém se serverem nebo kód chyby z IBM MQ.

502
Aktuální činitel zabezpečení nemůže publikovat zprávu, protože poskytovatel systému zpráv nepodporuje požadovanou funkci. Například, pokud je cesta ke třídě serveru mqweb neplatná.

503
Správce front není spuštěn.

Záhlaví odezvy

S odezvou jsou vrácena následující záhlaví:

Obsah-Jazyk
Určuje identifikátor jazyka zprávy odpovědi v případě chyb nebo výjimek. Používá se ve spojení se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakékoli chybové stavy nebo podmínky výjimky. Výchozí hodnota serveru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP , a to i v případě, že není k dispozici žádný obsah. Při úspěchu je hodnota nula.

Content-Type
Určuje typ těla odezvy. Při úspěchu je hodnota `text/plain;charset=utf-8`. V případě chyb nebo výjimek je hodnota `application/json;charset=utf-8`.

V 9.3.3 **ibm-mq-vyřešeno-qmgr**
Určuje název správce front, který byl použit pro požadavek. Pokud byl v prostředku URLpoužit jedinečný název, toto záhlaví identifikuje název správce front, který je k tomuto jedinečnému názvu přidružen. Pokud jedinečný název použitý v prostředku URL odkazuje na skupinu správců front, toto záhlaví identifikuje, který správce front v rámci skupiny byl použit.

Formát těla odezvy

Tělo odezvy je prázdné, pokud je zpráva úspěšně publikována. Pokud dojde k chybě, tělo odezvy obsahuje chybovou zprávu. Další informace viz REST API ošetření chyb.

Příklady

Následující příklad protokoluje uživatele s názvem `mquser` s heslem `mquser`. V cURLmůže požadavek na přihlášení vypadat jako v následujícím příkladu Windows . Token LTPA je uložen v souboru `cookiejar.txt` pomocí příznaku `-c` :

```
curl -k "https://localhost:9443/ibmmq/rest/v1/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"mquser\",\"password\":\"mquser\"}"
-c c:\cookiejar.txt
```

Po přihlášení uživatele se token LTPA a záhlaví `ibm-mq-rest-csrf-token` HTTP použijí k ověření dalších požadavků. `ibm-mq-rest-csrf-token` `token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- Následující příklad Windows cURL publikuje zprávu do řetězce tématu `myTopic` ve správci front `QM1`s použitím výchozích voleb. Zpráva obsahuje text *"Ahoj světe!"*:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/topic/myTopic/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain;charset=utf-8" --data "Hello World!"
```

- Následující příklad Windows cURL publikuje trvalou zprávu do řetězce tématu `myTopic/thisTopic` ve správci front `QM1`s vypršením 2 minut. Zpráva obsahuje text *"Ahoj světe!"*:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/topic/myTopic%2FthisTopic/
message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain;charset=utf-8" -H "ibm-mq-md-persistence: persistent"
-H "ibm-mq-md-expiry: 120000" --data "Hello World!"
```

Poznámky

Tyto informace byly vyvinuty pro produkty a služby poskytované v USA.

Společnost IBM nemusí nabízet produkty, služby nebo funkce uvedené v tomto dokumentu v jiných zemích. Informace o produktech a službách, které jsou ve vaší oblasti aktuálně dostupné, získáte od místního zástupce společnosti IBM. Odkazy na produkty, programy nebo služby společnosti IBM v této publikaci nejsou míněny jako vyjádření nutnosti použití pouze uvedených produktů, programů či služeb společnosti IBM. Místo toho lze použít jakýkoli funkčně ekvivalentní produkt, program nebo službu, které neporušují žádná práva k duševnímu vlastnictví IBM. Ověření funkčnosti produktu, programu nebo služby pocházející od jiného výrobce je však povinností uživatele.

Společnost IBM může vlastnit patenty nebo nevyřízené žádosti o patenty zahrnující předměty popsané v tomto dokumentu. Vlastnictví tohoto dokumentu neposkytuje licenci k těmto patentům. Dotazy týkající se licencí můžete posílat písemně na adresu:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Odpovědi na dotazy týkající se licencí pro dvoubajtové znakové sady (DBCS) získáte od oddělení IBM Intellectual Property Department ve vaší zemi, nebo tyto dotazy můžete zasílat písemně na adresu:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

Následující odstavec se netýká Spojeného království ani jiných zemí, ve kterých je takovéto vyjádření v rozporu s místními zákony: SPOLEČNOST INTERNATIONAL BUSINESS MACHINES CORPORATION TUTO PUBLIKACI POSKYTUJE "TAK, JAK JE" BEZ JAKÝCHKOLIV ZÁRUK, VYJÁDŘENÝCH VÝSLOVNĚ NEBO VYPLÝVAJÍCÍCH Z OKOLNOSTÍ, VČETNĚ, A TO ZEJMÉNA, ZÁRUK NEPORUŠENÍ PRÁV TŘETÍCH STRAN, PRODEJNOSTI NEBO VHODNOSTI PRO URČITÝ ÚČEL. Některé právní řády u určitých transakcí nepřipouštějí vyloučení záruk výslovně vyjádřených nebo vyplývajících z okolností, a proto se na vás toto omezení nemusí vztahovat.

Uvedené údaje mohou obsahovat technické nepřesnosti nebo typografické chyby. Údaje zde uvedené jsou pravidelně upravovány a tyto změny budou zahrnuty v nových vydáních této publikace. Společnost IBM může kdykoli bez upozornění provádět vylepšení nebo změny v produktech či programech popsaných v této publikaci.

Veškeré uvedené odkazy na webové stránky, které nespravuje společnost IBM, jsou uváděny pouze pro referenci a v žádném případě neslouží jako záruka funkčnosti těchto webů. Materiály uvedené na tomto webu nejsou součástí materiálů pro tento produkt IBM a použití uvedených stránek je pouze na vlastní nebezpečí.

Společnost IBM může použít nebo distribuovat jakékoli informace, které jí sdělíte, libovolným způsobem, který společnost považuje za odpovídající, bez vyžádání vašeho svolení.

Vlastníci licence k tomuto programu, kteří chtějí získat informace o možnostech (i) výměny informací s nezávisle vytvořenými programy a jinými programy (včetně tohoto) a (ii) oboustranného využití vyměňovaných informací, mohou kontaktovat informační středisko na adrese:

IBM Corporation
Koordinátor interoperability softwaru, oddělení 49XA
3605 Dálnice 52 N

Rochester, MN 55901
U.S.A.

Poskytnutí takových informací může být podmíněno dodržením určitých podmínek a požadavků zahrnujících v některých případech uhrazení stanoveného poplatku.

Licencovaný program popsaný v těchto informacích a veškerý licencovaný materiál, který je pro něj k dispozici, jsou poskytovány společností IBM na základě podmínek IBM Smlouvy se zákazníkem, IBM Mezinárodní licenční smlouvy pro programy nebo jiné ekvivalentní smlouvy mezi námi.

Jakékoli údaje o výkonnosti obsažené v této publikaci byly zjištěny v řízeném prostředí. Výsledky získané v jakémkoli jiném operačním prostředí se proto mohou výrazně lišit. Některá měření mohla být prováděna na vývojových verzích systémů a není zaručeno, že tato měření budou stejná i na běžně dostupných systémech. Některá měření mohla být navíc odhadnuta pomocí extrapolace. Skutečné výsledky mohou být jiné. Čtenáři tohoto dokumentu by měli zjistit použitelné údaje pro své specifické prostředí.

Informace týkající se produktů jiných výrobců pocházejí od dodavatelů těchto produktů, z jejich veřejných oznámení nebo z jiných veřejně dostupných zdrojů. Společnost IBM tyto produkty netestovala a nemůže potvrdit správný výkon, kompatibilitu ani žádné jiné výroky týkající se produktů jiných výrobců než IBM. Otázky týkající se kompatibility produktů jiných výrobců by měly být směrovány dodavatelům těchto produktů.

Veškerá tvrzení týkající se budoucího směru vývoje nebo záměrů společnosti IBM se mohou bez upozornění změnit nebo mohou být zrušena a reprezentují pouze cíle a plány společnosti.

Tyto údaje obsahují příklady dat a sestav používaných v běžných obchodních operacích. Aby byla představa úplná, používají se v příkladech jména osob a názvy společností, značek a produktů. Všechna tato jména a názvy jsou fiktivní a jejich podobnost se jmény, názvy a adresami používanými ve skutečnosti je zcela náhodná.

LICENČNÍ INFORMACE:

Tyto informace obsahují ukázkové aplikační programy ve zdrojovém jazyce ilustrující programovací techniky na různých operačních platformách. Tyto ukázkové programy můžete bez závazků vůči společnosti IBM jakýmkoli způsobem kopírovat, měnit a distribuovat za účelem vývoje, používání, odbytu či distribuce aplikačních programů odpovídajících rozhraní API pro operační platformu, pro kterou byly ukázkové programy napsány. Tyto příklady nebyly plně testovány za všech podmínek. Společnost IBM proto nemůže zaručit spolehlivost, upotřebitelnost nebo funkčnost těchto programů.

Při prohlížení těchto dokumentů v elektronické podobě se nemusí zobrazit všechny fotografie a barevné ilustrace.

Informace o programovacím rozhraní

Informace o programovacím rozhraní, jsou-li poskytnuty, jsou určeny k tomu, aby vám pomohly vytvořit aplikační software pro použití s tímto programem.

Tato příručka obsahuje informace o zamýšlených programovacích rozhraních, která zákazníkům umožňují psát programy za účelem získání služeb produktu WebSphere MQ.

Tyto informace však mohou obsahovat i diagnostické údaje a informace o úpravách a ladění. Informace o diagnostice, úpravách a vyladění jsou poskytovány jako podpora ladění softwarových aplikací.

Důležité: Tyto informace o diagnostice, úpravách a ladění nepoužívejte jako programovací rozhraní, protože se mohou měnit.

Ochranné známky

IBM, logo IBM , ibm.com, jsou ochranné známky společnosti IBM Corporation, registrované v mnoha jurisdikcích po celém světě. Aktuální seznam ochranných známek společnosti IBM je k dispozici na webu "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml. Další názvy produktů a služeb mohou být ochrannými známkami společnosti IBM nebo jiných společností.

Microsoft a Windows jsou ochranné známky společnosti Microsoft Corporation ve Spojených státech a případně v dalších jiných zemích.

UNIX je registrovaná ochranná známka skupiny The Open Group ve Spojených státech a případně v dalších jiných zemích.

Linux je registrovaná ochranná známka Linuse Torvaldse ve Spojených státech a případně v dalších jiných zemích.

Tento produkt zahrnuje software vyvinutý projektem Eclipse (https://www.eclipse.org/).

Java a všechny ochranné známky a loga založené na termínu Java jsou ochranné známky nebo registrované ochranné známky společnosti Oracle anebo příbuzných společností.

Číslo položky:

(1P) P/N: